

Legnagyobb közös osztó

Számelmélet

Diszkrét modellek alkalmazásai

1. Definíciók és tulajdonságai

1.1. Definíció. Az $a \in \mathbb{Z}$ és $b \in \mathbb{Z}$ számok *legnagyobb közös osztója* az a $c \in \mathbb{N}$ szám, amelyre $c \mid a \wedge c \mid b \wedge \forall d \in \mathbb{N} : d \mid a \wedge d \mid b \implies d \mid c$. Jelölés: $\text{lko}(a, b)$ vagy (a, b) . Angolul: *greatest common divisor*, $\text{gcd}(a, b)$.

Szavakkal megfogalmazva: a és b legnagyobb közös osztója az a c szám, amely közös osztójuk, de bármely d , amely szintén közös osztójuk, osztja c -t.

Például 12 és 20 legnagyobb közös osztója 4, mert közös osztó ($4 \mid 12$ és $4 \mid 20$), de bármilyen d számra, amely szintén közös osztó (ilyenek az 1, 2, 4), $d \mid 4$.

A legnagyobb közös osztónak van egy párja:

1.2. Definíció. Az $a \in \mathbb{Z}$ és $b \in \mathbb{Z}$ számok *legkisebb közös többszöröse* az a $c \in \mathbb{N}$ szám, amelyre $a \mid c \wedge b \mid c \wedge \forall d \in \mathbb{N} : a \mid d \wedge b \mid d \implies c \mid d$. Jelölés: $\text{lkkt}(a, b)$ vagy $[a, b]$. Angolul: *least common multiple*, $\text{lcm}(a, b)$.

Szavakkal megfogalmazva: a és b legkisebb közös többszöröse az a c szám, amely közös többszörösük, de bármely d , amely szintén közös többszörösük, többszöröse c -nek.

Például 12 és 20 legkisebb közös többszöröse 60, mert közös többszörös ($12 \mid 60$ és $20 \mid 60$), de bármilyen d számra, amely szintén közös többszörös (pl. 60, 120, 180, 240, ...), $60 \mid d$.

(Megjegyzés: a "legnagyobb" és a "legkisebb" nem a szokásos értelemben ($\leq, \geq$) értendő, hanem az oszthatóság mint részbenrendezés szerint, lásd az Oszthatóság jegyzetben az asszociáltak utáni egyik megjegyzést. Ez pozitív számokra egybeesik a szokásos rendezéssel, de például 0-ra nem, amely itt a "legnagyobb", azaz pl. $\text{lko}(0, 0) = 0$, hiába közös osztó minden egész szám.)

Az lko és az lkkt bármely két egész számra létezik és egyértelmű. (Megjegyzés: az egyértelműség azért van, mert $\mathbb{N}$ -re szűkítettük le c -t, ahol az oszthatóság antiszimmetrikus. Néha a definícióban megengedik a negatív számokat is, de akkor az egyértelműség elvész, például akkor 6 és 4 legnagyobb közös osztója 2 és -2 is lehetne.)

Sage-ben a legnagyobb közös osztót a $\text{gcd}(a, b)$ függvénnyel, a legkisebb közös többszöröst pedig az $\text{lcm}(a, b)$ -vel lehet kiszámítani.

Az lko és az lkkt főbb tulajdonságai:

1. azonos számokra: $\forall a \in \mathbb{N} : \text{lko}(a, a) = \text{lkkt}(a, a) = a$;
2. nullával: $\forall a \in \mathbb{N} : \text{lko}(a, 0) = a \wedge \text{lkkt}(a, 0) = 0$;
3. osztóval: $\forall a, b \in \mathbb{N} : a \mid b \implies \text{lko}(a, b) = a \wedge \text{lkkt}(a, b) = b$;
4. negatív számokra: $\forall a, b \in \mathbb{Z} : \text{lko}(a, b) = \text{lko}(|a|, |b|) \wedge \text{lkkt}(a, b) = \text{lkkt}(|a|, |b|)$;
5. kommutatív: $\forall a, b \in \mathbb{Z} : \text{lko}(a, b) = \text{lko}(b, a) \wedge \text{lkkt}(a, b) = \text{lkkt}(b, a)$;

6. asszociatív: $\forall a, b, c \in \mathbb{Z} : \text{lko}(a, \text{lko}(b, c)) = \text{lko}(\text{lko}(a, b), c) \wedge$
 $\text{lkkt}(a, \text{lkkt}(b, c)) = \text{lkkt}(\text{lkkt}(a, b), c);$
7. $\forall a, b \in \mathbb{Z} : \text{lko}(a + b, b) = \text{lko}(a, b) \wedge \text{lko}(a - b, b) = \text{lko}(a, b)$ (ez csak az lko-ra igaz);
8. kiemelés: $\forall a, b, m \in \mathbb{N} : \text{lko}(ma, mb) = m \cdot \text{lko}(a, b) \wedge \text{lkkt}(ma, mb) = m \cdot \text{lkkt}(a, b);$
9. korlátok: $\forall a, b \in \mathbb{N}^+ : 1 \leq \text{lko}(a, b) \leq \min(a, b) \wedge \max(a, b) \leq \text{lkkt}(a, b) \leq ab;$
10. $\forall a, b \in \mathbb{N} : \text{lko}(a, b) \cdot \text{lkkt}(a, b) = ab.$

Az utolsó előtti tulajdonságból látszik, ahogy az lko lehetséges legkisebb értéke 1. Ha ez a helyzet, annak külön neve is van:

1.3. Definíció. Két egész szám *relatív prím*, ha a legnagyobb közös osztójuk 1.

Például 5 és 8 relatív prímek, mert $\text{lko}(5, 8) = 1$.

Relatív prímek esetén az lkkt is különleges: a fenti utolsó tulajdonságból következik, hogy akkor $\text{lkkt}(a, b) = ab$, ami az lkkt-re a létező legnagyobb érték.

Speciális esetekben könnyű eldönteni, hogy két szám relatív prím-e:

- Az 1 és a -1 minden egész számhoz relatív prím.
- Szomszédos számok relatív prímek (pl. 8 és 9).
- Egy prímszám minden olyan egész számhoz relatív prím, amely nem a többszöröse. (Erre azt is mondhatjuk, hogy a "prím" erősebb fogalom, mint a "relatív prím".)
- A 0 csak az 1-hez és a -1 -hez relatív prím.

1.1. Általánosítás több számra

A legnagyobb közös osztót és a legkisebb közös többszöröst nemcsak két számra lehet értelmezni, hanem akárannyire:

1.4. Definíció. Tetszőleges $a_1, a_2, \dots, a_n \in \mathbb{Z}$ számokra

- $\text{lko}(a_1, a_2, \dots, a_n) := \text{lko}(a_1, \text{lko}(a_2, \text{lko}(\dots, a_n) \dots))$ és
- $\text{lkkt}(a_1, a_2, \dots, a_n) := \text{lkkt}(a_1, \text{lkkt}(a_2, \text{lkkt}(\dots, a_n) \dots)).$

Például $\text{lko}(10, 12, 16) = \text{lko}(10, \text{lko}(12, 16)) = \text{lko}(10, 4) = 2$.

Sage-ben a `gcd()` és az `lcm()` függvények több számra is használhatók, de akkor azokat listában (vagy egyéb sorozattal) kell megadni, pl. `gcd([10, 12, 16])`, vagy `lcm(range(1, 1000))`.

Jobban kell vigyáznunk, ha a relatív prímek fogalmát szeretnénk általánosítani, mert erre két külön fogalom is van:

1.5. Definíció. $a_1, a_2, \dots, a_n \in \mathbb{Z}$ *relatív prímek*, ha $\text{lko}(a_1, a_2, \dots, a_n) = 1$.

1.6. Definíció. $a_1, a_2, \dots, a_n \in \mathbb{Z}$ *páronként relatív prímek*, ha $\text{lko}(a_i, a_j) = 1$ bármely különböző i -re és j -re.

Például 8, 10 és 15 relatív prímek, mert $\text{lko}(8, 10, 15) = \text{lko}(8, 5) = 1$, de nem *páronként* relatív prímek, mert nem minden páronkénti lko 1: $\text{lko}(8, 10) = 2$, $\text{lko}(8, 15) = 1$ és $\text{lko}(10, 15) = 5$. Viszont például 5, 8 és 9 páronként (is) relatív prímek, mert $\text{lko}(5, 8) = \text{lko}(5, 9) = \text{lko}(8, 9) = 1$. Megjegyzés: a páronként relatív prímek mindig relatív prímek is.

1.2. Kiszámítás prímfelbontásból

Ha ismerjük a két szám prímtényezős felbontását, akkor abból könnyen kiszámíthatjuk a legnagyobb közös osztójukat és a legkisebb közös többszörösüket:

1.7. Tétel. $\text{lko}(a, b)$ prímtényezői felbontásában pontosan azok a prímtényezők szerepelnek, amelyek a -ban és b -ben is szerepelnek, és a kettő közül a kisebbik kitevővel.

1.8. Tétel. $\text{lkt}(a, b)$ prímtényezői felbontásában pontosan azok a prímtényezők szerepelnek, amelyek vagy a -ban, vagy b -ben szerepelnek, és a kettő közül a nagyobbik kitevővel.

Például $120 = 2^3 \cdot 3 \cdot 5$ és $36 = 2^2 \cdot 3^2$, ezért $\text{lko}(120, 36) = 2^2 \cdot 3 = 12$ és $\text{lkt}(120, 36) = 2^3 \cdot 3^2 \cdot 5 = 360$. Ez a számítás azonban feltételezi, hogy ismerjük a két szám prímfelbontását, amit nagy számok esetén nagyon nehéz kiszámítani. Ezért szükségünk lesz egy hatékonyabb algoritmusra.

2. Euklideszi algoritmus

Az euklideszi algoritmus egy hatékony eljárás a legnagyobb közös osztó kiszámítására.

Az alapötlet az, hogy ha az egyik számból kivonjuk a másikat, akkor a legnagyobb közös osztó nem változik: $\text{lko}(a, b) = \text{lko}(a - b, b)$. Ez azért hasznos, mert ha $a > b > 0$, akkor $a - b < a$, vagyis csökkentettük az egyik számot. Ezt a helyettesítést (felváltva a és b szerepével) addig csinálhatjuk, amíg mindkét szám elég kicsi nem lesz. Például:

$$\begin{aligned}\text{lko}(50, 22) &= \\ \text{lko}(28, 22) &= \\ \text{lko}(6, 22) &= \\ \text{lko}(6, 16) &= \\ \text{lko}(6, 10) &= \\ \text{lko}(6, 4) &= \\ \text{lko}(2, 4) &= \\ \text{lko}(2, 2) &= 2.\end{aligned}$$

A (naiv) algoritmus tehát:

1. Bemenet: $a, b \in \mathbb{N}^+$.
2. Ciklus amíg $a \neq b$:
 - (a) Ha $a > b$, akkor:
 - $a := a - b$,
 - (b) egyébként:
 - $b := b - a$.

3. Kimenet: a .

(Megjegyzés: ez így csak pozitív egész számokra működik. Ha valamelyik bemenő szám nulla, akkor rögtön tudjuk, hogy az lko a másik szám lesz. Ha valamelyik szám negatív, akkor vegyük az ellentettjét, mielőtt elindítjuk az algoritmust.)

Ez azonban még nem a leghatékonyabb formája az euklideszi algoritmusnak. Ha ugyanis az egyik szám jóval nagyobb, mint a másik, akkor szokszor vonjuk le belőle a másik számot (pl. 22-ből a 6-ot: $22 \rightarrow 16 \rightarrow 10 \rightarrow 4$). Ezeket a kivonásokat összevonhatjuk egyetlen műveletté, egy maradékos osztássá, például $22 \bmod 6 = 4$. Ezzel a gyorsítással a fenti számítás kevesebb lépésben végezhető el:

$$\begin{aligned}
\text{luko}(50, 22) &= \\
\text{luko}(6, 22) &= \\
\text{luko}(6, 4) &= \\
\text{luko}(2, 4) &= \\
\text{luko}(2, 0) &= 2.
\end{aligned}$$

Ezt rekurzívan úgy definiálhatjuk, hogy először vegyük a két bemenő számot csökkenő sorrendben, azaz legyen $r_1 := a$ és $r_2 := b$, ha $a > b$ (különben fordítva), és minden lépésben kiszámítjuk az eddigi utolsó két szám maradékát, azaz $r_k := r_{k-2} \bmod r_{k-1}$, egészen addig, amíg el nem érjük a nullát, és akkor az előző szám lesz a legnagyobb közös osztó, azaz ha $r_n = 0$, akkor $\text{luko}(a, b) = r_{n-1}$. Például: $r_1 = 50$, $r_2 = 22$, $r_3 = 6$, $r_4 = 4$, $r_5 = 2$, $r_6 = 0$.

Az algoritmusban mindig csak az utolsó két számot tároljuk (a és b helyén), azaz ha kezdetben $a = 50$ és $b = 22$, akkor a következő lépésben $a = 22$ és $b = 6$, és így tovább.

A teljes algoritmus tehát:

1. Bemenet: $a, b \in \mathbb{N}$.
2. Ciklus amíg $b \neq 0$:
 - (a) $t := b$,
 - (b) $b := a \bmod b$,
 - (c) $a := t$.
3. Kimenet: a .

Megjegyzés: a ciklusmag három utasítása helyett írhattuk volna azt is, hogy $a := a \bmod b$, majd cseréljük fel az a -t és a b -t – ezt a cserét oldottuk meg az ideiglenes t változóval. De megúszhatjuk a cserét, ha a programozási nyelv támogatja a *szimultán értékadást*, azaz egyszerre több változó értékét is meg tudjuk változtatni (pl. ilyen a Python és a Sage): ekkor az egész ciklusmagot írhatjuk egyetlen utasítással úgy is, hogy $a, b := b, a \bmod b$, azaz legyen a az előző b , és egyúttal legyen b az előző értékekből számított $a \bmod b$. Például Sage-ben: `a, b = b, a % b`.

(Megjegyzés 2: ez a verzió már nullára is működik. Negatív számok esetén továbbra is abszolút értéket kell venni – akár a bemenetekét, akár a végeredményét.)

(Megjegyzés 3: fent kikötöttük, hogy $a > b$. Akkor miért nem ellenőriztük ezt a bemenő számokra, és miért nem cseréltük fel őket, ha nem igaz? Mert ez valójában csak egy kényelmi kikötés volt: ha $a < b$, akkor is működik az algoritmus, csak eggyel több lépésben, és az első lépés pont felcseréli a két számot, mert például $22 \bmod 50 = 22$.)

Bebizonyítható, hogy az euklideszi algoritmus nagy számokra nagyságrendileg $\log \min(a, b)$ lépést végez. Ez sokkal gyorsabb, mint bármilyen ismert prímfaktorizációs algoritmus.

És hogyan lehet hatékonyan kiszámítani a legkisebb közös többszöröst? Ehhez először kiszámítjuk az luko -t az euklideszi algoritmussal, majd $\text{lkk}(a, b) = ab / \text{luko}(a, b)$ a fenti legutolsó tulajdonság alapján (vagy ha negatív számok is lehetnek: $\text{lkk}(a, b) = |ab| / \text{luko}(a, b)$).

3. Bővített euklideszi algoritmus

3.1. Tétel (Bézout-lemma). $\forall a, b \in \mathbb{Z} : \exists x, y \in \mathbb{Z} : ax + by = \text{luko}(a, b)$.

Ez azt jelenti, hogy bármely két szám legnagyobb közös osztója előállítható a két szám egész

együtthatós lineáris kombinációjaként. Például $\text{lko}(16, 10) = 2$, és valóban, például $2 = 2 \cdot 16 - 3 \cdot 10$, azaz itt az együtthatók lehetnek $x = 2$ és $y = -3$. Szemléletesen, ha egy számegyenesen lépkedünk, de egyszerre mindig csak 16 vagy 10-hosszú lépéseket tehetünk, akkor el tudunk jutni a 0-ból a 2-be úgy, hogy jobbra lépünk kétszer 16-ot, majd balra háromszor 10-et.

3.2. Definíció. A fenti tételben szereplő x -et és y -t a és b *Bézout-együtthatóinak* nevezzük.

Megjegyzés: a Bézout-együtthatók nem egyértelműek, lásd a lineáris diofantikus egyenleteket később. Sage-ben a Bézout-együtthatókat az `xgcd(a,b)` függvénnyel számíthatjuk ki, amely visszaad egy (g, x, y) hármast, ahol g a legnagyobb közös osztó, és x és y a Bézout-együtthatók. Például: `xgcd(16,10) == (2,2,-3)`.

De hogyan működik az `xgcd()` függvény, ill. hogyan számíthatjuk ki ezeket az együtthatókat a Sage segítségével nélkül?

Erre az euklideszi algoritmus egy módosítását használjuk, a *bővített euklideszi algoritmust*. Ez amellett, hogy kiszámítja a legnagyobb közös osztót, közben további értékeket is számol, amelyek végül a kívánt x és y együtthatók lesznek. Ezek minden lépésben az eredeti a és b *valamilyen* együtthatói, de $ax + by$ kezdetben nem $\text{lko}(a,b)$ -t állítja elő, hanem az algoritmus éppen aktuális r_i értékét. Mivel az első két érték a két bemenő szám: $r_1 = a$ és $r_2 = b$, ezért az első két egyenlet triviális: $r_1 = a = 1 \cdot a + 0 \cdot b$ és $r_2 = b = 0 \cdot a + 1 \cdot b$. Ezután az eredeti algoritmus szerint $r_3 = r_1 \bmod r_2$, vagy másképpen $r_3 = r_1 - q \cdot r_2$, ahol q a maradékos osztás hányadosa. A bővített algoritmus az utóbbi összefüggést használja, de nemcsak az r_i számokra, hanem a teljes egyenletre, azaz a $r_1 = \dots$ fenti egyenletből kivonja az $r_2 = \dots$ egyenlet q -szorosát.

Például 50 és 22 bővített euklideszi algoritmus során az együtthatók így fognak változni:

$$\begin{array}{rcll} a = r_1 = 50 & = & 1 \cdot 50 & + 0 \cdot 22 \\ b = r_2 = 22 & = & 0 \cdot 50 & + 1 \cdot 22 \quad (-2 \cdot) \\ \hline r_3 = 6 & = & 1 \cdot 50 & - 2 \cdot 22 \quad (-3 \cdot) \\ r_4 = 4 & = & -3 \cdot 50 & + 7 \cdot 22 \quad (-1 \cdot) \\ r_5 = 2 & = & 4 \cdot 50 & - 9 \cdot 22 \quad (-2 \cdot) \\ r_6 = 0 & = & -11 \cdot 50 & + 25 \cdot 22 \end{array}$$

Minden lépésben zárójelben jeleztük $-q$ -t, azaz hogy az egyenlet hányszorosát vontuk le az előző egyenletből, hogy megkapjuk a következőt. A megoldást az utolsó előtti sorból olvashatjuk le: $\text{lko}(50, 22) = r_5 = 2$, a Bézout-együtthatók pedig $x = 4$ és $y = -9$.

Nem kell mindig felírni az egyenleteket, elég az együtthatókat és a bal oldalt (az első két együtthatópár mindig a 2×2 -es egységmátrix):

$$\begin{array}{c|ccc} 50 & 1 & 0 & \\ 22 & 0 & 1 & (-2) \\ \hline 6 & 1 & -2 & (-3) \\ 4 & -3 & 7 & (-1) \\ 2 & 4 & -9 & (-2) \\ 0 & -11 & 25 & \end{array}$$

(Megjegyzés: ha bal oldalt kijött a 0, akkor a hozzá tartozó együtthatókat már nem szükséges kiszámítani, mert az előző sorban megkaptuk a Bézout-együtthatókat. De mit kaptunk az utolsó sorban? Vegyük észre, hogy azok – előjeltől eltekintve – éppen az eredeti két szám, leosztva a legnagyobb közös osztójukkal. Ez később még hasznos lesz.)

A bővített euklideszi algoritmus tehát így működik (szimultán értékadásokkal, lásd fent):

1. Bemenet: $a, b \in \mathbb{N}$.
2. $r', r := a, b$,
3. $x', x := 1, 0$,
4. $y', y := 0, 1$.
5. Ciklus amíg $r \neq 0$:
 - (a) $q := r' \operatorname{div} r$,
 - (b) $r', r := r, r' - q \cdot r$ (ugyanaz, mint: $r', r := r, r' \bmod r$; de ez csak az r -ekre igaz),
 - (c) $x', x := x, x' - q \cdot x$,
 - (d) $y', y := y, y' - q \cdot y$.
6. Kimenet: r', x', y' .

(Megjegyzés: a "vesszős" változók (r', x' stb.) az előző lépésből való értékeket jelölik. Sage-ben, ahol nincs "vessző", ezeket másképp kell jelölni, pl. `r_old`, `x_old`, vagy `rr`, `xx` stb.)

4. Lineáris diofantikus egyenletek

A *diofantikus egyenlet* olyan egész együtthatós egyenlet, amelynek megoldásait az egész számok körében keressük. Ezek legegyszerűbb típusa a *lineáris* diofantikus egyenlet, amelynek legegyszerűbb formája a következő:

$$ax + by = c,$$

ahol $a, b, c \in \mathbb{Z}$ paraméterek és $x, y \in \mathbb{Z}$ az ismeretlenek, amelyekre szeretnénk megoldani az egyenletet. Például: $16x + 10y = 6$.

A következő kérdésekre keressük a választ:

1. Van-e megoldása az egyenletnek?
2. Ha van, hogyan tudjuk felírni egy megoldását?
3. Hány megoldása van?
4. Hogyan tudjuk felírni az összes megoldását?

Az előző szakaszban már megoldottunk egy speciális esetet: ha $c = \operatorname{lko}(a, b)$, akkor tudjuk, hogy létezik megoldás, és a bővített euklideszi algoritmussal ki is tudunk számítani egyet.

Az is könnyen belátható, hogy ha c az lko többszöröse, azaz $\operatorname{lko}(a, b) \mid c$, akkor is van megoldás, és azt megkaphatjuk az előző egyenlet ($ax + by = \operatorname{lko}(a, b)$) megoldásaiból egyszerű szorzással. Például ha tudjuk, hogy a $16x + 10y = 2$ egy megoldása $x = 2$ és $y = -3$, akkor a $16x + 10y = 6$ egy megoldása ennek a 3-szorosa lesz: $x = 6$ és $y = -9$.

De mi a helyzet, ha $\operatorname{lko}(a, b) \nmid c$? Például nézzük a $16x + 10y = 3$ egyenletet ($2 \nmid 3$). Nyilvánvaló, hogy ennek nincs megoldása, hiszen a bal oldal csak páros lehet, a jobb oldal pedig páratlan. Ez általában is igaz: $ax + by$ mindenképpen osztható $\operatorname{lko}(a, b)$ -vel (hiszen mind a , mind b osztható vele), ezért ha a jobb oldal nem, akkor nincs megoldás. Ezzel bebizonyítottuk a következő tételt:

4.1. Tétel. *Az $ax + by = c$ lineáris diofantikus egyenletnek akkor és csak akkor van megoldása, ha $\operatorname{lko}(a, b) \mid c$.*

A következő kérdés, hogy ha van megoldás, akkor hány van. Ehhez menjünk vissza a bővített euklideszi algoritmushoz, és vessünk egy pillantást az utolsó (0-s) sorra. Például $a = 50$ és $b = 22$ esetén az utolsó *előtti* sor az volt, hogy $2 = 4 \cdot 50 - 9 \cdot 22$, és ez adta meg a megoldást, de az *utolsó*

sora az volt, hogy $0 = -11 \cdot 50 + 25 \cdot 22$. Miért érdekes ez? Mert ha összeadjuk a két egyenletet: $2 = -7 \cdot 50 + 16 \cdot 22$, akkor szintén megoldást kapunk, hiszen a bal oldal a 0 miatt nem változott. Sőt, a $0 = \dots$ egyenlet kétszeresét, háromszorosát, vagy bármely egész számú többszörösét is hozzáadhatjuk az eredeti megoldáshoz, és így mindig újabb megoldásokat kapunk.

Ezzel beláttuk, hogy végtelen sok megoldás van. Sőt, az is bebizonyítható (ezt mellőzzük), hogy csak a fenti módon megkapható megoldások vannak.

Belátható az is, hogy az utolsó, $0 = ax_s + by_s$ alakú egyenletben $|x_s| = b/\text{lko}(a, b)$ és $|y_s| = a/\text{lko}(a, b)$ (pl. $11 = 22/2$ és $25 = 50/2$), előjelük pedig ellentétes (ha a és b pozitívak). Ezzel készen van a következő tétel:

4.2. Tétel. *Ha az $ax + by = c$ diofantikus egyenlet megoldható, és egy megoldásuk x_0 és y_0 , akkor az összes megoldás felírható a következőképpen: $x = x_0 - k \cdot \frac{b}{\text{lko}(a, b)}$ és $y = y_0 + k \cdot \frac{a}{\text{lko}(a, b)}$, ahol k tetszőleges egész szám.*

(Ezt úgy kaptuk, hogy az $ax_0 + by_0 = c$ megoldáshoz hozzáadtuk a fenti $ax_s + by_s = 0$ egyenlet k -szorosát. Vagy onnan is nézhetjük, hogy ha a tétel x és y megoldását visszahelyettesítjük az eredeti egyenletbe, akkor kiesnek a k -s részek, és visszakapjuk az $ax_0 + by_0 = c$ megoldást.)

Például tekintsük a következő egyenletet: $50x + 22y = 160$. A bővített euklideszi algoritmusból tudjuk, hogy $4 \cdot 50 - 9 \cdot 22 = 2$. Ha ezt megszorozzuk 80-nal, akkor megkapjuk az eredeti egyenlet egy megoldását: $320 \cdot 50 - 720 \cdot 22 = 160$, azaz $x_0 = 320$ és $y_0 = -720$. A fenti tétellel pedig megkapjuk az összes megoldását: $x = 320 - 11k$ és $y = -720 + 25k$, ahol $k \in \mathbb{Z}$. Például ha $k = 30$, akkor $x = -10$ és $y = 30$.

Végezetül: mi a helyzet akkor, ha egy lineáris diofantikus egyenletet kifejezetten $\mathbb{N}$ fölött szeretnénk megoldani, azaz a negatív megoldások nem érdekelnek? Ekkor, ha a paraméterek (a , b és c) szintén $\mathbb{N}$ -ben vannak, akkor csak véges sok megoldás lesz, hiszen a fenti képletben x és y ellenkező irányban változik k -val, tehát mindkét irányban valamelyik előbb-utóbb negatív lesz. Az is előfordulhat, hogy egyáltalán nincs megoldás. Az összes megoldást megkaphatjuk, ha az $x \geq 0$ és $y \geq 0$ egyenlőtlenségeket megoldjuk k -ra, és kiválogatjuk az egész megoldásokat.

Például a fenti $50x + 22y = 160$ egyenlet $\mathbb{N}$ fölötti megoldásait úgy számíthatjuk ki, hogy megoldjuk az $x = 320 - 11k \geq 0$ és az $y = -720 + 25k \geq 0$ egyenlőtlenségeket: $720/25 \leq k \leq 320/11$. Mivel k egész, ezért $\lceil 720/25 \rceil \leq k \leq \lfloor 320/11 \rfloor$, vagyis $29 \leq k \leq 29$. Itt tehát egyetlen megoldás van: $k = 29$, azaz $x = 1$ és $y = 5$.

5. Szerző

Ezt a jegyzetet Uray M. János írta, részben felhasználva Nagy Ádám jegyzetét:

<https://compalg.elte.gitlab-pages.hu/dimoa-web/tematika/dimoa>

Hibákat, észrevételeket itt lehet jelezni: uray.janos@inf.elte.hu.