2. Komputeralgebra (Járai Antal és Kovács Attila)

A különféle matematikai számítások elvégzésére képes informatikai eszközök nélkülözhetetlenek a modern tudományban és ipari technológiában. Képesek vagyunk kiszámolni bolygók, csillagok pályáját, vezérelni atomerőműveket, egyenletekkel leírni, modellezni a természet számos törvényét. Ezek a számítások alapvetően kétfélék lehetnek: *numerikusak és szimbolikusak*.

Ámbár a numerikus számítások nemcsak elemi aritmetikai műveleteket foglalnak magukban, hanem olyan műveleteket is, mint matematikai függvények numerikus értékének, polinomok gyökeinek vagy mátrixok sajátértékének meghatározása, ezek a műveletek alapvetően számokon értelmezettek, és ezek a számok a legtöbb esetben nem pontosak, pontosságuk az adott számítógépes architektúra lebegőpontos ábrázolási módjától függ. A szimbolikus számításokat matematikai vagy informatikai objektumokat jelentő szimbólumokon értelmezzük. Ezek az objektumok lehetnek számok (egészek, racionális számok, valós és komplex számok, algebrai számok), de lehetnek polinomok, racionális és trigonometrikus függvények, egyenletek, egyenletrendszerek, algebrai struktúrák elemei, vagy éppen halmazok, listák, táblázatok.

A szimbolikus számítások elvégzésére alkalmas számítógépes rendszereket (amelyek legtöbbször numerikus számításokra és az eredmények grafikus megjelenítésére egyaránt képesek) *komputeralgebra-rendszereknek* vagy *szimbolikus-algebrai rendszereknek* nevezzük. Az „algebra” szó a szimbolikus objektumokkal végzett műveletek algebrai eredetére utal.

A komputeralgebra-rendszerek mint *számítógépes programok* alapfeladata: (1) matematikai objektumok szimbolikus ábrázolása, (2) aritmetika ezekkel az objektumokkal. A komputeralgebra, mint *tudományterület* feladata pedig erre az aritmetikára épülő hatékony algoritmusok keresése, elemzése és megvalósítása tudományos kutatásokhoz és alkalmazásokhoz.

Mivel a komputeralgebra-rendszerek szimbolikusan, (lényegében) tetszőleges pontossággal és hibamentesen képesek számolni, először tisztázni kell, milyen adatszerkezeteket lehet hozzárendelni a különféle objektumokhoz. A 2.1. alfejezet a matematikai *objektumok ábrázolásának* problémakörét taglalja. A továbbiakban a szimbolikus algoritmusok közül ismertetjük azokat, melyek az idők folyamán a mindennapi tudomány és technika elengedhetetlen részévé váltak.

A természettudományok többsége jelenségeit, gondolatait matematikai egyenletekkel írja le. A lineáris egyenletek, egyenletrendszerek szimbolikus megoldásainak vizsgálata a jól ismert eliminációs módszereken alapul. A nemlineáris egyenletrendszerek megoldásai-

nak megkeresésére először megvizsgáljuk az *euklideszi algoritmus* különféle változatait és a *rezultánsmódszert*. A hatvanas évek közepén Buchberger doktori dolgozatában egy hatékony módszert dolgozott ki többváltozós polinomegyenletek szimbolikus megoldásainak meghatározására, amit ma *Gröbner-bázis elmélet* néven ismerünk. Buchberger munkájára csak évekkel később kezdtek felfigyelni, azóta a terület a komputeralgebra egyik legnépszerűbb ága. Ezekről lesz szó a 2.2. és a 2.3. alfejezetekben.

A következő bemutatandó terület a *szimbolikus integrálás*. Habár a probléma formális természete már több, mint 100 éve ismert (Liouville-elmélet), csak 1969-ben tudott Risch hatékony algoritmust adni annak eldöntésére, hogy ha adott egy valós elemi f függvény, akkor az $\int f(x)dx$ integrál is elemi-e és ha igen, az algoritmus meg is adja az integrál értékét. A módszerrel a 2.4. alfejezetben foglalkozunk.

A fejezet végén áttekintjük a szimbolikus algoritmusok elméleti és gyakorlati vonatkozásait (2.5. alfejezet), külön részt szánva napjaink komputeralgebra-rendszereinek.

2.1. Adatábrázolás

A komputeralgebrában a legkülönfélébb matematikai objektumok fordulnak elő. Ahhoz, hogy ezekkel az objektumokkal számolni lehessen, ábrázolni és tárolni kell őket a számítógép memóriájában. Ez elméleti és gyakorlati problémák egész sorát veti fel. Ebben az alfejezetben ezekről a kérdésekről lesz szó.

Tekintsük az *egészeket*. Egyrészt matematikai tanulmányainkból tudjuk, hogy halmazuk megszámlálható számosságú, viszont informatikai szemszögből azzal is tisztában vagyunk, hogy számítógépünk csak véges sok egész tárolására képes. Az, hogy mekkora a legnagyobb egész, amit minden további erőfeszítés nélkül ábrázolni tudunk, attól függ, hogy számítógépes architektúránkban mekkora a gépi szó mérete. Ez tipikusan $16, 32, 48$ vagy 64 bit. Az egy gépi szóban ábrázolható egészeket *egyszeres pontosságú egészeknek* nevezzük. Nem biztos, hogy egy tetszőleges egész szám elfér egy gépi szóban, vagyis olyan adatstruktúrára van szükség, ami több gépi szó felhasználásával tetszőlegesen nagy egész ábrázolására képes. Természetesen a „tetszőlegesen nagy” nem jelent „végtelen nagyot”, hiszen valamilyen tervezési igény vagy a memória mérete mindenképpen korlátot szab. Emellett olyan adatábrázolást kell megvalósítani, amelyre hatékony műveletek építhetők. Az egészek reprezentációjának alapvetően két útja van:

- ***helyiértékes*** (a hagyományos decimális számrendszerbeli ábrázolás általánosítása), amelyben egy n egészet $\sum_{i=0}^{k-1} d_i B^i$ alakban írunk fel, ahol a B alapszám akármilyen, egynél nagyobb egész lehet. A hatékonyság növelése miatt B-t úgy érdemes választani, hogy $B-1$ beleférjen egy gépi szóba. A d_i jegyek $(0 \le i \le k-1)$ vagy a kanonikus $\{0 \le d_i \le B-1\}$ vagy a szimmetrikus $\{-\lfloor B/2 \rfloor < d_i \le \lfloor B/2 \rfloor\}$ jegyhalmazból való egyszeres pontosságú egészek. Az így leírható *többszörös pontosságú egészek* lineáris listás $[d_0, d_1, \ldots, d_{k-1}]$ ábrázolásának számítógépes megvalósítása történhet dinamikusan vagy statikusan, attól függően, hogy a lineáris listát láncolt listaként vagy tömbként implementáljuk.

- ***moduláris,*** amelyben az n egész megfelelő számú, egyszeres pontosságú, páronként relatív prím modulusokkal vett moduláris képeinek lineáris listájaként adható meg. A moduláris képekből n a kínai maradéktétel segítségével rekonstruálható.

A moduláris alak gyorsabb az összeadás, kivonás és szorzás műveleteket tekintve, de lényegesen lassabb például oszthatósági vizsgálatoknál (amelyek sok esetben elkerülhetetlenek). Nyilvánvaló, hogy az adatstruktúra megválasztása erősen befolyásolja algoritmusaink sebességét.

2.1. példa. Az alábbi példában az egyszerűség kedvéért természetes számokkal dolgozunk. Tegyük fel, hogy olyan számítógép architektúránk van, ahol a gépi szó 32 bites, vagyis számítógépünk az $I_1 = [0, 2^{32} - 1] = [0, 4\,294\,967\,295]$ intervallum egészeivel képes egész aritmetikát végezni. Erre az aritmetikára építve az architektúránkon valósítsunk meg olyan egész aritmetikát, amellyel az $I_2 = [0, 10^{50}]$ intervallumban is számolni tudunk.

A helyiértékes ábrázoláshoz legyen $B = 10^4$, továbbá

$$\begin{aligned} n_1 &= 12345678909876543210123456789 0 , \\ n_2 &= 2110 . \end{aligned}$$

Ekkor

$$\begin{aligned} n_1 &= [7890, 3456, 1012, 5432, 9876, 7890, 3456, 12] , \\ n_2 &= [2110] , \\ n_1 + n_2 &= [0, 3457, 1012, 5432, 9876, 7890, 3456, 12] , \\ n_1 * n_2 &= [7900, 3824, 6049, 1733, 9506, 9983, 3824, 6049, 2] , \end{aligned}$$

ahol az összeadást és a szorzást helyiértékesen számoltuk.

A moduláris ábrázoláshoz válasszunk páronként relatív prím számokat az I_1 intervallumból úgy, hogy szorzatuk nagyobb legyen 10^{50}-nél. Legyenek például

$$\begin{aligned} m_1 = 4294967291,\ m_2 = 4294967279,\ m_3 = 4294967231 , \\ m_4 = 4294967197,\ m_5 = 4294967189,\ m_6 = 4294967161 \end{aligned}$$

prímek, ahol $\prod_{i=1}^{6} m_i > 10^{50}$. Egy I_2 intervallumbeli egészet tehát az I_1 intervallumból vett számhatossal ábrázolunk.

Ekkor

$$\begin{aligned} n_1 &\equiv 2009436698 \pmod{m_1}, \quad n_1 \equiv 961831343 \pmod{m_2} , \\ n_1 &\equiv 4253639097 \pmod{m_3}, \quad n_1 \equiv 1549708 \pmod{m_4} , \\ n_1 &\equiv 2459482973 \pmod{m_5}, \quad n_1 \equiv 3373507250 \pmod{m_6} , \end{aligned}$$

valamint $n_2 \equiv 2110 \pmod{m_i}$, $(1 \le i \le 6)$, vagyis

$$\begin{aligned} n_1 + n_2 &= [2009438808, 961833453, 4253641207, 1551818, 2459485083, 3373509360] , \\ n_1 * n_2 &= [778716563, 2239578042, 2991949111, 3269883880, 1188708718, 1339711723] , \end{aligned}$$

ahol az összeadás és a szorzás koordinátánként modulárisan elvégezve értendő.

Általánosabban, a matematikai objektumok ábrázolásának három absztrakciós szintjét érdemes megkülönböztetni:

1. *Az objektumok szintje.* Ezen a szinten az objektumok formális matematikai objektumoknak tekinthetők. Például $2+2$, $3*3-5$ és 4 ugyanazt az objektumot jelölik. Hasonlóan, az $(x+1)^2(x-1)$ és x^3+x^2-x-1 polinomok az objektumok szintjén azonosnak tekinthetők.

2. *A forma szintje.* Itt már megkülönböztetjük az objektumok eltérő ábrázolásait. Például az $(x+1)^2(x-1)$ és x^3+x^2-x-1 polinomok ugyanannak a polinomnak különböző reprezentációi, hiszen az előbbi egy szorzat, utóbbi egy összeg.
3. *Az adatstruktúra szintje.* Itt a számítógép memóriájában eltérő ábrázolásokat tekintjük különbözőknek. Az x^3+x^2-x-1 polinomnak ezen a szinten többféle reprezentációja is lehet, a polinomot leírhatja például

   - egy együtthatókból álló tömb: $[-1,-1,1,1]$,
   - egy láncolt lista: $[-1,0] \to [-1,1] \to [1,2] \to [1,3]$.

A különböző matematikai objektumok számítógépes ábrázolásához a komputeralgebra-rendszerek tervezőinek dönteniük kell mind a forma, mind az adatstruktúra szintjén. A döntést olyan kérdések nehezítik, mint a reprezentáció memóriaigénye, olvashatósága, vagy az ábrázolás számítási ideje. Például az

$$\begin{aligned} f(x) &= (x-1)^2(x+1)^3(2x+3)^4 \\ &= 16x^9-80x^8+88x^7+160x^6-359x^5+x^4+390x^3-162x^2-135x+81 \end{aligned}$$

polinom szorzat alakja kifejezőbb, mint az összeg alakja, de utóbbi előnyösebb, ha mondjuk az x^5-es tag együtthatójára vagyunk kíváncsiak. A forma szintjére vonatkozó döntési nehézségeket szemlélteti az alábbi példa:

- $x^{1000}-1$ és $(x-1)(x^{999}+x^{998}+\cdots+x+1)$,
- $(x+1)^{1000}$ és $x^{1000}+1000x^{999}+\cdots+1000x+1$.

A matematikai objektumok minden igényt kielégítő ábrázolására tökéletes módszert nem ismerünk. A gyakorlatban az objektumoknak különböző reprezentációi is megengedettek. Ez azt a problémát veti fel, hogy ugyanazon objektum eltérő ábrázolása esetén meg kell tudnunk állapítani azok egyenlőségét, konvertálni kell tudnunk egyik alakból a másikba és az egyértelmű ábrázoláshoz egyszerűsítéseket kell tudnunk azokon végrehajtani. A forma szintjén például minden egész számot felírhatunk valamely B alapú számrendszerben, míg az adatstruktúra szintjén a forma szintjén kapott lineáris listát láncolt listaként vagy tömbként reprezentálhatjuk.

A *racionális számok* egészek párjaiból, a számlálóból és a nevezőből állnak. Memória takarékosság érdekében, valamint két racionális szám könnyű összehasonlíthatósága miatt célszerű a számláló és a nevező legnagyobb közös osztójával egyszerűsített alakot ábrázolni. Mivel az egészek euklideszi gyűrűt alkotnak, a legnagyobb közös osztó euklideszi algoritmussal gyorsan számolható. Az ábrázolás egyértelműségéhez a nevezőt érdemes pozitívnak választani, a racionális szám előjele így a számláló előjele lesz.

A *többváltozós polinomok* (az $R[x_1,x_2,\ldots,x_n]$ n-változós polinomgyűrű elemei, ahol R gyűrű) $a_1x^{e_1}+a_2x^{e_2}+\cdots+a_nx^{e_n}$ alakúak, ahol $a_i \in R\setminus\{0\}$, $e_i=(e_{i_1},\ldots,e_{i_n})$ és x^{e_i}-t írtunk $x_1^{e_{i_1}}x_2^{e_{i_2}}\ldots x_n^{e_{i_n}}$ helyett. A forma szintjén az alábbi reprezentációs lehetőségek adódnak.

1. Kiterjesztett vagy faktorizált ábrázolás, ahol a polinom összegként vagy szorzatként jelenik meg:

   - x^2y-x^2+y-1 ,
   - $\left(x^2+1\right)(y-1)$.

2. Rekurzív vagy disztributív ábrázolás (csak többváltozós esetben). Például kétváltozós polinomgyűrűben $f(x,y)$ tekinthető $R[x,y]$-belinek, $(R[x])[y]$-belinek vagy $(R[y])[x]$-belinek:

   - $x^2y^2 + x^2 + xy^2 - 1$,
   - $(x^2 + x)y^2 + x^2 - 1$,
   - $(y^2 + 1)x^2 + y^2x - 1$.

Az adatstruktúra szintjén ritka vagy teljes reprezentáció lehetséges, például a ritka $x^4 - 1$ polinom teljes ábrázolása $x^4 + 0x^3 + 0x^2 + 0x - 1$. A gyakorlatban a többváltozós polinomok ritka ábrázolása a célravezető.

A $\sum_{i=0}^{\infty} a_i x^i$ alakú *hatványsorok* legegyszerűbb ábrázolása az, ha valamilyen véges rendig adjuk csak meg az együtthatók sorozatát, így lényegében egyváltozós polinomoknak tekintjük őket. Ezzel a megközelítéssel az a probléma, hogy különböző hatványsorokhoz tartozhat ugyanaz a reprezentáció. Ezt elkerülendő, a hatványsort az együtthatói sorozatát generáló függvénnyel szokás leírni. A generáló függvény egy olyan kiszámítható f függvény, amire $f(i) = a_i$. A hatványsorokkal végzett műveletekhez ekkor elegendő azt ismerni, hogy hogyan kell az operandusok együtthatóiból előállítani a művelet eredményét reprezentáló sorozat együtthatóit. Például az f és g hatványsorok szorzatát jelölő h hatványsor együtthatóit a $h_i = \sum_{k=0}^{i} f_k g_{i-k}$ függvény segítségével származtathatjuk. Ekkor a h_i együtthatókat csak akkor kell ténylegesen kiszámolni, ha szükség van az értékükre. Ezt a komputeralgebrában is gyakran használt technikát ***késleltetett kiértékelésnek*** nevezzük.

Mivel a komputeralgebra-rendszerek szimbolikusan számolnak, az algoritmusok *műveletigényének* vizsgálatán kívül mindig szükség van *memóriaigényük* vizsgálatára is, hiszen a memóriaigény befolyásolja a tényleges futási időt.[1] Tekintsünk egyszerű példaként egy n ismeretlenes, n egyenletből álló lineáris egyenletrendszert, ahol minden egyes egész együttható elfér a számítógép ω hosszúságú rekeszében. Kizárólag egész aritmetikát és Gauss-eliminációt alkalmazva a redukció eredményeként kapott együtthatók egyenként $2^{n-1}\omega$ tárhelyet igényelhetnek. Ha az együtthatók polinomok lennének és polinomaritmetikát használnánk, az eredmény polinomok együtthatóinak mérete, csakúgy mint a fokszámuk, exponenciális növekedést mutatna. A megfigyelt exponenciális növekedés ellenére a kapott végeredmény mégis „normális” méretű, hiszen a Cramer-szabály miatt a megoldások determinánsok hányadosaiként is megkaphatók, amelyek pedig közelítőleg csak $n\omega$ tárhelyet igényelnek. Ezt a jelenséget nevezzük ***köztes számítási tárrobbanásnak.*** Előfordulása gyakori a komputeralgebra algoritmusokban.

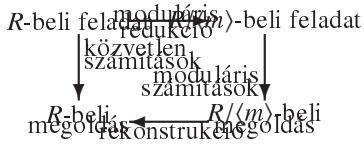
2.2. példa. Egész aritmetikát használva oldjuk meg az alábbi lineáris egyenletrendszert.

$$\begin{aligned} 37x + 22y + 22z &= 1 , \\ 31x - 14y - 25z &= 97 , \\ -11x + 13y + 15z &= -86 . \end{aligned}$$

Először a második egyenlet x változóját eliminálják. Szorozzuk meg az első sort 31-gyel, a másodikat -37-tel és adjuk össze őket. Ha ezt a módszert alkalmazzuk a harmadik egyenlet x változójának

[1] A futási időt mi a RAM-modellnek megfelelően műveletszámban mérjük. Ha Turing-gép modellt és konstans hosszú gépi szavakat használnánk, akkor ilyen probléma nem merülne fel, mert a tár mindig alsó korlátja az időnek.

unitlength 1mm



2.1. ábra. A moduláris algoritmusok általános sémája.

eliminációjára, az eredmény az alábbi lesz:

$$\begin{aligned} 37x + 22y + 22z &= 1\,, \\ 1200y + 1607z &= -3558\,, \\ 723y + 797z &= -3171\,. \end{aligned}$$

Most y eliminálásához a második sort 723-mal, a harmadikat -1200-zal szorozzuk, majd összeadjuk őket. Az eredmény:

$$\begin{aligned} 37x + 22y + 22z &= 1\,, \\ 1200y + 1607z &= -3558\,, \\ 205461z &= 1232766\,. \end{aligned}$$

Tovább folytatva az eljárást és sorban eliminálva a változókat végül azt kapjuk, hogy

$$\begin{aligned} 1874311479932400x &= 5622934439797200\,, \\ 246553200y &= -2712085200\,, \\ 205461z &= 1232766\,. \end{aligned}$$

Egyszerűsítés után $x = 3, y = -11, z = 6$ adódik. Természetesen, ha a számítások közben a legnagyobb közös osztókkal egyszerűsítünk, az együtthatók nagysága kevésbé drasztikusan nő.

A számítási tárrobbanás elkerülésére moduláris módszerek használatosak: ahelyett, hogy a számításainkat az R struktúra (pl. euklideszi gyűrű) egészeivel végeznénk, valamely faktorstruktúrában dolgozunk, majd az eredményt „visszatranszformáljuk” R-be (2.1. ábra). A moduláris redukció és a moduláris számítások általában hatékonyan elvégezhetők, a rekonstrukciós lépés pedig valamilyen interpolációs stratégiával történhet. Megjegyezzük, hogy a moduláris algoritmusok nagyon gyakoriak a komputeralgebrában, de nem univerzálisak.

2.2. Polinomok közös gyökei

Legyen R egy integritási tartomány, továbbá legyenek

$$f(x) = f_0 + f_1 x + \cdots + f_{m-1}x^{m-1} + f_m x^m \in R[x],\ f_m \neq 0\,, \tag{2.1}$$

$$g(x) = g_0 + g_1 x + \cdots + g_{n-1}x^{n-1} + g_n x^n \in R[x],\ g_n \neq 0 \tag{2.2}$$

tetszőleges polinomok, $n, m \in \mathbb{N}, n + m > 0$. Állapítsuk meg, hogy mi annak a szükséges és elégséges feltétele, hogy a két polinomnak legyen közös gyöke R-ben.

2.2.1. Klasszikus és bővített euklideszi algoritmus

Ha T test, akkor $T[x]$ euklideszi gyűrű. Emlékeztetőül, az R integritási tartományt euklideszi gyűrűnek nevezzük a $\varphi : R \setminus \{0\} \to \mathbb{N}$ euklideszi függvénnyel, ha bármely $a, b \in R$ $(b \neq 0)$ esetén létezik olyan $q, r \in R$, hogy $a = qb + r$, ahol $r = 0$ vagy $\varphi(r) < \varphi(b)$, továbbá minden $a, b \in R \setminus \{0\}$ esetén $\varphi(ab) \geq \varphi(a)$. A $q = a \operatorname{quo} b$ elemet ***hányadosnak,*** az $r = a \operatorname{rem} b$ elemet ***maradéknak*** nevezzük. Ha egy R euklideszi gyűrűben dolgozunk, azt szeretnénk, ha a legnagyobb közös osztó egyértelműen meghatározható lenne. Ehhez az R gyűrű egységszorzók által meghatározott ekvivalencia-osztályainak mindegyikéből egyetlen elem kiválasztása szükséges. (Például az egészek $\{0\}, \{-1, 1\}, \{-2, 2\}, \ldots$ osztályaiból mindig a nemnegatívat választjuk.) Így minden $a \in R$ egyértelműen írható fel

$$a = \operatorname{unit}(a) \cdot \operatorname{normal}(a)$$

alakban, ahol $\operatorname{normal}(a)$-t az a elem ***normálalakjának*** nevezzük. Tekintsünk egy T test feletti $R = T[x]$ euklideszi gyűrűt. Ekkor az $a \in R$ elem normálalakja legyen a megfelelő normált főpolinom, vagyis $\operatorname{normal}(a) = a/\operatorname{lc}(a)$, ahol $\operatorname{lc}(a)$ jelenti az a polinom főegyütthatóját. Foglaljuk össze a lényegesebb eseteket:

- ha $R = \mathbb{Z}$, akkor $\operatorname{unit}(a) = \operatorname{sgn}(a)$ $(a \neq 0)$ és $\varphi(a) = \operatorname{normal}(a) = |\, a \,|$,
- ha $R = T[x]$ (T test), akkor $\operatorname{unit}(a) = \operatorname{lc}(a)$ (az a polinom főegyütthatója a $\operatorname{unit}(0) = 1$ megállapodással), $\operatorname{normal}(a) = a/\operatorname{lc}(a)$ és $\varphi(a) = \deg a$.

Az alábbi algoritmus tetszőleges euklideszi gyűrűben kiszámítja két elem legnagyobb közös osztóját. Megjegyezzük, hogy a világ egyik legősibb algoritmusáról van szó, amit Euklidész már i. e. 300 körül ismert.

KLASSZIKUS-EUKLIDESZ(a, b)

```
1  c ← normal(a)
2  d ← normal(b)
3  while d ≠ 0
4        do r ← c rem d
5           c ← d
6           d ← r
7  return normal(c)
```

Az egészek gyűrűjében a 4. sor maradék képzése $c - \lfloor c/d \rfloor$-t jelenti. Ha $R = T[x]$, ahol T test, akkor a 4. sor maradék képzése az EGYHATÁROZATLANÚ-POLINOMOK-MARADÉKOS-OSZTÁSA(c, d) algoritmussal számolható, melynek elemzését az 2.2-1. gyakorlatra hagyjuk. A 2.2. ábra a KLASSZIKUS-EUKLIDESZ működését mutatja $\mathbb{Z}$-ben és $\mathbb{Q}[x]$-ben. Megjegyezzük, hogy $\mathbb{Z}$-ben a program a **while** ciklusba mindig nemnegatív számokkal lép be, a maradék képzés mindig nemnegatív számot eredményez, így a 7. sorban a normalizálás felesleges.

A KLASSZIKUS-EUKLIDESZ algoritmus futási idejének vizsgálata előtt annak egy bővített változatával foglalkozunk.

iteráció	r	c	d
–	–	18	30
1	18	30	18
2	12	18	12
3	6	12	6
4	0	6	0

(a) KLASSZIKUS-EUKLIDESZ$(-18, 30)$ működése.

iteráció	r	c	d
–	–	$x^4 - \frac{17}{3}x^3 + \frac{13}{3}x^2 - \frac{23}{3}x + \frac{14}{3}$	$x^3 - \frac{20}{3}x^2 + 7x - 2$
1	$4x^2 - \frac{38}{3}x + \frac{20}{3}$	$x^3 - \frac{20}{3}x^2 + 7x - 2$	$4x^2 - \frac{38}{3}x + \frac{20}{3}$
2	$-\frac{23}{4}x + \frac{23}{6}$	$4x^2 - \frac{38}{3}x + \frac{20}{3}$	$-\frac{23}{4}x + \frac{23}{6}$
3	0	$-\frac{23}{4}x + \frac{23}{6}$	0

(b) KLASSZIKUS-EUKLIDESZ$(12x^4 - 68x^3 + 52x^2 - 92x + 56, -12x^3 + 80x^2 - 84x + 24)$ működése.

2.2. ábra. A KLASSZIKUS-EUKLIDESZ algoritmus működésének bemutatása $\mathbb{Z}$-ben és $\mathbb{Q}[x]$-ben. Az **(a)** esetben a bemenő adatok $a = -18, b = 30, a, b \in \mathbb{Z}$. A pszeudokód első két sora a bemenő számok abszolút értékét számolja ki. A harmadik sortól az hatodik sorig tartó ciklus négyszer fut le, a különböző iterációkban számolt r, c és d értékeket mutatja a táblázat. A KLASSZIKUS-EUKLIDESZ$(-18,30)$ algoritmus eredményül a 6-ot szolgáltatja. A **(b)** esetben a bemenő paraméterek $a = 12x^4 - 68x^3 + 52x^2 - 92x + 56, b = -12x^3 + 80x^2 - 84x + 24 \in \mathbb{Q}[x]$. A program első két sora a polinomok normálalakját eredményezi, majd a **while** ciklus háromszor fut le. Az algoritmus kimenete a normal$(c) = x - 2/3$ polinom.

BŐVÍTETT-EUKLIDESZ(a, b)

```
1  (r_0, u_0, v_0) ← (normal(a), 1, 0)
2  (r_1, u_1, v_1) ← (normal(b), 0, 1)
3  while r_1 ≠ 0
4        do q ← r_0 quo r_1
5           r ← r_0 − q r_1
6           u ← (u_0 − q u_1)
7           v ← (v_0 − q v_1)
8           (r_0, u_0, v_0) ← (r_1, u_1, v_1)
9           (r_1, u_1, v_1) ← (r, u, v)
10 return (normal(r_0), u_0/(unit(a) · unit(r_0)), v_0/(unit(b) · unit(r_0)))
```

Ismert, hogy az R euklideszi gyűrűben az $a, b \in R$ elemek legnagyobb közös osztója alkalmas $u, v \in R$ elemekkel kifejezhető $\text{lnko}(a, b) = au + bv$ alakban. De nem csak egy ilyen számpár létezik. Ha ugyanis u_0, v_0 megfelelők, akkor $u_1 = u_0 + bt$ és $v_1 = v_0 - at$ is azok minden $t \in R$ esetén:

$$au_1 + bv_1 = a(u_0 + bt) + b(v_0 - at) = au_0 + bv_0 = \text{lnko}(a, b) \ .$$

A KLASSZIKUS-EUKLIDESZ algoritmust úgy egészítettük ki, hogy eredményül ne csak a legnagyobb közös osztót szolgáltassa, hanem az iméntieknek megfelelően egy konkrét $u, v \in R$ számpárt is megadjon.

Legyen $a, b \in R$, ahol R euklideszi gyűrű a φ euklideszi függvénnyel. A Bővített-Euklidesz pszeudokód első két sora kezdeti értékadásainak megfelelően az

$$r_0 = u_0 a + v_0 b \quad \text{és} \quad r_1 = u_1 a + v_1 b \tag{2.3}$$

egyenletek nyilván teljesülnek. Megmutatjuk, hogy a (2.3) egyenlőségek a pszeudokód **while** ciklusának transzformációira invariánsak. Tegyük fel, hogy a ciklus valamely iterációjának végrehajtása előtt a (2.3) feltételek teljesülnek. Ekkor a pszeudokód 4–5. sora szerint

$$r = r_0 - q r_1 = u_0 a + v_0 b - q(a u_1 + b v_1) = a(u_0 - q u_1) + b(v_0 - q v_1) \,,$$

amiből a 6–7. sorok miatt

$$r = a(u_0 - q u_1) + b(v_0 - q v_1) = au + bv.$$

A 8–9. sorok olyan értékadásokat jelentenek, melyben u_0, v_0 felveszi u_1 és v_1, majd u_1, v_1 felveszi u és v értékeit, továbbá r_0, r_1 felveszi r_1 és r értékét. Ezért (2.3) egyenlőségei a **while** ciklus kiértékelése után is teljesülnek. Mivel a ciklus újabb és újabb végrehajtásakor $\varphi(r_1) < \varphi(r_0)$, így a 8–9. sorok értékadásai során keletkezett $\{\varphi(r_i)\}$ sorozat a természetes számok szigorúan monoton csökkenő sorozatát alkotja, ezért a vezérlés előbb utóbb kilép a **while** ciklusból. A legnagyobb közös osztó az algoritmus maradékos osztás sorozatának utolsó nem nulla maradéka, a 8–9. soroknak megfelelően r_0.

2.3. példa. Vizsgáljuk meg a Bővített-Euklidesz algoritmus maradéksorozatát az

$$a(x) = 63x^5 + 57x^4 - 59x^3 + 45x^2 - 8 \,, \tag{2.4}$$

$$b(x) = -77x^4 + 66x^3 + 54x^2 - 5x + 99 \tag{2.5}$$

polinomok esetében:

$$\begin{aligned}
r_0 &= x^5 + \frac{19}{21}x^4 - \frac{59}{63}x^3 + \frac{5}{7}x^2 - \frac{8}{63} \,, \\
r_1 &= x^4 - \frac{6}{7}x^3 - \frac{54}{77}x^2 + \frac{5}{77}x - \frac{9}{7} \,, \\
r_2 &= \frac{6185}{4851}x^3 + \frac{1016}{539}x^2 + \frac{1894}{1617}x + \frac{943}{441} \,, \\
r_3 &= \frac{771300096}{420796475}x^2 + \frac{224465568}{420796475}x + \frac{100658427}{38254225} \,, \\
r_4 &= -\frac{125209969836038125}{113868312759339264}x - \frac{3541728593586625}{101216278008301568} \,, \\
r_5 &= \frac{4717580163635699927436051 21}{1803229860333151158054368 75} \,.
\end{aligned}$$

Az pszeudokód 10. sorának végrehajtása előtt az u_0, v_0 változók értékei:

$$\begin{aligned}
u_0 &= \frac{113868312759339264}{125209969836038125}x^3 - \frac{662639052858978333785656224}{819649936515068708206531 25}x^2 \\
&\quad - \frac{172214445262403690128205666 1}{901614930166575579027184375}x + \frac{145175798748706922498167 8954}{901614930166575579027184375} \,, \\
v_0 &= -\frac{113868312759339264}{125209969836038125}x^4 - \frac{650693816081118388788135 36}{819649936515068708206531 25}x^3 \\
&\quad + \frac{178270505434627626751446079}{819649936515068708206531 25}x^2 + \frac{638085922305129542614635 3}{819649936515068708206531 25}x \\
&\quad - \frac{179818001183413133012445617}{819649936515068708206531 25} \,.
\end{aligned}$$

A visszatérési értékek:

$$\begin{aligned}
\mathrm{lnko}(a,b) &= 1,\\
u &= \frac{2580775248128}{467729710968369}x^3 - \frac{3823697946464}{779549518280615}x^2\\
&\quad -\frac{27102209423483}{2338648554841845}x + \frac{7615669511954}{779549518280615},\\
v &= \frac{703847794944}{155909903656123}x^4 + \frac{3072083769824}{779549518280615}x^3\\
&\quad -\frac{25249752472633}{2338648554841845}x^2 - \frac{301255883677}{779549518280615}x + \frac{25468935587159}{2338648554841845}.
\end{aligned}$$

Láthatjuk, hogy az együtthatók drasztikus növekedést mutatnak. Felvetődik a kérdés: miért nem normalizálunk a **while** ciklus *minden* iterációjában? Ez az ötlet vezet el a polinomok euklideszi algoritmusa normalizált változatához.

Bővített-Euklidesz-normalizált(a, b)

```
1  e_0 ← unit(a)
2  (r_0, u_0, v_0) ← (normal(a), e_0^{-1}, 0)
3  e_1 ← unit(b)
4  (r_1, u_1, v_1) ← (normal(b), 0, e_1^{-1})
5  while r_1 ≠ 0
6        do q ← r_0 quo r_1
7           s ← r_0 rem r_1
8           e ← unit(s)
9           r ← normal(s)
10          u ← (u_0 − q u_1)/e
11          v ← (v_0 − q v_1)/e
12          (r_0, u_0, v_0) ← (r_1, u_1, v_1)
13          (r_1, u_1, v_1) ← (r, u, v)
14 return (r_0, u_0, v_0)
```

2.4. példa. Nézzük meg a Bővített-Euklidesz-normalizált algoritmus során keletkezett maradéksorozatot és az e együtthatósorozatot a korábbi (2.4), (2.5) polinomokra:

$$\begin{aligned}
r_0 &= x^5 + \frac{19}{21}x^4 - \frac{59}{63}x^3 + \frac{5}{7}x^2 - \frac{8}{63}, & e_0 &= 63,\\
r_1 &= x^4 - \frac{6}{7}x^3 - \frac{54}{77}x^2 + \frac{5}{77}x - \frac{9}{7}, & e_1 &= -77,\\
r_2 &= x^3 + \frac{9144}{6185}x^2 + \frac{5682}{6185}x + \frac{10373}{6185}, & e_2 &= \frac{6185}{4851},\\
r_3 &= x^2 + \frac{2338183}{8034376}x + \frac{369080899}{257100032}, & e_3 &= \frac{771300096}{420796475},\\
r_4 &= x + \frac{166651173}{5236962760}, & e_4 &= -\frac{222685475860375}{258204790837504},\\
r_5 &= 1, & e_5 &= \frac{156579848512133360531}{1097031157985072704 00}.
\end{aligned}$$

A pszeudokód 14. sorának végrehajtásakor az $\mathrm{lnko}(a,b) = r_0, u = u_0, v = v_0$ változók értékei:

$$\begin{aligned}
\mathrm{lnko}(a,b) &= 1,\\
u &= \frac{2580775248128}{467729710968369}x^3 - \frac{3823697946464}{779549518280615}x^2\\
&\quad -\frac{27102209423483}{2338648554841845}x + \frac{7615669511954}{779549518280615},\\
v &= \frac{703847794944}{155909903656123}x^4 + \frac{3072083769824}{779549518280615}x^3\\
&\quad -\frac{25249752472633}{2338648554841845}x^2 - \frac{301255883677}{779549518280615}x + \frac{25468935587159}{2338648554841845}.
\end{aligned}$$

$\mathbb{Q}[x]$-ben az együtthatók nagyságát tekintve az euklideszi algoritmus normalizált változatának előnye szembetűnő, de az együtthatók növekedését így sem kerültük el. A Bővített-Euklidesz-normalizált algoritmus gépi architektúra függő leírásához, elemzéséhez bevezetjük az alábbi jelölést. Legyen

$$\begin{aligned}
\lambda(a) &= \lfloor \log_2 |a|/w \rfloor + 1, \text{ ha } a \in \mathbb{Z}\setminus\{0\}, \text{ és } \lambda(0) = 0,\\
\lambda(a) &= \max\{\lambda(b),\lambda(c)\}, \text{ ha } a = b/c \in \mathbb{Q},\ b,c \in \mathbb{Z},\ \mathrm{lnko}(b,c) = 1,\\
\lambda(a) &= \max\{\lambda(b),\lambda(a_0),\ldots,\lambda(a_n)\}, \text{ ha } a = \sum_{0\le i\le n} a_i x^i/b \in \mathbb{Q}[x],\\
&\quad a_i \in \mathbb{Z},\ b \in \mathbb{N}^+,\ \mathrm{lnko}(b,a_0,\ldots,a_n) = 1,
\end{aligned}$$

ahol w a számítógépes architektúra szóhossza bitekben. Könnyű meggondolni, hogy ha $a,b \in \mathbb{Z}[x]$ és $c,d \in \mathbb{Q}$, akkor

$$\begin{aligned}
\lambda(c+d) &\le \lambda(c)+\lambda(d)+1,\\
\lambda(a+b) &\le \max\{\lambda(a),\lambda(b)\}+1,\\
\lambda(cd),\lambda(c/d) &\le \lambda(c)+\lambda(d),\\
\lambda(ab) &\le \lambda(a)+\lambda(b)+\lambda(\min\{\deg a,\deg b\}+1).
\end{aligned}$$

Az alábbi tételeket bizonyítás nélkül közöljük.

2.1. tétel. *Ha $a,b \in \mathbb{Z}$ és $\lambda(a) = m \ge n = \lambda(b)$, akkor a* Klasszikus-Euklidesz *és a* Bővített-Euklidesz *algoritmusok $O(mn)$ gépi szóban mért elemi aritmetikai műveletet igényelnek.*

2.2. tétel. *Ha F test, $a,b,\in F[x]$, $\deg(a) = m \ge n = \deg(b)$, akkor a* Klasszikus-Euklidesz, *a* Bővített-Euklidesz *és a* Bővített-Euklidesz-normalizált *algoritmusok $O(mn)$ F-beli elemi aritmetikai műveletet igényelnek.*

Vajon az együtthatók imént látott növekedése pusztán csak a példaválasztásból fakad? Vizsgáljunk meg a Bővített-Euklidesz-Normalizált algoritmusban egyetlen maradékos osztást. Legyen $a = bq + e^*r$, ahol

$$\begin{aligned}
a &= x^m + \frac{1}{c}\sum_{i=0}^{m-1} a_i x^i \in \mathbb{Q}[x],\\
b &= x^n + \frac{1}{d}\sum_{i=0}^{n-1} b_i x^i \in \mathbb{Q}[x],
\end{aligned}$$

$r \in \mathbb{Q}[x]$ főpolinomok, $a_i, b_i \in \mathbb{Z}$, $e^* \in \mathbb{Q}$, $c, d \in \mathbb{N}^+$, és tekintsük az $n = m - 1$ esetet. Ekkor

$$\begin{aligned} q &= x + \frac{a_{m-1}d - b_{n-1}c}{cd}\,, \\ \lambda(q) &\leq \lambda(a) + \lambda(b) + 1\,, \\ e^*r &= a - qb = \frac{acd^2 - xbcd^2 - (a_{m-1}d - b_{n-1}c)bd}{cd^2}\,, \\ \lambda(e^*r) &\leq \lambda(a) + 2\lambda(b) + 3\,. \end{aligned} \tag{2.6}$$

Vegyük észre, hogy a (2.6) becslés az r maradék polinom együtthatóira is érvényes, vagyis $\lambda(r) \leq \lambda(a) + 2\lambda(b) + 3$. Így $\lambda(a) \sim \lambda(b)$ esetén maradékos osztásonként az együtthatók mérete legfeljebb kb. háromszorosára nőhet. Pszeudovéletlen polinomokra a becslés élesnek tűnik, a kísérletezni vágyó Olvasónak ajánljuk a 2-1. feladatot. A legrosszabb esetre kapott becslés azt sejteti, hogy

$$\lambda(r_l) = O(3^l \cdot \max\{\lambda(a), \lambda(b)\}),$$

ahol l jelöli a Bővített-Euklidesz-Normalizált algoritmus futási idejét, vagyis lényegében azt, hogy a **while** ciklus hányszor hajtódik végre. Szerencsére, ez az exponenciális növekedés nem teljesül az algoritmus minden iterációjában, végeredményben pedig az együtthatók növekedése a bemenet függvényében polinomiálisan korlátos. A későbbiekben látni fogjuk, hogy moduláris technika alkalmazásával az együtthatók növekedése teljesen elkerülhető.

Összefoglalva, az euklideszi algoritmus segítségével az $f, g \in R[x]$ (R test) polinomok legnagyobb közös osztóját kiszámítva f-nek és g-nek pontosan akkor van közös gyöke, ha a legnagyobb közös osztójuk nem konstans. Ha ugyanis $\mathrm{lnko}(f, g) = d \in R[x]$ nem konstans, akkor d gyökei f-nek és g-nek is gyökei, hiszen d osztója f-nek és g-nek is. Megfordítva, ha f-nek és g-nek van közös gyöke, akkor a legnagyobb közös osztójuk nem lehet konstans, mert a közös gyök ennek is gyöke.

2.2.2. Primitív euklideszi algoritmus

Amennyiben R euklideszi gyűrű vagy alaptételes gyűrű (amelyben érvényes a számelmélet alaptételének megfelelő állítás, miszerint bármely nem nulla és nem egység elem sorrendtől és egységszorzóktól eltekintve egyértelműen bontható irreducibilis elemek szorzatára), akkor a helyzet bonyolultabb, hiszen $R[x]$-ben nem feltétlenül létezik euklideszi algoritmus. Szerencsére, mégis több módszer kínálkozik, melyek használhatóságának két oka van: (1) $R[x]$ alaptételes gyűrű, (2) alaptételes gyűrűben két vagy több elem legnagyobb közös osztója mindig létezik.

Az első kínálkozó módszer az, hogy a legnagyobb közös osztó számítását R hányadostestében végezzük el. A $p(x) \in R[x]$ polinomot ***primitív polinomnak*** nevezzük, ha nincs olyan R-beli prím, ami $p(x)$ összes együtthatóját osztaná. Gauss híres lemmája szerint primitív polinomok szorzata is primitív, melynek következménye, hogy f, g primitív polinomok esetén pontosan akkor lesz $d = \mathrm{lnko}(f, g) \in R[x]$, ha $d = \mathrm{lnko}(f, g) \in H[x]$, ahol H jelöli R hányadostestét. Vagyis az $R[x]$-beli legnagyobb közös osztó számítás visszavezethető $H[x]$-belire. Sajnos, ez a megközelítés nem igazán hatékony, mert a H hányadostestben használt aritmetika lényegesen költségesebb, mint az R-beli.

Második lehetőségként egy, az euklideszi algoritmushoz hasonló algoritmus segíthet: integritási tartomány feletti egyhatározatlanú polinomgyűrűben ún. pszeudo-maradékos osztást lehet definiálni. A (2.1), (2.2) polinomokat használva ha $m \geq n$, akkor létezik olyan

$q, r \in R[x]$, hogy

$$g_n^{m-n+1} f = gq + r \ ,$$

ahol $r = 0$ vagy $\deg r < \deg g$. A q polinomot az f és g polinomok ***pszeudo-hányadosának,*** az r polinomot ***pszeudo-maradékának*** nevezzük. Jelölésben $q = \mathrm{pquo}(f, g), r = \mathrm{prem}(f, g)$.

2.5. példa. Legyen

$$f(x) = 12x^4 - 68x^3 + 52x^2 - 92x + 56 \in \mathbb{Z}[x] \ , \tag{2.7}$$

$$g(x) = -12x^3 + 80x^2 - 84x + 24 \in \mathbb{Z}[x] \ . \tag{2.8}$$

Ekkor $\mathrm{pquo}(f, g) = -144(x + 1)$, $\mathrm{prem}(f, g) = 1152(6x^2 - 19x + 10)$.

Másrészt egységszorzótól eltekintve minden $f(x) \in R[x]$ polinom egyértelműen írható fel

$$f(x) = \mathrm{cont}(f) \cdot \mathrm{pp}(f)$$

alakban, ahol $\mathrm{cont}(f) \in R$ és $\mathrm{pp}(f) \in R[x]$ primitív polinom. Ekkor $\mathrm{cont}(f)$-et f ***összetevőjének,*** $\mathrm{pp}(f)$-et az $f(x)$ polinom ***primitív részének*** nevezzük. A felírások egyértelműsége az egységek normalizálásával érhető el. Például $\mathbb{Z}$-ben az egységek $\{-1, 1\}$ halmazából mindig a pozitívat választjuk.

Az alábbi algoritmus pszeudo-maradékos osztások sorozatát hajtja végre. Az algoritmus felhasználja a pszeudo-maradékot kiszámító prem() függvényt, feltételezi az R-beli legnagyobb közös osztó, valamint az $R[x]$-beli polinomok összetevőjének és primitív részének kiszámíthatóságát. Bemenete az $a, b \in R[x]$ polinomok, ahol R alaptételes gyűrű. Az algoritmus eredménye a $\mathrm{lnko}(a, b) \in R[x]$ polinom.

PRIMITÍV-EUKLIDESZ(a, b)

```
1  c ← pp(f)
2  d ← pp(g)
3  while d ≠ 0
4        do r ← prem(c, d)
5           c ← d
6           d ← pp(r)
7  γ ← lnko(cont(a), cont(b))
8  δ ← γc
9  return δ
```

Az algoritmus működését a 2.3. ábra szemlélteti. A PRIMITÍV-EUKLIDESZ algoritmus futási idejének nagyságrendje megegyezik az euklideszi algoritmus korábban látott változatainak futási idejével.

PRIMITÍV-EUKLIDESZ algoritmus azért nagyon lényeges, mert az R test feletti többváltozós $R[x_1, x_2, \ldots x_t]$ polinomgyűrű alaptételes gyűrű, így az algoritmust úgy alkalmazzuk, hogy kiszámoljuk a legnagyobb közös osztót mondjuk $R[x_2 \ldots, x_t][x_1]$-ben, majd rekurzívan az $R[x_3, \ldots, x_t], \ldots, R[x_t]$ alaptételes gyűrűkben. Vagyis a többváltozós polinomgyűrűk rekurzív szemlélete természetes módon vezet a PRIMITÍV-EUKLIDESZ algoritmus rekurzív alkalmazásához.

Észrevehetjük, hogy az algoritmus a korábban látottakhoz hasonlóan együttható növekedést mutat.

iteráció	r	c	d
–	–	$3x^4 - 17x^3 + 13x^2 - 23x + 14$	$-3x^3 + 20x^2 - 21x + 6$
1	$108x^2 - 342x + 108$	$-3x^3 + 20x^2 - 21x + 6$	$6x^2 - 19x + 10$
2	$621x - 414$	$6x^2 - 19x + 10$	$3x - 2$
3	0	$3x - 2$	0

2.3. ábra. A Primitív-Euklidesz algoritmus működésének bemutatása az $a(x) = 12x^4 - 68x^3 + 52x^2 - 92x + 56$, $b(x) = -12x^3 + 80x^2 - 84x + 24 \in \mathbb{Z}[x]$ bemenő adatok esetén. A program első két sora a bemeneti polinomok primitív részét számolja ki. A harmadik sortól a hatodik sorig tartó ciklus háromszor fut le, a különböző iterációkban számolt r, c és d értékeket mutatja a táblázat. A program 7. sorában a γ változó értéke $\mathrm{lnko}(4, 4) = 4$. A Primitív-Euklidesz(a, b) algoritmus eredményül $4 \cdot (3x - 2)$-t szolgáltat.

Vizsgáljuk meg részletesebben a $\mathbb{Z}[x]$ alaptételes gyűrűt. A legnagyobb közös osztó együtthatóinak nagyságára vonatkozó becslést az alábbi, bizonyítás nélkül közölt tétel mutatja.

2.3. tétel (Landau–Mignotte). *Legyen* $a(x) = \sum_{i=0}^{m} a_i x^i$, $b(x) = \sum_{i=0}^{n} b_i x^i \in \mathbb{Z}[x]$, $a_m \neq 0 \neq b_n$, *továbbá* $b(x) \mid a(x)$. *Ekkor*

$$\sum_{i=1}^{n} |b_i| \leq 2^n \left| \frac{b_n}{a_m} \right| \sqrt{\sum_{i=0}^{m} a_i^2} \, .$$

2.4. következmény. *Az előző tétel jelöléseivel az* $lnko(a,b) \in \mathbb{Z}[x]$ *polinom bármely együtthatója abszolút értékben kisebb, mint*

$$2^{\min\{m,n\}} \cdot lnko(a_m, b_n) \cdot \min\left\{\frac{1}{|a_m|}\sqrt{\sum_{i=1}^{m} a_i^2}, \frac{1}{|b_n|}\sqrt{\sum_{i=1}^{n} b_i^2}\right\}.$$

Bizonyítás. Az a és b polinomok legnagyobb közös osztója nyilván osztja a-t és b-t, a foka pedig legfeljebb az a és b polinomok fokainak minimuma. Továbbá a legnagyobb közös osztó főegyütthatója osztója a_m-nek és b_n-nek is, így $\mathrm{lnko}(a_m, b_n)$-nek is. ■

2.6. példa. A 2.4. következmény szerint a (2.4), (2.5) polinomok legnagyobb közös osztója bármely együtthatójának abszolút értéke legfeljebb $\lfloor 32/9\sqrt{3197}\rfloor = 201$, a (2.7), (2.8) polinomok esetében pedig legfeljebb $\lfloor 32\sqrt{886}\rfloor = 952$.

2.2.3. A rezultáns

Az alábbiakban ismertetendő módszer a legáltalánosabb keretek között tárgyalja az (2.1), (2.2) polinomok közös gyökeire vonatkozó szükséges és elégséges feltételeket. További előnye, hogy magasabb fokú algebrai egyenletrendszerek megoldására is alkalmazható.

Legyen tehát R egy integritási tartomány és H a hányadosteste. Tekintsük H-nak azt a legszűkebb K bővítését, melyben a (2.1)-beli $f(x)$ polinom és a (2.2)-beli $g(x)$ polinom is lineáris faktorokra bomlik. Jelöljük az $f(x)$ polinom (K-beli) gyökeit $\alpha_1, \alpha_2, \ldots, \alpha_m$-nel, a $g(x)$ polinom gyökeit pedig $\beta_1, \beta_2, \ldots, \beta_n$-nel. Készítsük el a következő szorzatot:

$$\begin{aligned}\mathrm{res}(f,g) &= f_m^n g_n^m (\alpha_1-\beta_1)(\alpha_1-\beta_2)\cdots(\alpha_1-\beta_n)\\ &\quad\cdot(\alpha_2-\beta_1)(\alpha_2-\beta_2)\cdots(\alpha_2-\beta_n)\\ &\quad\vdots\\ &\quad\cdot(\alpha_m-\beta_1)(\alpha_m-\beta_2)\cdots(\alpha_m-\beta_n)\\ &= f_m^n g_n^m \prod_{i=1}^{m}\prod_{j=1}^{n}(\alpha_i-\beta_j).\end{aligned}$$

Nyilvánvaló, hogy $\mathrm{res}(f,g)$ akkor és csak akkor lesz 0, ha valamilyen i-re és j-re $\alpha_i = \beta_j$, azaz ha f-nek és g-nek van közös gyöke. Ezt a $\mathrm{res}(f,g)$ szorzatot az f és g polinomok ***rezultánsának*** nevezzük. Vegyük észre, hogy a rezultáns értéke függ az f és g polinomok sorrendjétől, azonban a különböző sorrendben képzett rezultánsok legfeljebb csak előjelben térhetnek el egymástól:

$$\begin{aligned}\mathrm{res}(g,f) &= g_n^m f_m^n \prod_{j=1}^{n}\prod_{i=1}^{m}(\beta_j-\alpha_i)\\ &= (-1)^{mn} f_m^n g_n^m \prod_{i=1}^{m}\prod_{j=1}^{n}(\alpha_i-\beta_j) = (-1)^{mn}\mathrm{res}(f,g).\end{aligned}$$

A rezultánsnak ez az alakja a gyakorlatban természetesen használhatatlan, mivel a gyökök ismeretét tételezi fel. Vizsgáljuk meg tehát a rezultáns különböző alakjait. Mivel

$$\begin{aligned} f(x) &= f_m(x-\alpha_1)(x-\alpha_2)\cdots(x-\alpha_m) \quad (f_m \neq 0)\,, \\ g(x) &= g_n(x-\beta_1)(x-\beta_2)\cdots(x-\beta_n) \quad (g_n \neq 0)\,, \end{aligned}$$

ezért

$$\begin{aligned} g(\alpha_i) &= g_n(\alpha_i-\beta_1)(\alpha_i-\beta_2)\cdots(\alpha_i-\beta_n) \\ &= g_n\prod_{j=1}^{n}(\alpha_i-\beta_j)\,. \end{aligned}$$

Így

$$\begin{aligned} \operatorname{res}(f,g) &= f_m^n\prod_{i=1}^{m}\Bigl(g_n\prod_{j=1}^{n}(\alpha_i-\beta_j)\Bigr) \\ &= f_m^n\prod_{i=1}^{m}g(\alpha_i) = (-1)^{mn}g_n^m\prod_{j=1}^{n}f(\beta_j)\,. \end{aligned}$$

Ámbár ez az alak sokkal barátságosabb, még mindig feltételezi legalább az egyik polinom gyökeinek ismeretét. Az alábbiakban azt nézzük meg, hogyan lehetne a rezultánst pusztán csak a polinomok együtthatói segítségével kifejezni. Ez a vizsgálat vezet el a rezultáns Sylvester-féle alakjához.

Tegyük fel, hogy a (2.1)-beli f és a (2.2)-beli g polinomoknak van közös gyöke. Ez azt jelenti, hogy van olyan $\alpha \in K$ szám, amelyre

$$\begin{aligned} f(\alpha) &= f_m\alpha^m + f_{m-1}\alpha^{m-1} + \cdots + f_1\alpha + f_0 = 0\,, \\ g(\alpha) &= g_n\alpha^n + g_{n-1}\alpha^{n-1} + \cdots + g_1\alpha + g_0 = 0\,. \end{aligned}$$

A két egyenletet szorozzuk meg rendre az $\alpha^{n-1}, \alpha^{n-2}, \ldots, \alpha, 1$, illetve az $\alpha^{m-1}, \alpha^{m-2}, \ldots, \alpha, 1$ számokkal. Ekkor az első egyenletből n, a második egyenletből m újabb egyenletet nyerünk. Ezt az $m+n$ egyenletet fogjuk úgy fel, mint egy $m+n$ ismeretlenre vonatkozó homogén lineáris egyenletrendszert, melynek $\alpha^{m+n-1}, \alpha^{m+n-2}, \ldots, \alpha, 1$ a megoldása. A megoldás nyilván nem-triviális, hiszen 1 is a gyökök között szerepel. Ismert, hogy az olyan homogén lineáris egyenletrendszernek, amely ugyanannyi egyenletből áll, mint ahány ismeretlent tartalmaz, csak abban az esetben van nemtriviális megoldása, ha a rendszer determinánsa zérus. Vagyis arra jutottunk, hogy f-nek és g-nek csak akkor lehet közös gyöke, ha a

$$D = \begin{vmatrix} f_m & \cdots & \cdots & \cdots & f_0 & & & \uparrow \\ & \ddots & & & & \ddots & & n \\ & & f_m & \cdots & \cdots & \cdots & f_0 & \downarrow \\ g_n & \cdots & \cdots & g_0 & & & & \uparrow \\ & \ddots & & & \ddots & & & m \\ & & \ddots & & & \ddots & & \\ & & & g_n & \cdots & \cdots & g_0 & \downarrow \end{vmatrix} \tag{2.9}$$

determináns nulla (a ki nem írt és nem pontozott helyeken mindenütt nullák állnak). A közös gyök létezésének tehát szükséges feltétele, hogy az $(m+n)$-edrendű D determináns 0 legyen. Az alábbiakban bebizonyítjuk, hogy a D determináns megegyezik az f és g polinomok rezultánsával, amiből az következik, hogy $D = 0$ a közös gyökök létezésének elégséges feltétele is. A (2.9) determinánst nevezzük az f és g polinomok rezultánsa ***Sylvester-féle alakjának.***

2.5. tétel. *A korábbi jelölésekkel*

$$D = f_m^n \prod_{i=1}^{m} g(\alpha_i) \ .$$

Bizonyítás. m-re vonatkozó teljes indukcióval dolgozunk. $m = 0$-ra $f = f_m = f_0$, így a jobb oldal f_0^n. A bal oldalon D egy n-edrendű determináns, melynek a főátlójában csupa f_0 áll, a többi helyen pedig nulla. Így $D = f_0^n$, az állítás tehát igaz. A továbbiakban tegyük fel, hogy $m > 0$ és hogy a bizonyítandó állítás $n-1$-re igaz. Ha tehát f helyett az

$$f^*(x) = f_m(x-\alpha_1)\cdots(x-\alpha_{m-1}) = f^*_{m-1}x^{m-1} + f^*_{m-2}x^{m-2} + \cdots + f^*_1 x + f^*_0$$

polinomot vesszük, akkor f^*ra és g-re az állítás teljesül:

$$D^* = \begin{vmatrix} f^*_{m-1} & \cdots & \cdots & \cdots & f^*_0 & & \\ & \ddots & & & & \ddots & \\ & & f^*_{m-1} & \cdots & \cdots & \cdots & f^*_0 \\ g_n & \cdots & \cdots & g_0 & & & \\ & \ddots & & & \ddots & & \\ & & \ddots & & & \ddots & \\ & & & g_n & \cdots & \cdots & g_0 \end{vmatrix} = f^{*m}_{m-1} \prod_{i=1}^{m-1} g(\alpha_i) \ .$$

Mivel $f = f^*(x-\alpha_m)$, ezért f és f^* együtthatói között az

$$f_m = f^*_{m-1}, f_{m-1} = f^*_{m-2} - f^*_{m-1}\alpha_m, \ldots, f_1 = f^*_0 - f^*_1\alpha_m, f_0 = -f^*_0\alpha_m$$

összefüggések állnak fenn. Így

$$D = \begin{vmatrix} f^*_{m-1} & f^*_{m-2} - f^*_{m-1}\alpha_m & \cdots & \cdots & -f^*_0\alpha_m & & \\ & \ddots & & & & \ddots & \\ & & f^*_{m-1} & \cdots & \cdots & \cdots & -f^*_0\alpha_m \\ g_n & \cdots & \cdots & g_0 & & & \\ & \ddots & & & \ddots & & \\ & & \ddots & & & \ddots & \\ & & & g_n & \cdots & \cdots & g_0 \end{vmatrix} .$$

A determinánst a következőképpen alakítjuk át: az első oszlop α_m-szeresét hozzáadjuk a második oszlophoz, az új második oszlop α_m-szeresét a harmadik oszlophoz stb., végig valamennyi oszlopon. Ezáltal az első n sorból eltűnnek az α_m-ek, vagyis az átalakított D első n sora megegyezik a fenti D^* első n sorával. Az utolsó m sorban az elsőből vonjuk ki a második α_m-szeresét, majd hasonlóan mindegyikből a rákövetkező α_m-szeresét. Végül D-ből az alábbi determináns lesz:

$$D = \begin{vmatrix} f_{m-1}^* & \cdots & \cdots & \cdots & f_0^* & & & \\ & \ddots & & & & \ddots & & \\ & & f_{m-1}^* & \cdots & \cdots & \cdots & f_0^* & \\ g_n & \cdots & \cdots & g_0 & & & & \\ & \ddots & & & \ddots & & & \\ & & \ddots & & & \ddots & & \\ & & & g_n & \cdots & \cdots & g_0 & \\ & & & & g_n & g_n\alpha_m + g_{n-1} & \cdots & g(\alpha_m) \end{vmatrix} .$$

Az utolsó oszlop szerint kifejtve, a $D = D^* g(\alpha_m)$ egyenlőséghez jutunk, amiből az indukciós feltevés alapján $D = f_m^n \prod_{i=1}^m g(\alpha_i)$ következik. ■

Azt kaptuk tehát, hogy $D = \mathrm{res}(f, g)$, vagyis az f és g polinomoknak akkor és csak akkor van közös gyökük K-ban, ha a D determináns eltűnik.

Algoritmikus szempontból magasabb fokú polinomok esetén a rezultáns Sylvester-féle alakjának kiszámolása egy nagy determináns kiszámítását jelenti. Az alábbi tétel szerint a pszeudo-maradékos osztás egyszerűsítheti a számításokat.

2.6. tétel. *A (2.1)-beli f és (2.2)-beli g polinomokra $m \geq n > 0$ esetén*

$$\begin{cases} res(f,g) = 0, & ha\ prem(f,g) = 0\ , \\ g_n^{(m-n)(n-1)+d} res(f,g) = (-1)^{mn} res(g,r), & ha\ r = prem(f,g) \neq 0 és\ d = \deg(r)\ . \end{cases}$$

Bizonyítás. A (2.9) determináns első sorát szorozzuk meg g_n^{m-n+1}-gyel. Legyenek $q = q_{m-n}x^{m-n} + \cdots + q_0 \in R[x]$ és $r = r_d x^d + \cdots + r_0 \in R[x]$ azok az egyértelműen meghatározott polinomok, melyekre

$$\begin{aligned} g_n^{m-n+1}(f_m x^m + \cdots + f_0) \quad = \quad & (q_{m-n}x^{m-n} + \cdots + q_0)(g_n x^n + \cdots + g_0) \\ & + r_d x^d + \cdots + r_0\ , \end{aligned}$$

ahol $r = \mathrm{prem}(f, g)$. Ekkor a rezultáns $(n+1)$-edik sorát q_{m-n}-nel, az $(n+2)$-edik sorát q_{m-n-1}-gyel stb. szorozva, majd az első sorból kivonva a

$$g_n^{m-n+1}\mathrm{res}(f,g) = \begin{vmatrix}
0 & \cdots & 0 & r_d & \cdots & \cdots & r_0 & & & \\
 & f_m & \cdots & \cdots & \cdots & \cdots & \cdots & f_0 & & \\
 & & \ddots & & & & & & \ddots & \\
 & & & f_m & \cdots & \cdots & \cdots & \cdots & \cdots & f_0 \\
g_n & \cdots & \cdots & \cdots & g_0 & & & & & \\
 & \ddots & & & & \ddots & & & & \\
 & & \ddots & & & & \ddots & & & \\
 & & & \ddots & & & & \ddots & & \\
 & & & & \ddots & & & & \ddots & \\
 & & & & & g_n & \cdots & \cdots & \cdots & g_0
\end{vmatrix}$$

determinánst kapjuk. Itt r_d az első sor $(m-d+1)$-edik oszlopában van, r_0 pedig az első sor $(m+1)$-edik oszlopában.

Hasonló módon folytatva szorozzuk meg a második sort g_n^{m-n+1}-gyel, majd szorozzuk meg az $(n+2)$-edik, $(n+3)$-adik, … sort q_{m-n}-nel, q_{m-n-1}-gyel stb., és vonjuk ki őket a második sorból. Ugyanígy a harmadik, …, n-edik sorra. Az eredmény:

$$g_n^{n(m-n+1)}\mathrm{res}(f,g) = \begin{vmatrix}
 & & & r_d & \cdots & \cdots & r_0 & & & \\
 & & & & \ddots & & & \ddots & & \\
 & & & & & \ddots & & & \ddots & \\
 & & & & & & r_d & \cdots & \cdots & r_0 \\
g_n & \cdots & \cdots & \cdots & g_0 & & & & & \\
 & \ddots & & & & \ddots & & & & \\
 & & \ddots & & & & \ddots & & & \\
 & & & \ddots & & & & \ddots & & \\
 & & & & \ddots & & & & \ddots & \\
 & & & & & g_n & \cdots & \cdots & \cdots & g_0
\end{vmatrix}.$$

Sorcserék után azt kapjuk, hogy

$$g_n^{n(m-n+1)}\operatorname{res}(f,g) = (-1)^{mn}\begin{vmatrix} g_n & \cdots & \cdots & \cdots & g_0 & & & & & & \\ & \ddots & & & & \ddots & & & & & \\ & & \ddots & & & & \ddots & & & & \\ & & & g_n & \cdots & \cdots & \cdots & g_0 & & & \\ & & & & \ddots & & & & \ddots & & \\ & & & & & g_n & \cdots & \cdots & \cdots & g_0 & \\ & & & r_d & \cdots & \cdots & r_0 & & & & \\ & & & & \ddots & & & \ddots & & & \\ & & & & & \ddots & & & \ddots & & \\ & & & & & & r_d & \cdots & \cdots & r_0 & \end{vmatrix} .$$

Vegyük észre, hogy

$$\begin{vmatrix} g_n & \cdots & \cdots & \cdots & g_0 & & & \\ & \ddots & & & & \ddots & & \\ & & g_n & \cdots & \cdots & \cdots & g_0 & \\ r_d & \cdots & \cdots & r_0 & & & & \\ & \ddots & & & \ddots & & & \\ & & \ddots & & & \ddots & & \\ & & & r_d & \cdots & \cdots & r_0 & \end{vmatrix} = \operatorname{res}(g,r) ,$$

ezért

$$g_n^{n(m-n+1)}\operatorname{res}(f,g) = (-1)^{mn} g_n^{m-d}\operatorname{res}(g,r) ,$$

amiből

$$g_n^{(m-n)(n-1)+d}\operatorname{res}(f,g) = (-1)^{mn}\operatorname{res}(g,r) \tag{2.10}$$

következik. ■

A (2.10) egyenlet egy nagyon fontos kapcsolatot ír le. Ahelyett, hogy az esetleg óriási méretű $\operatorname{res}(f,g)$ determinánst számítanánk ki, pszeudo-maradékos osztások sorozatát végezzük el, majd minden lépésnél (2.10)-et alkalmazzuk. Csak akkor számoljuk ki a rezultánst, ha már több pszeudo-maradékos osztás nem végezhető el. A tétel fontos következménye az alábbi

2.7. következmény. *Léteznek olyan $u, v \in R[x]$ polinomok, melyre $\operatorname{res}(f,g) = fu + gv$, ahol* $\deg u < \deg g$, $\deg v < \deg f$.

Bizonyítás. A rezultáns determináns alakjában az i-edik oszlopot szorozzuk meg x^{m+n-i}-vel és adjuk az utolsó oszlophoz minden $i = 1, \ldots, (m+n-1)$-re. Az eredmény az alábbi lesz:

$$\mathrm{res}(f,g) = \begin{vmatrix} f_m & \cdots & \cdots & f_0 & \cdots & x^{n-1}f \\ & \ddots & & & \ddots & \vdots \\ & & f_m & \cdots & \cdots & f \\ g_n & \cdots & \cdots & g_0 & \cdots & x^{m-1}g \\ & \ddots & & & \ddots & \vdots \\ & & g_n & \cdots & \cdots & g \end{vmatrix} .$$

A determinánst az utolsó oszlopa szerint kifejtve, majd f-et és g-t kiemelve kapjuk az állításban szereplő egyenlőséget a fokokra vonatkozó megszorításokkal. ■

A rezultánsmódszer legfontosabb előnye a korábban látott módszerekhez képest, hogy a bemeneti polinomok szimbolikus együtthatókat is tartalmazhatnak.

2.7. példa. Legyen

$$\begin{aligned} f(x) &= 2x^3 - \xi x^2 + x + 3 \in \mathbb{Q}[x] , \\ g(x) &= x^2 - 5x + 6 \in \mathbb{Q}[x] . \end{aligned}$$

Ekkor f és g $\mathbb{Q}$-beli közös gyökeinek létezését az euklideszi algoritmus variánsai segítségével nem tudjuk eldönteni, míg a rezultánsmódszerrel igen: pontosan akkor van közös gyök, ha

$$\mathrm{res}(f,g) = \begin{vmatrix} 2 & -\xi & 1 & 3 & \\ & 2 & -\xi & 1 & 3 \\ 1 & -5 & 6 & & \\ & 1 & -5 & 6 & \\ & & 1 & -5 & 6 \end{vmatrix} = 36\xi^2 - 429\xi + 1260 = 3(4\xi - 21)(3\xi - 20) = 0 ,$$

vagyis ha $\xi = 20/3$, vagy $\xi = 21/4$.

A rezultáns jelentősége nemcsak abban áll, hogy segítségével két polinom közös gyökének létezése eldönthető, hanem abban is, hogy használatával algebrai egyenletrendszerek rekurzív módon visszavezethetők egyismeretlenes algebrai egyenletek megoldására.

2.8. példa. Legyen

$$f(x,y) = x^2 + xy + 2x + y - 1 \in \mathbb{Z}[x,y] , \tag{2.11}$$

$$g(x,y) = x^2 + 3x - y^2 + 2y - 1 \in \mathbb{Z}[x,y] . \tag{2.12}$$

Értelmezzük az f és g polinomokat úgy, mint $(\mathbb{Z}[x])[y]$-beli elemeket. Pontosan akkor létezik közös gyökük, ha

$$\mathrm{res}_y(f,g) = \begin{vmatrix} x+1 & x^2+2x-1 & 0 \\ 0 & x+1 & x^2+2x-1 \\ -1 & 2 & x^2+3x-1 \end{vmatrix} = -x^3 - 2x^2 + 3x = 0 .$$

$\mathbb{Z}$-beli közös gyökök tehát az $x \in \{-3, 0, 1\}$ esetben létezhetnek. Minden x-hez (immáron $\mathbb{Z}[y]$-ban) visszahelyettesítéssel megoldjuk a (2.11), (2.12) egyenleteket, amikor is azt kapjuk, hogy az egyenletek egész megoldásai a $(-3, 1), (0, 1), (1, -1)$ számpárok.

Megjegyezzük, hogy a rezultánsmódszer többváltozós polinomegyenlet-rendszerek megoldásainak megkeresésére is alkalmas, ámbár nem igazán hatékony. Az egyik probléma az, hogy a determináns kiszámítása során számítási tárrobbanás lép fel. Megfigyelhetjük, hogy az egyhatározatlanú m és n-edfokú polinomok rezultánsa determináns alakjának kiszámítása a szokásos Gauss-eliminációval $O((m+n)^3)$-ös műveletigényű, míg az euklideszi algoritmus változatai kvadratikusak. A másik probléma, hogy a számítási bonyolultság erősen függ a határozatlanok sorrendjétől. Sokkal hatékonyabb, ha a polinomegyenlet-rendszer *összes változóját egyszerre eliminálják*. Ez az út vezet el a többváltozós rezultánsok elméletéhez.

2.2.4. Moduláris legnagyobb közös osztó

A polinomok közös gyökeinek létezésére és meghatározására szolgáló eddigi módszerek mindegyikére jellemző volt a számítási tárrobbanás. Ösztönösen vetődik fel a kérdés: van-e lehetőség moduláris módszerek alkalmazására? Az alábbiakban az $a(x), b(x) \in \mathbb{Z}[x]$ esetet vizsgáljuk ($a, b \neq 0$). Tekintsük a (2.4), (2.5) $\in \mathbb{Z}[x]$ polinomokat és legyen $p = 13$ prím. Ekkor $\mathbb{Z}_p[x]$-ben a KLASSZIKUS-EUKLIDESZ algoritmus maradéksorozata az alábbi lesz:

$$\begin{aligned} r_0 &= 11x^5 + 5x^4 + 6x^3 + 6x^2 + 5 \,, \\ r_1 &= x^4 + x^3 + 2x^2 + 8x + 8 \,, \\ r_2 &= 3x^3 + 8x^2 + 12x + 1 \,, \\ r_3 &= x^2 + 10x + 10 \,, \\ r_4 &= 7x \,, \\ r_5 &= 10 \,. \end{aligned}$$

Azt kapjuk tehát, hogy $\mathbb{Z}_p[x]$-ben az a és b polinomok relatív prímek. Az alábbi tétel a $\mathbb{Z}[x]$-ben és $\mathbb{Z}_p[x]$-ben vett legnagyobb közös osztók közötti kapcsolatot írja le.

2.8. tétel. *Legyen $a, b \in \mathbb{Z}[x]$, $a, b \neq 0$. Legyen p olyan prím, amelyre $p \nmid lc(a)$ és $p \nmid lc(b)$. Legyen továbbá $c = lnko(a, b) \in \mathbb{Z}[x]$, $a_p = a\,remp$, $b_p = b\,remp$ és $c_p = c\,remp$. Ekkor*

(1) $\deg(lnko(a_p, b_p)) \geq \deg(lnko(a, b))$,

(2) ha $p \nmid res(a/c, b/c)$, akkor $lnko(a_p, b_p) = c_p$.

Bizonyítás. (1) bizonyítása: mivel $c_p \mid a_p$ és $c_p \mid b_p$ ezért $c_p \mid \text{lnko}(a_p, b_p)$. Így

$$\deg(\text{lnko}(a_p, b_p)) \geq \deg(\text{lnko}(a, b) \bmod p).$$

Azonban a feltételek miatt $p \nmid \text{lc}(\text{lnko}(a, b))$, ezért

$$\deg(\text{lnko}(a, b) \bmod p) = \deg(\text{lnko}(a, b)) \,.$$

(2) bizonyítása: mivel $\text{lnko}(a/c, b/c) = 1$, valamint c_p nemtriviális, ezért

$$\text{lnko}(a_p, b_p) = c_p \cdot \text{lnko}(a_p/c_p, b_p/c_p) \,. \tag{2.13}$$

Ha $\text{lnko}(a_p, b_p) \neq c_p$, akkor (2.13) jobb oldala nemtriviális, így $\text{res}(a_p/c_p, b_p/c_p) = 0$. De a rezultáns az együtthatók megfelelő szorzatainak összege, így $p \mid \text{res}(a/c, b/c)$, ami ellentmondás. ■

2.9. következmény. *Véges sok olyan p prím van, amire $p \nmid lc(a)$, $p \nmid lc(b)$ és* $\deg(lnko(a_p, b_p)) > \deg(lnko(a, b))$. ■

Amikor a 2.8. tétel (1) állításában egyenlőség teljesül, akkor azt mondjuk, hogy p egy „szerencsés prím”. Máris körvonalazhatunk egy legnagyobb közös osztót kiszámoló moduláris algoritmust.

MODULÁRIS-LNKO-NAGYPRÍM(a, b)

```
1  M ← a Landau–Mignotte-konstans (a 2.4. következmény szerint)
2  H ← {}
3  while IGAZ
4        do p ← olyan prím, melyre p ≥ 2M, p ∉ H, p ∤ lc(a) és p ∤ lc(b)
5           c_p ← lnko(a_p, b_p)
6           if c_p | a és c_p | b
7              then return c_p
8              else H ← H ∪ {p}
```

Az algoritmus első sora a Landau–Mignotte-korlát kiszámítását kéri. A negyedik sor szerint olyan „elég nagy” prímet kell választani, amely nem osztja sem a, sem b főegyüttható́ját. Az ötödik sor (például a $\mathbb{Z}_p[x]$-beli KLASSZIKUS-EUKLIDESZ algoritmussal) kiszámolja az a és b polinomok legnagyobb közös osztóját modulo p. A kapott polinom együtthatóit szimmetrikus ábrázolással tároljuk. A hatodik sor $c_p \mid a$ és $c_p \mid b$ teljesülését vizsgálja, melynek igazsága esetén c_p a keresett legnagyobb közös osztó. Ha ez nem teljesül, akkor p egy „szerencsétlen prím”, így új prímet választunk. Mivel a 2.8. tétel szerint csak véges sok „szerencsétlen prím” van, ezért az algoritmus előbb-utóbb véget ér. Amennyiben a prímeket megfelelő stratégia szerint választjuk, a H halmaz alkalmazása szükségtelen.

A MODULÁRIS-LNKO-NAGYPRÍM hátránya, hogy a bemeneti polinomok fokszámának növekedtével a Landau–Mignotte-konstans exponenciálisan nő, így ez esetben nagy prímekkel kell számolni. Felmerül a kérdés, hogyan módosítsuk az algoritmust, hogy „sok kis prímmel” számolhassunk? Mivel $\mathbb{Z}_p[x]$-ben a legnagyobb közös osztó az együtthatók konstanssal vett szorzata erejéig egyértelmű, ezért az új algoritmusban ügyelni kell a részpolinomok együtthatóira. Mielőtt tehát a kínai maradéktételt alkalmazzuk a különböző prímekkel vett moduláris legnagyobb közös osztók együtthatóira, minden lépésnél normalizálni kell az $\mathrm{lnko}(a_p, b_p)$ főegyüttható́ját. Amennyiben a_m és b_n az a és b polinomok főegyütthatói, akkor $\mathrm{lnko}(a, b)$ főegyüttható́ja osztja $\mathrm{lnko}(a_m, b_n)$-t. Primitív a és b polinomok esetén $\mathrm{lnko}(a_p, b_p)$ főegyüttható́ját ezért $\mathrm{lnko}(a_m, b_n)$ mod p-re normalizáljuk, majd a legvégén vesszük az eredmény polinom primitív részét. Ahogy a MODULÁRIS-LNKO-NAGYPRÍM algoritmusnál, a moduláris számítások eredményét most is szimmetrikus ábrázolással tároljuk. Ezek a meggondolások vezetnek az alábbi, kis prímeket használó moduláris legnagyobb közös osztó algoritmushoz.

MODULÁRIS-LNKO-KISPRÍMEK(a, b)

```
1  d ← lnko(lc(a), lc(b))
2  p ← olyan prím, melyre p ∤ d
3  H ← {p}
```

```
 4  P ← p
 5  c_p ← lnko(a_p, b_p)
 6  g_p ← (d mod p) · c_p
 7  (n, i, j) ← (3, 1, 1)
 8  while IGAZ
 9        do if j = 1
10              then if deg g_p = 0
11                        then return 1
12              (g, j, P) ← (g_p, 0, p)
13           if n ≤ i
14              then g ← pp(g)
15                   if g | a és g | b
16                      then return g
17           p ← olyan prím, melyre p ∤ d és p ∉ H
18           H ← H ∪ {p}
19           c_p ← lnko(a_p, b_p)
20           g_p ← (d mod p) · c_p
21           if deg g_p < deg g
22              then (i, j) ← (1, 1)
23           if j = 0
24              then if deg g_p = deg g
25                        then g_1 = EH-ÉPÍT(g, g_p, P, p)
26                             if g_1 = g
27                                then i ← i + 1
28                                else i ← 1
29                             P ← P · p
30                             g ← g_1
```

EH-ÉPÍT(a, b, m_1, m_2)

```
1  p ← 0
2  c ← 1/m_1 mod m_2
3  for i ← deg a downto 0
4      do r ← a_i mod m_1
5         s ← (b_i − r) mod m_2
6         p ← p + (r + s · m_1)x^i
7  return p
```

Észrevehetjük, hogy a MODULÁRIS-LNKO-KISPRÍMEK algoritmusban nincs szükség annyi kis prímre, mint amennyit a Landau–Mignotte-korlát meghatároz. Amennyiben a g polinom értéke néhány iteráción keresztül nem változik, a 13–16. sorokban teszteljük, hogy g valóban a legnagyobb közös osztó-e. Ezen iterációk számát tárolja a hatodik sor n változója. Megjegyezzük, hogy n értékét a bemeneti polinomoktól függően változtatni is lehetne. Az algoritmusban használt prímeket célszerű egy olyan előre eltárolt listából választani, amely az architektúrának megfelelő gépi szóban elférő prémeket tartalmazza; ilyenkor a H halmaz

használata szükségtelen. A 2.9. következmény miatt a MODULÁRIS-LNKO-KISPRÍMEK algoritmus befejeződik.

Az EH-ÉPÍT algoritmus a bemeneti a, b polinomok azonos fokú tagjainak együtthatóira felírt, modulo m_1 és m_2 vett két lineáris kongruenciából álló egyenletrendszer megoldását számolja ki a kínai maradéktételnek megfelelően. Nagyon fontos, hogy az eredmény polinom együtthatóit szimmetrikus ábrázolással tároljuk.

2.9. példa. Először vizsgáljuk meg MODULÁRIS-LNKO-KISPRÍMEK algoritmus működését a korábban is vizsgált (2.4), (2.5) polinomokra. A könnyebb érthetőség kedvéért kis prímekkel fogunk számolni. Emlékeztetőül,

$$\begin{aligned} a(x) &= 63x^5 + 57x^4 - 59x^3 + 45x^2 - 8 \in \mathbb{Z}[x] , \\ b(x) &= -77x^4 + 66x^3 + 54x^2 - 5x + 99 \in \mathbb{Z}[x] . \end{aligned}$$

Az algoritmus első hat sorának végrehajtása után a $p = 5$ választással $d = 7, c_p = x^2 + 3x + 2$ és $g_p = 2x^2 + x - 1$ lesznek. Mivel a 7. sor miatt a j változó értéke 1, ezért a 10–12. sorok végrehajtódnak. A g_p polinom nem nulla, ezért $g = 2x^2 + x - 1$, $j = 0$, valamint $P = 5$ lesznek a végrehajtás utáni változóértékek. A 13. sorban a feltétel értéke nem teljesül, így újabb prímet választunk. A $p = 7$ rossz választás, a $p = 11$ viszont megengedett. A 19–20. sorok szerint ekkor $c_p = 1$, $g_p = -4$. Mivel $\deg g_p < \deg g$, ezért $j = 1$ lesz és a 25–30. sorok nem hajtódnak végre. A g_p polinom konstans, így a 11. sorban a visszatérési érték 1 lesz, jelezve, hogy az a és b polinomok relatív prímek.

2.10. példa. Második példánkban tekintsük a korábbi

$$\begin{aligned} a(x) &= 12x^4 - 68x^3 + 52x^2 - 92x + 56 \in \mathbb{Z}[x] , \\ b(x) &= -12x^3 + 80x^2 - 84x + 24 \in \mathbb{Z}[x] , \end{aligned}$$

polinomokat. Legyen ismét $p = 5$. Az algoritmus első hat sora után $d = 12, c_p = x + 1, g_p = 2x + 2$. A 10–12. sorok végrehajtása után $P = 5, g = 2x + 2$ lesznek a változók értékei. A következő prím legyen $p = 7$. Így az új értékek $c_p = x + 4, g_p = -2x - 1$. Mivel $\deg g_p = \deg g$, ezért a 25–30. sorok után $P = 35$ és g új értéke $12x - 8$ lesz. Az i változó értéke továbbra is 1. A következő prímet válasszuk 11-nek. Ekkor $c_p = g_p = x + 3$. A g_p és g fokai megegyeznek, így g együtthatóit módosítjuk. Ekkor $g_1 = 12x - 8$ és mivel $g = g_1$, ezért $i = 2$, valamint $P = 385$ lesznek. Az új prím legyen 13. Ekkor $c_p = x + 8, g_p = -x + 5$. A g_p és g fokai továbbra is megegyeznek, ezért a 25–30 sorok végrehajtódnak, a változók értékei $g = 12x - 8, P = 4654, i = 3$ lesznek.

A 17–18. sorok végrehajtása után kiderül, hogy $g \mid a$ és $g \mid b$ ezért a legnagyobb közös osztó $g = 12x - 8$.

Az alábbi tételt bizonyítás nélkül közöljük.

2.10. tétel. *A* MODULÁRIS-LNKO-KISPRÍMEK *algoritmus megfelelően működik. Az algoritmus számítási bonyolultsága* $O(m^3(\log m + \lambda(K))^2)$ *gépi szóban mért művelet, ahol* $m = \min\{\deg a, \deg b\}$, *K pedig az a és b polinomok Landau–Mignotte-korlátja.*

Gyakorlatok

2.2-1. Legyen R egy kommutatív egységelemes gyűrű, $a = \sum_{i=0}^{m} a_i x^i \in R[x]$, $b = \sum_{i=0}^{n} b_i x^i \in R[x]$, továbbá b_n egység, $m \geq n \geq 0$. Az alábbi algoritmus az a és b polinomokkal végzett maradékos osztás eredményeként előállítja azokat a $q, r \in R[x]$ polinomokat, melyekre $a = qb + r$ és $\deg r < n$ vagy $r = 0$.

EGYHATÁROZATLANÚ-POLINOMOK-MARADÉKOS-OSZTÁSA(a, b)

```
1  r ← a
2  for i ← m − n downto 0
3      do if deg r = n + i
4           then q_i ← lc(r)/b_n
5                r ← r − q_i x^i b
6           else q_i ← 0
7  q ← Σ_{i=0}^{m−n} q_i x^i és r
8  return q
```

Bizonyítsuk be, hogy az algoritmus legfeljebb

$$(2 \deg b + 1)(\deg q + 1) = O(m^2)$$

R-beli műveletet igényel.

2.2-2. Mi a különbség $\mathbb{Z}$-ben a BŐVÍTETT-EUKLIDESZ és a BŐVÍTETT-EUKLIDESZ-NORMALIZÁLT algoritmusok között?

2.2-3. Bizonyítsuk be, hogy $\operatorname{res}(f \cdot g, h) = \operatorname{res}(f, h) \cdot \operatorname{res}(g, h)$.

2.2-4. Az $f(x) \in R[x]$ polinom ($\deg f = m$, $\operatorname{lc}(f) = f_m$) ***diszkriminánsának*** a

$$\operatorname{discr} f = \frac{(-1)^{\frac{m(m-1)}{2}}}{f_m} \operatorname{res}(f, f') \in R$$

elemet nevezzük, ahol f' az f x-beli deriváltját jelenti. Az f polinomnak nyilván akkor és csak akkor van többszörös gyöke, ha diszkriminánsa nulla. Számítsuk ki ($\operatorname{discr} f$)-et általános másod- és harmadfokú polinomok esetében.

2.3. Gröbner-bázis

Legyen F test, $R = F[x_1, x_2, \ldots, x_n]$ az F feletti n-határozatlanú polinomok gyűrűje, továbbá legyen $f_1, f_2, \ldots, f_s \in R$. Állapítsuk meg, hogy mi annak a szükséges és elégséges feltétele, hogy az $f_1, f_2, \ldots, f_s$ polinomoknak legyen közös gyöke R-ben. Látható, hogy a probléma az előző alfejezet $s = 2$ esetének bizonyos értelemben vett általánosítása.

Jelölje

$$I = \langle f_1, \ldots, f_s \rangle = \left\{ \sum_{1 \le i \le s} q_i f_i : q_i \in R \right\}$$

az $f_1, \ldots, f_s$ polinomok által generált ideált. Ekkor az $f_1, \ldots, f_s$ polinomok az I ***ideál bázisát*** alkotják. Az I ideál ***varietásán*** a

$$V(I) = \left\{ u \in F^n : f(u) = 0 \text{ minden } f \in I \text{ polinomra} \right\}$$

halmazt értjük. A varietás ismerete természetesen az $f_1, \ldots, f_s$ polinomok közös megoldásainak ismeretét is jelenti. A varietásról, illetve az I ideálról feltehető legfontosabb kérdések:

- $V(I) \neq \emptyset$?

- $V(I)$ „mekkora” ?
- Adott $f \in R$ esetén $f \in I$?
- $I = R$?

Az I ideál Gröbner-bázisa egy olyan bázis, ahol ezeket a kérdéseket könnyű megválaszolni. Mindenekelőtt vizsgáljuk meg az $n = 1$ esetet. Mivel $F[x]$ euklideszi gyűrű, ezért

$$\langle f_1, \ldots, f_s \rangle = \langle \text{lnko}(f_1, \ldots, f_s) \rangle \,. \tag{2.14}$$

Feltehetjük, hogy $s = 2$. Legyen $f, g \in F[x]$, és osszuk el maradékosan f-et g-vel. Ekkor egyértelműen léteznek olyan $q, r \in F[x]$ polinomok, melyekre $f = qg+r$, ahol $\deg r < \deg g$. Vagyis

$$f \in \langle g \rangle \Leftrightarrow r = 0 \,,$$

továbbá $V(g) = \{u_1, \ldots, u_d\}$, amennyiben $x - u_1, \ldots, x - u_d$ a $g \in F[x]$ polinom összes különböző lineáris faktora. Sajnos, a (2.14) egyenlőség két vagy több határozatlan esetén már nem teljesül. Sőt, akármilyen test feletti többváltozós polinomgyűrű nem euklideszi gyűrű, így a maradékos osztás lehetőségét is újra kell gondolni. Ebben az irányban haladunk tovább.

2.3.1. Monomiális rendezés

A ‘$\preceq$‘ $\subseteq \mathbb{N}^n$ teljes rendezési relációt ***megengedettnek*** nevezzük, ha

(i) $(0, \ldots, 0) \preceq v$ minden $v \in \mathbb{N}^n$-re,
(ii) minden $v_1, v_2, v \in \mathbb{N}^n$ esetén $v_1 \preceq v_2 \Rightarrow v_1 + v \preceq v_2 + v$.

Nem nehéz bizonyítani, hogy $\mathbb{N}^n$ minden megengedett rendezése egyben jólrendezés is (vagyis bármely nemüres részhalmazának van legkisebb eleme). A korábbi jelöléseket figyelembe véve tekintsük a

$$T = \{x_1^{i_1} \cdots x_n^{i_n}\}$$

halmazt, melynek elemeit ***monomoknak*** nevezzük. Vegyük észre, hogy T zárt az $F[x_1, \ldots, x_n]$-beli szorzásra, valamint a művelettel kommutatív monoidot alkot. Az $\mathbb{N}^n \to T$, $(i_1, \ldots, i_n) \mapsto x_1^{i_1} \cdots x_n^{i_n}$ leképezés izomorfizmus, ezért egy T-beli $\preceq$ megengedett teljes rendezésre

(i) $1 \preceq t$ minden $t \in T$-re,
(ii) minden $t_1, t_2, t \in T$ esetén $t_1 \prec t_2 \Rightarrow t_1 t \prec t_2 t$.

A T-beli megengedett rendezéseket ***monomiális rendezéseknek*** nevezzük. Tekintsünk néhány példát. Legyen

$$\alpha = x_1^{i_1} \cdots x_n^{i_n},\ \beta = x_1^{j_1} \cdots x_n^{j_n} \in T \,.$$

Ekkor az alábbi rendezéseket szokás definiálni.

- ***Lexikografikus rendezés.***
  $\alpha \prec_{lex} \beta \Leftrightarrow$ létezik olyan $l \in \{1, \ldots, n\}$, hogy $i_l < j_l$ és $i_{l+1} = j_{l+1}, \ldots, i_n = j_n$.
- ***Tiszta lexikografikus rendezés.***
  $\alpha \prec_{plex} \beta \Leftrightarrow$ létezik olyan $l \in \{1, \ldots, n\}$, hogy $i_l < j_l$ és $i_1 = j_1, \ldots, i_{l-1} = j_{l-1}$.

- ***Összfokszám szerint, majd lexikografikus rendezés.***
  $\alpha \prec_{grlex} \beta \Leftrightarrow i_1 + \cdots + i_n < j_1 + \cdots + j_n$ vagy $(i_1 + \cdots + i_n = j_1 + \cdots + j_n$ és $\alpha \prec_{lex} \beta)$.
- ***Összfokszám szerint, majd fordított lexikografikus rendezés.***
  $\alpha \prec_{tdeg} \beta \Leftrightarrow i_1 + \cdots + i_n < j_1 + \cdots + j_n$ vagy $(i_1 + \cdots + i_n = j_1 + \cdots + j_n$ és létezik olyan $l \in \{1, \ldots, n\}$, melyre $i_l > j_l$ és $i_{l+1} = j_{l+1}, \ldots, i_n = j_n)$.

A $\prec_{tdeg}$ monomiális rendezést némely szerzők $\prec_{grrevlex}$-nek jelölik („graded reverse lexicographic order"), hiszen a rendezés a monomok fokszámösszehasonlítás utáni lexikografikus rendezésének felel meg.

2.11. példa. Legyen $<=\prec_{plex}$. Ekkor $z \prec y \prec x$ esetén

$$\begin{aligned} 1 &\prec z \prec z^2 \prec \cdots \prec y \prec yz \prec yz^2 \prec \cdots \\ &\prec y^2 \prec y^2z \prec y^2z^2 \prec \cdots x \prec xz \prec xz^2 \prec \cdots \\ &\prec xy \prec xy^2 \prec \cdots \prec x^2 \prec \cdots . \end{aligned}$$

Legyen $<=\prec_{tdeg}$. Ekkor $z \prec y \prec x$ esetén

$$\begin{aligned} 1 &\prec z \prec y \prec x \\ &\prec z^2 \prec yz \prec xz \prec y^2 \prec xy \prec x^2 \\ &\prec z^3 \prec yz^2 \prec xz^2 \prec y^2z \prec xyz \\ &\prec x^2z \prec y^3 \prec xy^2 \prec x^2y \prec x^3 \prec \cdots . \end{aligned}$$

A továbbiakban mindig feltesszük, hogy valamilyen rögzített monomiális rendezés mellett dolgozunk. Legyen $f = \sum_{\alpha \in \mathbb{N}^n} c_\alpha x^\alpha \in R$ egy nem nulla polinom, $c_\alpha \in F$ és legyen adott egy $\prec$ monomiális rendezés. Ekkor $c_\alpha x^\alpha$ $(c_\alpha \neq 0)$ az f polinom ***tagjai,*** $\operatorname{discr}(f) = \max\{\alpha \in \mathbb{N}^n : c_\alpha \neq 0\}$ a polinom ***multifoka*** (a maximum a monomiális rendezés mellett értendő), $\operatorname{lc}(f) = c_{\operatorname{discr}(f)} \in F \setminus \{0\}$ a polinom ***főegyütthatója,*** $\operatorname{lm}(f) = x^{\operatorname{discr}(f)} \in R$ a polinom főmonomja, $\operatorname{lc}(f) = \operatorname{lc}(f) \cdot \operatorname{lm}(f) \in R$ a polinom ***főtagja.*** Legyenek továbbá $\operatorname{lc}(0) = \operatorname{lc}(0) = \operatorname{lm}(0) = 0$ és $\operatorname{discr}(0) = -\infty$.

2.12. példa. Tekintsük az $f(x, y, z) = 2xyz^2 - 3x^3 + 4y^4 - 5xy^2z \in \mathbb{Q}[x, y, z]$ polinomot. Amennyiben $<=\prec_{plex}$ és $z \prec y \prec x$, akkor

$$\operatorname{discr}(f) = (3, 0, 0),\ \operatorname{lc}(f) = -3x^3,\ \operatorname{lm}(f) = x^3,\ \operatorname{lc}(f) = -3\ ,$$

ha pedig $<=\prec_{tdeg}$ és $z \prec y \prec x$, akkor

$$\operatorname{discr}(f) = (0, 4, 0),\ \operatorname{lc}(f) = 4y^4,\ \operatorname{lm}(f) = y^4,\ \operatorname{lc}(f) = 4\ .$$

2.3.2. Többváltozós polinomok maradékos osztása

Ebben a pontban adott $f, f_1, \ldots, f_s \in R$ többváltozós polinomok és adott $\prec$ monomiális rendezés esetén olyan $q_1, \ldots, q_s \in R$ és $r \in R$ polinomokat keresünk, melyekre $f = q_1f_1 + \cdots + q_sf_s + r$ és r egyetlen monomja sem osztható $\operatorname{lc}(f_1), \ldots, \operatorname{lc}(f_s)$ egyikével sem.

TÖBBHATÁROZATLANÚ-POLINOMOK-MARADÉKOS-OSZTÁSA$(f, f_1, \ldots, f_s)$

```
1  r ← 0
2  p ← f
3  for i ← 1 to s
4      do q_i ← 0
5  while p ≠ 0
6        do if valamilyen i ∈ {1,...,s}-re lc(f_i) | lc(p)
7              then az egyik ilyen i-re q_i ← q_i + lc(p)/lc(f_i)
8                   p ← p − lc(p)/lc(f_i) f_i
9              else r ← r + lc(p)
10                  p ← p − lc(p)
11 return q_1,...,q_s és r
```

Az algoritmus helyes működése abból következik, hogy az 5–10. sorok **while** ciklusának minden iterációjában fennállnak az alábbi tulajdonságok:

(i) $\operatorname{discr}(p) \leq \operatorname{discr}(f)$ és $f = p + q_1 f_1 + \cdots + g_s f_s + r$,
(ii) $q_i \neq 0 \Rightarrow \operatorname{discr}(q_i f_i) \leq \operatorname{discr}(f)$ minden $1 \leq i \leq s$ esetén,
(iii) r egyetlen tagja sem osztója egyik $\operatorname{lc}(f_i)$-nek sem.

Algoritmusunknak van egy gyenge pontja: a többváltozós polinomok maradékos osztása nem egyértelmű. A 7. sorban a megfelelő i értékek közül tetszőlegesen választhatunk.

2.13. példa. Legyen $f = x^2y + xy^2 + y^2 \in \mathbb{Q}[x,y]$, $f_1 = xy - 1$, $f_2 = y^2 - 1$, a monomiális rendezés $\prec_{plex}$, $y \prec_{plex} x$, és a 7. sorban mindig a legkisebb indexű megfelelő i értéket válasszuk. Ekkor az algoritmus eredménye $q_1 = x + y$, $q_2 = 1$, $r = x + y + 1$. De ha f_1 és f_2 szerepét felcseréljük, vagyis $f_1 = y^2 - 1$ és $f_2 = xy - 1$, akkor az algoritmus kimenete $q_1 = x + 1$, $q_2 = x$ és $r = 2x + 1$ lesz.

Az iménti példában látott módon (nevezetesen, hogy az pszeudokód 7. sorában mindig a legkisebb megfelelő pozitív i értéket választjuk) az algoritmus determinisztikussá tehető. Ilyenkor a $q_1, \ldots, q_s$ ***hányadosok*** és az r ***maradék*** egyértelmű, amit úgy jelölünk, hogy $r = f \operatorname{rem} (f_1, \ldots, f_s)$.

Az $s = 1$ esetben az algoritmus választ ad az ideál-tartalmazás problémájára: $f \in \langle f_1 \rangle$ akkor és csak akkor, ha az f polinom f_1 polinommal vett maradékos osztásakor a maradék nulla. Sajnos $s \geq 2$ esetén ez nem teljesül: a $\prec_{plex}$ monomiális rendezéssel

$$xy^2 - x \operatorname{rem} (xy + 1, y^2 - 1) = -x - y,$$

a hányadosok pedig $q_1 = y, q_2 = 0$. Másrészről viszont $xy^2 - x = x \cdot (y^2 - 1) + 0$, ami azt mutatja, hogy $xy^2 - x \in \langle xy + 1, y^2 - 1 \rangle$.

2.3.3. Monomiális ideálok és Hilbert-féle bázistétel

További célunk tetszőleges polinomideálhoz olyan bázis keresése, hogy ezzel a bázissal vett maradékos osztáskor a maradék egyértelmű legyen, így választ tudjunk adni az ideál-tartalmazás problémára. Vajon ilyen bázis egyáltalán létezik? És ha igen, véges elemszámú?

Az $I \subseteq R$ ideált ***monomiális ideálnak*** nevezzük, ha létezik olyan $A \subseteq \mathbb{N}^n$, melyre

$$I = \langle x^A \rangle = \langle \{x^\alpha \in T : \alpha \in A\} \rangle ,$$

vagyis az ideált monomok generálják.

2.11. lemma. *Legyen $I = \langle x^A \rangle \subseteq R$ egy monomiális ideál, és $\beta \in \mathbb{N}^n$. Ekkor*

$$x^\beta \in I \Leftrightarrow \exists \alpha \in A \quad x^\alpha \mid x^\beta \ .$$

Bizonyítás. A $\Leftarrow$ irány nyilvánvaló. Megfordítva, legyen $\alpha_1, \ldots, \alpha_s \in A$ és $q_1, \ldots, q_s \in R$, melyekre $x^\beta = \sum_i q_i x^{\alpha_i}$. Ekkor az összegnek legalább egy olyan $q_i x^{\alpha_i}$ tagja létezik, melyben x^β előfordul, így $x^{\alpha_i} \mid x^\beta$. ∎

A lemma legfontosabb következménye, hogy monomiális ideálok pontosan akkor egyeznek meg, ha ugyanazokat a monomokat tartalmazzák.

2.12. lemma (Dickson-lemma). *Minden monomiális ideál végesen generálható, vagyis minden $A \subseteq \mathbb{N}^n$-hez létezik olyan $B \subseteq A$ véges halmaz, melyre $\langle x^A \rangle = \langle x^B \rangle$.*

2.13. lemma. *Legyen I egy ideál $R = F[x_1, \ldots, x_n]$-ben. Ha $G \subseteq I$ egy olyan véges halmaz, melyre $\langle lc(G) \rangle = \langle lc(I) \rangle$, akkor $\langle G \rangle = I$.*

Bizonyítás. Legyen $G = \{g_1, \ldots, g_s\}$. Ha $f \in I$ egy tetszőleges polinom, akkor G-vel vett maradékos osztás szerint $f = q_1 g_1 + \cdots + q_s g_s + r$, ahol $q_1, \ldots, q_s, r \in R$ olyanok, hogy vagy $r = 0$, vagy r egyetlen tagja sem osztható egyetlen g_i főtagjával sem. De ekkor $r = f - q_1 g_1 - \cdots - g_s g_s \in I$, és így $\mathrm{lc}(r) \in \mathrm{lc}(I) \subseteq \langle \mathrm{lc}(g_1), \ldots, \mathrm{lc}(g_s) \rangle$. A (2.11) lemma miatt ezért $r = 0$, vagyis $f \in \langle g_1, \ldots, g_s \rangle = \langle G \rangle$. ∎

Amennyiben a Dickson-lemmát $\langle \mathrm{lc}(I) \rangle$-re alkalmazzuk, továbbá figyelembe vesszük, hogy a zérópolinom a $\langle 0 \rangle$ ideált generálja, az alábbi nevezetes eredményt kapjuk.

2.14. tétel (Hilbert-féle bázistétel). *Minden $I \subseteq R = F[x_1, \ldots, x_n]$ ideál végesen generálható, vagyis létezik olyan $G \subseteq I$ véges halmaz, melyre $\langle G \rangle = I$ és $\langle lc(G) \rangle = \langle lc(I) \rangle$.*

2.15. következmény (ideállánc-feltétel). *Legyen $I_1 \subseteq I_2 \subseteq \cdots$ R-beli ideálok egy növekvő lánca. Ekkor létezik olyan $n \in \mathbb{N}$, melyre $I_n = I_{n+1} = \cdots$.*

Bizonyítás. Legyen $I = \cup_{j \geq 1} I_j$. Ekkor I ideál, ami a Hilbert-féle bázistétel miatt végesen generálható. Legyen $I = \langle g_1, \ldots, g_s \rangle$. Az $n = \min\{j \geq 1 : g_1, \ldots, g_s \in I_j\}$ választással azt kapjuk, hogy $I_n = I_{n+1} = \cdots = I$. ∎

Azokat a gyűrűket, melyekben teljesül az ideál-lánc feltétel, ***Noether-gyűrűknek*** nevezzük. Speciálisan, amennyiben F test, akkor $F[x_1, \ldots, x_n]$ Noether-gyűrű.

Legyen $\prec$ egy monomiális rendezés R-en és $I \subseteq R$ egy ideál. A $G \subseteq I$ véges halmazt az I ideál $\prec$ rendezésre vonatkozó ***Gröbner-bázisának*** nevezzük, ha $\langle \mathrm{lc}(G) \rangle = \langle \mathrm{lc}(I) \rangle$. A Hilbert-féle bázistétel következménye az alábbi

2.16. következmény. *$R = F[x_1, \ldots, x_n]$ minden I ideáljának van Gröbner-bázisa.*

Könnyű megmutatni, hogy egy G Gröbner-bázissal vett maradékos osztáskor a maradék nem függ a báziselemek sorrendjétől. Ilyenkor az $f \mathrm{rem} G = r \in R$ jelölés használatos. Az alábbi tétel szerint a Gröbner-bázis segítségével az ideál-tartalmazás problémája egyszerűen megválaszolható.

2.17. tétel. *Legyen G az $I \subseteq R$ ideál $\prec$ monomiális rendezésre vonatkozó Gröbner-bázisa és legyen $f \in R$. Ekkor $f \in I \Leftrightarrow f \mathrm{rem} G = 0$.*

Bizonyítás. Bebizonyítjuk, hogy egyértelműen létezik olyan $r \in R$, amelyre (1) $f - r \in I$, (2) r egyetlen tagja sem osztható $\mathrm{lc}(G)$ egyetlen monomjával sem. Ilyen r létezése a maradékos osztásból következik. Az egyértelműséghez feltesszük, hogy $f = h_1 + r_1 = h_2 + r_2$ valamilyen $h_1, h_2 \in I$-re és r_1 vagy r_2 egyik tagja sem osztható $\mathrm{lc}(G)$ egyetlen monomjával sem. Ekkor $r_1 - r_2 = h_2 - h_1 \in I$, továbbá a 2.11. lemma miatt $\mathrm{lc}(r_1 - r_2)$ osztható $\mathrm{lc}(g)$-vel valamely $g \in G$-re. Ez azt jelenti, hogy $r_1 - r_2 = 0$. ■

Ha tehát G az R egy Gröbner-bázisa, akkor minden $f, g, h \in R$ esetén $g = f \mathrm{rem} G$ és $h = f \mathrm{rem} G \Rightarrow g = h$.

2.3.4. A Buchberger-algoritmus

Észrevehetjük, hogy a Hilbert-féle bázistétel nem konstruktív: nem ad választ arra, hogyan konstruáljuk meg egy I ideál Gröbner-bázisát. Az alábbiakban úgy okoskodunk, hogy azt vizsgáljuk, egy véges halmaz mikor nem Gröbner-bázisa az I ideálnak.

Legyenek $g, h \in R$ nem nulla polinomok, $\alpha = (\alpha_1, \ldots, \alpha_n) = \mathrm{discr}(g)$, $\beta = (\beta_1, \ldots, \beta_n) = \mathrm{discr}(h)$, $\gamma = (\max\{\alpha_1, \beta_1\}, \ldots, \max\{\alpha_n, \beta_n\})$. A g és h polinomok ***S*-polinomján** az

$$S(g,h) = \frac{x^\gamma}{\mathrm{lc}(g)} g - \frac{x^\gamma}{\mathrm{lc}(h)} h \in R$$

polinomot értjük. Észrevehetjük, hogy $S(g,h) = -S(h,g)$, továbbá $x^\gamma / \mathrm{lc}(g), x^\gamma / \mathrm{lc}(h) \in R$ így $S(g,h) \in \langle g, h \rangle$. A most következő, bizonyítás nélkül közölt tétel egy egyszerű tesztet ad egy halmaz Gröbner-bázis mivoltára.

2.18. tétel. *A $G = \{g_1, \ldots, g_s\} \subseteq R$ halmaz akkor és csak akkor lesz a $\langle G \rangle$ ideál Gröbner-bázisa, ha*

$$S(g_i, g_j) \mathrm{rem}\ (g_1, \ldots, g_s) = 0 \ \text{ minden } 1 \le i < j \le s \ \text{ esetén}.$$

Az S-polinomok segítségével könnyen adható Gröbner-bázist konstruáló algoritmus (Buchberger, 1965): adott $f_1, \ldots, f_s \in R = F[x_1, \ldots, x_n]$ és adott $\prec$ monomiális rendezés mellett az alábbi algoritmus megadja az $I = \langle f_1, \ldots, f_s \rangle$ ideál egy $G \subseteq R$ Gröbner-bázisát.

GRÖBNER-BÁZIS$(f_1, \ldots, f_s)$

```
1  G ← {f_1, ..., f_s}
2  P ← {(f_i, f_j) | f_i, f_j ∈ G, i < j, f_i ≠ f_j}
3  while P ≠ ∅
4        do (f, g) ← egy tetszőleges pár P-ből
5           P ← P \ (f, g)
6           r ← S(f, g)rem G
7           if r ≠ 0
8              then G ← G ∪ {r}
9                   P ← P ∪ {(f, r) | f ∈ G}
10 return G
```

Először megmutatjuk, hogy a GRÖBNER-BÁZIS algoritmus helyesen működik. Az algoritmus futásának bármely pillanatában G az I ideál bázisa, hiszen kezdetben az, a továbbiakban pedig kizárólag olyan elemek kerülnek G-be, melyeket G elemeiből képzett S-polinomok G-vel vett maradékos osztásaként kapunk. Amennyiben az algoritmus befejeződik, az összes lehetséges képezhető S-polinom G-vel vett maradéka nulla, így a (2.18) tétel miatt G egy Gröbner-bázis.

Most bebizonyítjuk, hogy az algoritmus befejeződik. Legyenek G és G^* a pszeudokód **while** ciklusa két egymást követő végrehajtásakor kapott halmazok. Nyilván $G \subseteq G^*$, továbbá $\langle \mathrm{lc}(G)\rangle \subseteq \langle \mathrm{lc}(G^*)\rangle$. De a (2.15) következmény miatt az egymást követő iterációk $\langle \mathrm{lc}(G)\rangle$ ideállánca stabilizálódik, vagyis véges számú lépés után $\langle \mathrm{lc}(G)\rangle = \langle \mathrm{lc}(G^*)\rangle$ teljesül. Azt állítjuk, hogy ekkor $G = G^*$. Legyen $f, g \in G$ és $r = S(f,g)\mathrm{rem}\ G$. Ekkor $r \in G^*$ és vagy $r = 0$ vagy $\mathrm{lc}(r) \in \langle \mathrm{lc}(G^*)\rangle = \langle \mathrm{lc}(G)\rangle$, amiből a maradék képzésének definíciója miatt $r = 0$ következik.

2.14. példa. Legyen $F = \mathbb{Q}$, $<=<_{plex}$, $z \prec y \prec x$, $f_1 = x - y - z$, $f_2 = x + y - z^2$, $f_3 = x^2 + y^2 - 1$. Ekkor a pszeudokód első sora miatt $G = \{f_1, f_2, f_3\}$, a második sorból pedig $P = \{(f_1, f_2), (f_1, f_3), (f_2, f_3)\}$.

A **while** ciklus első végrehajtása során válasszuk az (f_1, f_2) párt. Ekkor $P = \{(f_1, f_3), (f_2, f_3)\}$, $S(f_1, f_2) = -2y - z + z^2$ és $r = f_4 = S(f_1, f_2)\mathrm{rem}\ G = -2y - z + z^2$. Ezért $G = \{f_1, f_2, f_3, f_4\}$ és $P = \{(f_1, f_3), (f_2, f_3), (f_1, f_4), (f_2, f_4), (f_3, f_4)\}$.

A ciklus második végrehajtásakor válasszuk az (f_1, f_3) párt. Ekkor $P = P \setminus \{f_1, f_3\}$, $S(f_1, f_3) = -xy - xz - y^2 + 1$, $r = f_5 = S(f_1, f_3)\mathrm{rem}\ G = -1/2z^4 - 1/2z^2 + 1$, így $G = \{f_i \mid 1 \le i \le 5\}$ és $P = \{(f_2, f_3), (f_1, f_4), \ldots, (f_3, f_4), (f_1, f_5), \ldots, (f_4, f_5)\}$.

A ciklus harmadik végrehajtása során válasszuk az (f_2, f_3) párt. Ekkor $P = P \setminus \{(f_2, f_3)\}$, $S(f_2, f_3) = xy - xz^2 - y^2 + 1$, $r = S(f_2, f_3)\mathrm{rem}\ G = 0$.

A **while** ciklus negyedik végrehajtása során válasszuk az (f_1, f_4) párt. Ekkor $P = P \setminus \{f_1, f_4\}$, $S(f_1, f_4) = 2y^2 + 2yz + xz - xz^2$, $r = S(f_1, f_4)\mathrm{rem}\ G = 0$.

Hasonlóan, az összes fennmaradó pár S-polinomjának G-vel vett maradékos osztásakor a maradék 0, így a visszatérő érték $G = \{x - y - z, x + y - z^2, x^2 + y^2 - 1, -2y - z + z^2, -1/2z^4 - 1/2z^2 + 1\}$ egy Gröbner-bázis.

2.3.5. Redukált Gröbner-bázis

A Buchberger-algoritmus által eredményezett Gröbner-bázis általában nem minimális és nem egyértelmű. Szerencsére egy kis ravaszkodással mindkettő elérhető.

2.19. lemma. *Ha G az $I \subseteq R$ ideál egy Gröbner-bázisa és $lc(g) \in \langle lc(G \setminus \{g\})\rangle$, akkor $G \setminus \{g\}$ is egy Gröbner-bázisa I-nek.*

Azt mondjuk, hogy a $G \subseteq R$ halmaz az $I = \langle G\rangle$ ideál ***minimális Gröbner-bázisa,*** ha Gröbner-bázis és minden $g \in G$ esetén

- $\mathrm{lc}(g) = 1$,
- $\mathrm{lc}(g) \notin \langle \mathrm{lc}(G \setminus \{g\})\rangle$.

A G Gröbner-bázis egy $g \in G$ eleme ***redukált*** a G-re nézve, ha g egyetlen monomja sincs az $\langle \mathrm{lc}(G \setminus \{g\})\rangle$ ideálban. Egy minimális G Gröbner-bázis redukált, ha G-re vonatkozóan minden eleme redukált.

2.20. tétel. *Minden ideálhoz egyértelműen létezik egy redukált Gröbner-bázis.*

2.15. példa. A 2.14. példát alapul véve nemcsak G, hanem $G' = \{x-y-z, -2y-z+z^2, -1/2z^4-1/2z^2+1\}$ is Gröbner-bázis. Nem nehéz megmutatni, hogy $G_r = \{x - 1/2z^2 - 1/2z, y - 1/2z^2 - 1/2z, z^4 + z^2 - z\}$ redukált Gröbner-bázis.

2.3.6. A Gröbner-bázis számítási bonyolultsága

A Gröbner-bázis elmélet kialakulása óta eltelt fél évszázad sem volt elég teljesen tisztázni az algoritmus számítási bonyolultságát. A tapasztalatok azt mutatják, hogy számítási tárrobbanással állunk szemben. De most, ellentétben az euklideszi algoritmus variánsainál látott számítási tárrobbanással, a növekedés ütemét nem lehet kordában tartani. A Gröbner-bázis számításnál fellépő számítási bonyolultságot szokás az *EXPSPACE*-teljes osztályba sorolni. Legyenek $f, f_1, \ldots, f_s \in F[x_1, \ldots, x_n]$ az F test feletti legfeljebb d-ed fokú polinomok ($<=\prec_{tdeg}$). Ha $f \in \langle f_1, f_2, \ldots f_s\rangle$, akkor

$$f = f_1 g_1 + \cdots + f_s g_s$$

olyan $g_1, \ldots, g_s \in F[x_1, \ldots, x_n]$ polinomokra, melyek fokai $\beta = \beta(n, d) = (2d)^{2^n}$-nel felülről korlátosak. Ez a duplán exponenciális korlát lényegében elkerülhetetlen, amit számos példa bizonyít. Sajnos, az $F = \mathbb{Q}$ esetben az ideál-tartalmazás probléma ilyen. Szerencsére, speciális esetekben drasztikus redukció érhető el. Ha $f = 1$ (a Hilbert-féle Nullstellensatz), akkor a $d = 2$ esetben $\beta = 2^{n+1}$, a $d > 2$ esetben pedig $\beta = d^n$. Márpedig a $V(f_1, \ldots, f_s)$ varietás pontosan akkor üres, ha $1 \in \langle f_1, f_2, \ldots f_s\rangle$, vagyis a polinomegyenlet-rendszerek megoldhatósága *PSPACE*-beli probléma. Számos eredmény szól amellett, hogy bizonyos feltételek mellett az (általános) ideál-tartalmazás probléma is *PSPACE*-beli. Ilyen feltétel például, hogy $\langle f_1, f_2, \ldots f_s\rangle$ zéró-dimenziós.

Ezen számítási bonyolultság ellenére a Gröbner-bázis elmélet komoly sikereket könyvelhet el: automatikus geometriai tételbizonyítás, robotok mozgásvezérlése és polinomegyenlet-rendszerek megoldása talán a legelterjedtebb alkalmazási területek. Az alábbiakban felsoroljuk azokat az területeket, ahol az elmélet sikeresen alkalmazható.

- *Polinomegyenletek ekvivalenciája.* Két polinomhalmaz pontosan akkor generálja ugyanazt az ideált, ha Gröbner-bázisuk megegyezik (tetszőleges monomiális rendezés mellett).

- *Polinomegyenletek megoldhatósága.* Az $f_i(x_1,\ldots,x_n)=0,\ 1\le i\le s$ egyenletrendszer pontosan akkor oldható meg, ha az $1\notin\langle f_1,\ldots,f_s\rangle$.
- *Polinomegyenletek megoldáshalmaza végessége.* Az $f_i(x_1,\ldots,x_n)=0,\ 1\le i\le s$ egyenletrendszernek pontosan akkor van véges számú megoldása, ha $\langle f_1,\ldots,f_s\rangle$-ben minden x_i változóhoz van olyan polinom, hogy az adott monomiális rendezés melletti főtagja x_i valamilyen hatványa.
- *Polinomegyenletek véges megoldásai száma.* Legyen az $f_i(x_1,\ldots,x_n)=0,\ 1\le i\le s$ egyenletrendszernek véges sok megoldása. Ekkor multiplicitással számolva a polinomegyenlet megoldásszáma azon monomok halmazának számossága, melyek nem többszörösei a $\langle f_1,\ldots,f_s\rangle$ Gröbner-bázisa elemei egyetlen főtagjának sem (tetszőleges monomiális rendezés mellett).
- *Kifejezések egyszerűsítése.*

A legutóbbira példát is mutatunk.

2.16. példa. Legyenek $a,b,c\in\mathbb{R}$ olyanok, hogy

$$a+b+c=3,\ \ a^2+b^2+c^2=9,\ \ a^3+b^3+c^3=24\ .$$

Számítsuk ki $a^4+b^4+c^4$ értékét. Legyenek tehát $f_1=a+b+c-3$, $f_2=a^2+b^2+c^2-9$ és $a^3+b^3+c^3-24$ az $\mathbb{R}[a,b,c]$ elemei, továbbá legyen $\prec=\prec_{plex}$, $c\prec b\prec a$. Ekkor az $\langle f_1,f_2,f_3\rangle$ Gröbner-bázisa

$$G=\{a+b+c-3,\ b^2+c^2-3b-3c+bc,\ 1-3c^2+c^3\}\ .$$

Így $a^4+b^4+c^4\mathrm{rem}\,G=69$, ami a feladat megoldása.

Gyakorlatok

2.3-1. Bizonyítsuk be, hogy a $\prec_{lex},\prec_{plex},\prec_{grlex}$ és $\prec_{tdeg}$ monomiális rendezések valóban megengedettek.

2.3-2. Legyen $\prec$ egy monomiális rendezés R-en, $f,g\in R\setminus\{0\}$. Lássuk be az alábbiakat:

a. $\mathrm{discr}(fg)=\mathrm{discr}(f)+\mathrm{discr}(g)$,

b. ha $f+g\neq 0$, akkor $\mathrm{discr}(f+g)\le\max\{\mathrm{discr}(f),\mathrm{discr}(g)\}$, ahol $\mathrm{discr}(f)\neq\mathrm{discr}(g)$ esetén egyenlőség teljesül.

2.3-3. Legyen $f=2x^4y^2z-3x^4yz^2+4xy^4z^2-5xy^2z^4+6x^2y^4z-7x^2yz^4\in\mathbb{Q}[x,y,z]$.

a. Rendezzük a polinom tagjait $\prec_{plex},\prec_{grlex}$ és $\prec_{tdeg}$ monomiális rendezések esetén, ahol mindhárom esetben $z\prec y\prec x$.

b. Mindhárom monomiális rendezés esetén határozzuk meg $\mathrm{discr}(f),\mathrm{lc}(f),\mathrm{lm}(f)$ és $\mathrm{lc}(f)$-et.

2.3-4.⋆ Bizonyítsuk be a Dickson-lemmát.

2.3-5. Számítsuk ki az $I=\langle x^2+y-1,xy-x\rangle\subseteq\mathbb{Q}[x,y]$ ideál Gröbner-bázisát és redukált Gröbner-bázisát a $\prec=\prec_{lex}$ monomiális rendezés esetén, ahol $y\prec x$. Határozzuk meg, hogy az alábbi polinomok közül melyik eleme az ideálnak: $f_1=x^2+y^2-y$, $f_2=3xy^2-4xy+x+1$.

2.4. Szimbolikus integrálás

A határozatlan integrálás problémája egy adott f függvényhez olyan g függvényt találni, amelynek deriváltja f, azaz amelyre $g'(x)=f(x)$; ezen összefüggés jelölésére az

$\int f(x)\,dx = g(x)$ jelölést is használjuk. A bevezető analízis előadásokon a határozatlan integrálási problémák megoldására különböző módszerekkel próbálkozunk, amelyek között heurisztikus módon próbálunk választani: helyettesítések, trigonometrikus helyettesítések, parciális integrálás stb. Csak a racionális törtfüggvények integrálására szokás algoritmikus módszert használni.

Megmutatható, hogy a határozatlan integrálás teljes általánosságban algoritmussal megoldhatatlan probléma. Tehát csak arra van lehetőségünk, hogy minél nagyobb algoritmussal megoldható részt keressünk.

Az első lépés a probléma algebraizálása: teljesen elvonatkoztatunk minden analízisbeli fogalomtól, és a differenciálást úgy tekintjük, mint egy új (egyváltozós) algebrai műveletet, amely az összeadással és a szorzással adott kapcsolatban van, és ennek a műveletnek az „inverzét” keressük. Ez a felfogás vezetett a differenciálalgebra fogalmának bevezetéséhez.

A komputeralgebra rendszerek (például a Maple) integráló rutinja, hasonlóan hozzánk, először néhány heurisztikus módszerrel próbálkozik. Polinomok (vagy kicsit általánosabban, véges Laurent-sorok) integrálja könnyen meghatározható. Ezután egy egyszerű táblázatban való keresés következik (a Maple esetén például 35 alapintegrál felhasználása). Lehet persze könyvekből bevitt integráltáblázatokat is használni. Ezután speciális eseteket kereshetünk, amelyre megfelelő módszerek ismertek. Például

$$e^{ax+b}\sin(cx+d)\cdot p(x)$$

alakú integrálok esetén, ahol p polinom, parciális integrálás használható az integrál meghatározására. Ha a fenti módszerek sikertelenek, akkor helyettesítéssel próbálkozik az úgynevezett „beosztás a deriválttal” módszer formájában: ha az integrandus összetett kifejezés, akkor minden $f(x)$ részkifejezésére osztunk f deriváltjával, majd kipróbáljuk, hogy $u = f(x)$ helyettesítés után a kapott kifejezésből eltűnik-e x. Ezek az egyszerű módszerek meglepően sok határozatlan integrál kiszámítására elegendőek. Nagy előnyük, hogy az egyszerű problémákat gyorsan oldják meg. Ha nem vezetnek célhoz, akkor próbálkozunk az algoritmikus módszerekkel. Ezek között az első a racionális törtfüggvények integrálása. Mint látni fogjuk, a komputeralgebra rendszerekben használt változat lényegesen eltér a kézi számolásra használt változattól, mert a cél az, hogy a futási idő még bonyolult esetekben is kicsi legyen, és az eredmény a lehető legegyszerűbb formában keletkezzen. Az elemi függvények integrálására használt Risch-algoritmus a racionális törtfüggvények integrálására használt algoritmusokon alapul. Ezt is ismertetjük, de nem tárgyaljuk teljes részletességgel. A bizonyításokra inkább csak utalunk.

2.4.1. Racionális függvények integrálása

Ebben a pontban bevezetjük a differenciáltest és a differenciáltest bővítésének fogalmát, majd ismertetjük Hermite módszerét.

Differenciáltest

Legyen K egy nulla karakterisztikájú test, amelyen adott egy $f \mapsto f'$ leképezése K-nak önmagába az alábbi két tulajdonsággal:

(1) $(f+g)' = f' + g'$ (additivitás);

(2) $(fg)' = f'g + g'f$ (Leibniz-szabály).

Ekkor az $f \mapsto f'$ leképezést ***differenciáloperátornak, differenciálásnak*** vagy ***deriválásnak,*** K-t pedig ***differenciáltestnek*** nevezzük. A $C = \{c \in K : c' = 0\}$ halmaz a ***konstansok részteste*** K-ban. Ha $f' = g$, akkor azt is írjuk, hogy $f = \int g$. Nyilván bármely $c \in C$ konstansra $f + c = \int g$. Egy $0 \neq f \in K$ elem ***logaritmikus deriváltján*** f'/f-et értjük. (Formálisan ez „$\log(f)$ deriváltja”.)

2.21. tétel. *Az előző definíció jelöléseivel, a deriválás szokásos tulajdonságai teljesülnek:*

(1) $0' = 1' = (-1)' = 0$;

(2) *a differenciálás C-lineáris:* $(af + bg)' = af' + bg'$, *ha* $f, g \in K$, $a, b \in C$;

(3) *ha* $g \neq 0$, f *tetszőleges, akkor* $(f/g)' = (f'g - g'f)/g^2$;

(4) $(f^n)' = nf'f^{n-1}$, *ha* $0 \neq f \in K$ *és* $n \in \mathbb{Z}$;

(5) $\int fg' = fg - \int gf'$, *ha* $f, g \in K$ (***parciális integrálás***).

2.17. példa. (1) Az előző definíció jelöléseivel, az $f \mapsto 0$ leképezés K-n a *triviális deriválás*, erre $C = K$.

(2) Legyen $K = \mathbb{Q}(x)$. Egyetlen olyan differenciálás van $\mathbb{Q}(x)$-en, a szokásos, amelyre $x' = 1$. Erre a konstansok $\mathbb{Q}$ elemei. Valóban, indukcióval $n' = 0$, ha $n \in \mathbb{N}$, és így $\mathbb{Z}$ illetve $\mathbb{Q}$ elemei is konstansok. Indukcióval adódik, hogy a hatványfüggvények deriváltja a szokásos, ahonnan a linearitás miatt a polinomoké is, így a hányados differenciálási szabálya szerint kapjuk az állítást. Nem nehéz kiszámolni, hogy a szokásos differenciálásra a konstansok $\mathbb{Q}$ elemei.

(3) Ha $K = C(x)$, ahol C tetszőleges nulla karakterisztikájú test, akkor egyetlen olyan differenciálás van K-n, a szokásos, amelyre a konstansok részteste C és $x' = 1$; az állítás hasonlóan adódik, mint az előző.

Ha C tetszőleges nulla karakterisztikájú test és $K = C(x)$ a szokásos differenciálással, akkor $1/x$ nem deriváltja semminek. (Az állítás bizonyítása nagyon hasonlít annak bizonyításához, hogy $\sqrt{2}$ irracionális, de kettővel való oszthatóság helyett az x polinommal való oszthatósággal kell dolgoznunk.)

A példa azt mutatja, hogy $1/x$ és más hasonló függvények integrálásához a differenciáltestet bővíteni kell. A racionális törtfüggvények integrálásához elég lesz logaritmusokkal bővíteni.

Differenciáltest bővítése

Legyen L egy differenciáltest, $K \subset L$ pedig egy részteste L-nek. Ha a differenciálás nem vezet ki K-ból, akkor azt mondjuk, hogy K egy ***differenciálrészteste*** L-nek, illetve hogy L egy ***differenciálbővítése*** K-nak. Ha valamely $f, g \in L$-re $f' = g'/g$, azaz ha f deriváltja a g logaritmikus deriváltja, akkor azt írjuk, hogy $f = \log g$. (Megjegyezzük, hogy log, ugyanúgy, mint $\int$, nem függvény, hanem reláció. Más szóval a log itt egy absztrakt fogalom, nem pedig egy meghatározott alapú logaritmus függvény.) Ha $g \in K$ választható, akkor azt mondjuk, hogy f ***logaritmikus K felett.***

2.18. példa. (1) Legyen $g = x \in K = \mathbb{Q}(x)$, $L = \mathbb{Q}(x, f)$, ahol f egy új határozatlan, és legyen $f' = g'/g = 1/x$, azaz $f = \log(x)$. Ekkor $\int 1/x\,dx = \log(x)$.

(2) Hasonlóan,

$$\int \frac{1}{x^2 - 2} = \frac{\sqrt{2}}{4}\log(x - \sqrt{2}) - \frac{\sqrt{2}}{4}\log(x + \sqrt{2})$$

a $\mathbb{Q}(\sqrt{2})(x, \log(x-\sqrt{2}), \log(x+\sqrt{2}))$ differenciáltestben van.

(3) Mivel

$$\int \frac{1}{x^3+x} = \log(x) - \frac{1}{2}\log(x+i) - \frac{1}{2}\log(x-i) = \log(x) - \frac{1}{2}\log(x^2+1)\,,$$

az integrál tekinthető

$$\mathbb{Q}(x, \log(x), \log(x^2+1))$$

elemének is és

$$\mathbb{Q}(i)(x, \log(x), \log(x-i), \log(x+i))$$

elemének is. Nyilván célszerűbb az első lehetőséget választani, mert ekkor az alaptest bővítésére nincs szükség.

Hermite módszere

Legyen K egy nulla karakterisztikájú test, $f, g \in K[x]$ nemnulla és relatív prím polinomok. Az $\int f/g$ integrál kiszámításához Hermite módszerével olyan $a, b, c, d \in K[x]$ polinomokat találhatunk, amelyekre

$$\int \frac{f}{g} = \frac{c}{d} + \int \frac{a}{b}\,, \tag{2.15}$$

ahol $\deg a < \deg b$ és b négyzetmentes főpolinom. A c/d racionális függvényt az integrál ***racionális részének,*** az $\int a/b$ kifejezést pedig az integrál ***logaritmikus részének*** nevezzük. A módszer elkerüli g-nek a (felbontási testben vagy valamely még bővebb testben) lineáris tényezőkre való bontását, sőt, még a K felett irreducibilis tényezőkre való felbontást is.

Nyilván feltehetjük, hogy g főpolinom. Maradékos osztással $f = pg + h$, ahol $\deg h < \deg g$, így $f/g = p + h/g$. A p polinomrész integrálása triviális. Határozzuk meg g négyzetmentes felbontását, azaz keressünk olyan $g_1, \ldots, g_m \in K[x]$ négyzetmentes és páronként relatív prím főpolinomokat, amelyekre $g_m \neq 1$ és $g = g_1 g_2^2 \cdots g_m^m$. Bontsunk parciális törtekre (ez euklideszi algoritmussal megvalósítható):

$$\frac{h}{g} = \sum_{i=1}^{m} \sum_{j=1}^{i} \frac{h_{i,j}}{g_i^j}\,,$$

ahol minden $h_{i,j}$ foka kisebb, mint g_i foka.

A Hermite-redukció a következő lépés ismétlése: ha $j > 1$, akkor az $\int h_{i,j}/g_i^j$ integrált egy racionális függvény és egy, az eredetihez hasonló alakú integrál összegére redukáljuk, amelyben j eggyel kisebb. Felhasználva, hogy g_i négyzetmentes, azt kapjuk, hogy $\mathrm{lnko}(g_i, g_i') = 1$, így a bővített euklidészi algoritmussal kaphatunk olyan $s, t \in K[x]$ polinomokat, amelyekre $sg_i + tg_i' = h_{i,j}$ és $\deg s, \deg t < \deg g_i$. Innen parciális integrálással

$$\begin{aligned}
\int \frac{h_{i,j}}{g_i^j} &= \int \frac{t \cdot g_i'}{g_i^j} + \int \frac{s}{g_i^{j-1}} \\
&= \frac{-t}{(j-1)g_i^{j-1}} + \int \frac{t'}{(j-1)g_i^{j-1}} + \int \frac{s}{g_i^{j-1}} \\
&= \frac{-t}{(j-1)g_i^{j-1}} + \int \frac{s + t'/(j-1)}{g_i^{j-1}}\,.
\end{aligned}$$

Megmutatható, hogy gyors algoritmusokat választva, ha $\deg f, \deg g < n$, akkor az eljárás $O(M(n)\log n)$ darab K testbeli műveletet igényel, ahol $M(n)$ a legfeljebb n-ed fokú polinomok szorzásához szükséges műveletek számának korlátja.

Hermite módszerének van olyan változata is, amely elkerüli h/g parciális törtekre bontását. Ha $m = 1$, akkor g négyzetmentes. Ha $m > 1$, akkor legyen

$$g^* = g_1 g_2^2 \cdots g_{m-1}^{m-1} = \frac{g}{g_m^m} \ .$$

Mivel $\mathrm{lnko}(g_m, g^* g_m') = 1$, léteznek olyan $s, t \in K[x]$ polinomok, amelyekre

$$s g_m + t g^* g_m' = h \ .$$

Mindkét oldalt osztva $g = g^* g_m^m$-el és parciálisan integrálva

$$\int \frac{h}{g} = \frac{-t}{(m-1) g_m^{m-1}} + \int \frac{s + g^* t'/(m-1)}{g^* g_m^{m-1}} \ ,$$

így m-et eggyel csökkentettük.

Megjegyezzük, hogy a és c a határozatlan együtthatók módszerével is meghatározhatók (Horowitz módszere). Osztás után feltehetjük, hogy $\deg f < \deg g$. Mint az algoritmusból látszik, $d = g_2 g_3^2 \cdots g_m^{m-1}$ és $b = g_1 g_2 \cdots g_m$ választható. (2.15) differenciálásával az a polinom $\deg b$ darab és a c polinom $\deg d$ darab együtthatójára, tehát összesen n darab együtthatóra egy lineáris egyenletrendszert kapunk. Ez a módszer általában nem olyan gyors, mint Hermite módszere.

Az alábbi algoritmus az x változó adott f/g racionális függvényére elvégzi a Hermite-redukciót.

HERMITE-REDUKCIÓ(f, g)

```
 1  p ← quo(f, g)
 2  h ← rem(f, g)
 3  (g[1], ..., g[m]) ← NÉGYZETMENTES(g)
 4  bontsuk (h/g)-t parciális törtekre, számítsuk ki a g[i]^j-hez tartozó h[i, j] számlálókat
 5  rac ← 0
 6  int ← 0
 7  for i ← 1 to m
 8      do int ← int + h[i, 1]/g[i]
 9      for j ← 2 to i
10          do n ← j
11          while n > 1
12                  do s és t meghatározása az s · g[i] + t · g[i]' = h[i, j] egyenletből
13                  n ← n − 1
14                  rac ← rac − (t/n)/g[i]^n
15                  h[i, n] ← s + t'/n
16          int ← int + h[i, 1]/g[i]
17  red ← rac + ∫ p + ∫ int
18  return red
```

Ha valamely nulla karakterisztikájú K testre az $\int a/b$ integrált akarjuk kiszámítani, ahol

$a, b \in K[x]$ nem nulla relatív prím polinomok, $\deg a < \deg b$, b négyzetmentes és főpolinom, akkor eljárhatunk úgy, hogy a b polinom L felbontási testében felírjuk b-t gyöktényezős alakban: $b = \prod_{k=1}^{n}(x - ła_k)$, majd L felett parciális törtekre bontunk: $a/b = \sum_{k=1}^{n} c_k/(x - ła_k)$, végül integrálunk:

$$\int \frac{a}{b} = \sum_{k=1}^{n} c_k \log(x - ła_k) \in L(x, \log(x - ła_1), \ldots, \log(x - ła_n)) \; .$$

Ennek az eljárásnak, mint az $1/(x^3 + x)$ függvény példáján láttuk, a hátránya, hogy az L testbővítés foka túl magas lehet. Előfordulhat az is, hogy L foka K felett $n!$, ami teljesen kezelhetetlen esetekhez vezet. Másrészt az sem világos, hogy kell-e az adott esetben az alaptestet bővíteni, például az $1/(x^2 - 2)$ függvény esetében nem lehet-e az integrálást az alaptest bővítése nélkül elvégezni. A következő tétel lehetővé teszi, hogy L testbővítés fokát a lehető legkisebbre válasszuk.

2.22. tétel (Rothstein–Trager-féle integráló algoritmus). *Legyen K nulla karakterisztikájú test, $a, b \in K[x]$ nem nulla relatív prím polinomok, $\deg a < \deg b$, b négyzetmentes és főpolinom. Ha L algebrai bővítése K-nak, $c_1, \ldots, c_k \in L \setminus K$ páronként különböző, $v_1, \ldots, v_k \in L[x] \setminus L$ pedig négyzetmentes és páronként relatív prím főpolinomok, akkor az alábbiak ekvivalensek:*

(1) $$\int \frac{a}{b} = \sum_{i=1}^{k} c_i \log v_i \; ,$$

(2) *Az $r = res_x(b, a - yb') \in K[y]$ polinom L felett lineáris tényezőkre bomlik, $c_1, \ldots, c_k$ pontosan az r különböző gyökei, és $v_i = lnko(b, a - c_i b')$, ha $i = 1, \ldots, k$. Itt res_x az x határozatlanban vett rezultáns.*

2.19. példa. Tekintsük ismét az $\int 1/(x^3+x)\,dx$ integrál kiszámításának problémáját. Ebben az esetben

$$r = \mathrm{res}_x(x^3 + x, 1 - y(3x^2 + 1)) = -4z^3 + 3y + 1 = -(2y + 1)^2(y - 1) \; ,$$

amelynek gyökei $c_1 = 1$ és $c_2 = -1/2$. Így

$$\begin{aligned} v_1 &= \mathrm{lnko}(x^3 + x, 1 - (3x^2 + 1)) = x \; , \\ v_2 &= \mathrm{lnko}(x^3 + x, 1 + \frac{1}{2}(3x^2 + 1)) = x^2 + 1 \; . \end{aligned}$$

Az előző tétel alapján könnyen felírható integráló algoritmus kicsit javítható: $v_i = \mathrm{lnko}(b, a - c_i b')$-nek (az L test feletti számításokkal történő) kiszámítása helyett v_i megkapható K feletti számítással is, a Bővített-Euklidesz-normalizált algoritmus segítségével. Ezt egymástól függetlenül Trager, illetve Lazard és Rioboo vette észre. Nem nehéz belátni, hogy az így adódó teljes integráló algoritmus futásideje $O(nM(n)\lg n)$, ha $\deg f, \deg g < n$.

2.23. tétel (Lazard–Rioboo–Trager-formula). *Az előző tétel jelöléseivel, jelölje e a c_i-nek mint az $r = res_x(b, a - yb')$ polinom gyökének a multiplicitását. Ekkor*

(1) $\deg v_i = e$;

(2) *ha* $w(x,y) \in K(y)[x]$ *jelöli a* b*-re és* $a - yb'$*-re* $K(y)[x]$*-ben végrehajtott* Bővített-Euklidesz-normalizált*-algoritmus* e*-ed fokú maradékát, akkor* $v_i = w(x, c_i)$.

Az alábbi algoritmus a Rothstein–Trager-módszer Lazard–Rioboo–Trager-féle javítása. Az x változó adott a/b racionális függvényére kiszámítjuk $\int a/b$-t, ahol b négyzetmentes főpolinom és $\deg a < \deg b$.

Logaritmikus-rész-integrál(a, b, x)

```
 1  a szubrezultáns algoritmussal legyen r(y) ← res_x(b, a − yb′), továbbá
 2  a számítás során legyen w_e(x, y) az e-ed fokú maradék
 3  (r_1(y), ..., r_k(y)) ←Négyzetmentes(r(y))
 4  int ← 0
 5  for i ← 1 to k
 6      do if r_i(y) ≠ 1
 7            then w(y) ← w_i(x, y) együtthatóinak lnko-ja
 8                 (l(y), s(y), t(y)) ←Bővített-Euklidesz-normalizált(w(y), r_i(y))
 9                 w_i(x, y) ←Primitív-rész(rem(s(y) · w_i(x, y), r_i(y)))
10                 (r_{i,1}, ..., r_{i,k}) ←Faktorok(r_i)
11                 for j ← 1 to k
12                     do d ← deg(r_{i,j})
13                        c ←Megoldás(r_{i,j}(y) = 0, y)
14                        if d = 1
15                           then int ← int + c · log(w_i(x, c))
16                           else for n ← 1 to d
17                                   do int ← int + c[n] · log(w_i(x, c[n]))
18  return int
```

2.20. példa. Tekintsük ismét az $\int 1/(x^2-2)\,dx$ integrál kiszámításának problémáját. Ebben az esetben

$$r = \operatorname{res}_x(x^2 - 2, 1 - y \cdot 2x) = -8y^2 + 1\,.$$

A polinom irreducibilis $\mathbb{Q}[x]$-ben, így $\mathbb{Q}$ bővítése elkerülhetetlen. Az r gyökei $\pm 1/\sqrt{8}$. A $\mathbb{Q}(y)$ feletti Bővített-Euklidesz-normalizált-algoritmusból $w_1(x,y) = x - 1/(2y)$, így az integrál

$$\int \frac{1}{x^2-2}\,dx = \frac{1}{\sqrt{8}}\log(x - \sqrt{2}) - \frac{1}{\sqrt{8}}\log(x + \sqrt{2})\,.$$

2.4.2. Risch integráló algoritmusa

Meglepő módon, a racionális függvények integrálására talált módszerek általánosíthatók a szokásos függvényeket (sin, exp stb.) és inverzeiket tartalmazó kifejezések integrálására. A komputeralgebra rendszerek meglepően bonyolult kifejezések integrálását is elvégzik, néha azonban látszólag igen egyszerű esetekben sem adják meg az integrált, például az

$\int x/(1+e^x)\,dx$ kifejezést kiértékeletlenül kapjuk vissza, vagy az eredmény nem elemi speciális függvényt tartalmaz, például az integrállogaritmus függvényt. Ez azért van, mert ezekben az esetekben az integrál nem is fejezhető ki „zárt alakban”.

Bár a „zárt alakban” kifejezhető integrálokra vonatkozó alapvető eredményt Liouville 1833-ban találta, a megfelelő algoritmikus módszereket Risch csak 1968-ban fejlesztette ki.

Elemi függvények

A „zárt alakban” megadható függvényeknek azokat a függvényeket szokás tekinteni, amelyek felépíthetők a racionális függvények, az exponenciális és logaritmus függvény, a trigonometrikus és hiperbolikus függvények és inverzeik, valamint gyökvonások, illetve sokkal általánosabban polinom függvények „inverzei”, azaz egyenletek gyökeinek képzése segítségével, tehát ezen függvények egymásba helyettesítésével.

Észrevehetjük, hogy míg az $\int 1/(1+x^2)\,dx$ integrált $\operatorname{arctg}(x)$ alakban szokás megadni, a racionális törtfüggvény integrálására megadott algoritmus az eredményt

$$\int \frac{1}{1+x^2}\,dx = \frac{i}{2}\log(x+i) - \frac{i}{2}\log(x-i)$$

alakban adja. Mivel $\mathbb{C}$-ben a trigonometrikus és hiperbolikus függvények kifejezhetők az exponenciális függvény, inverzeik pedig a logaritmus függvény segítségével, csak az exponenciális és a logaritmus függvényre szorítkozhatunk. Meglepő módon kiderül, hogy az integrálok kifejezéséhez (algebrai számokkal való bővítések mellett) itt is csak logaritmusokkal való bővítésekre van szükség.

Exponenciális elemek

Legyen L a K differenciáltest differenciálbővítése. Ha egy $\theta \in L$ elemhez van olyan $u \in K$, hogy $\theta'/\theta = u'$, azaz ha θ logaritmikus deriváltja K valamely elemének a deriváltja, akkor azt mondjuk, hogy θ ***exponenciális*** K felett, és azt írjuk $\theta = \exp(u)$. Ha csak az teljesül, hogy a $\theta \in L$ elemhez van olyan $u \in K$, hogy $\theta'/\theta = u$, azaz, ha θ logaritmikus deriváltja K eleme, akkor azt mondjuk, hogy θ ***hiperexponenciális*** K felett.

A K felett logaritmikus, exponenciális vagy hiperexponenciális elemek lehetnek algebraiak vagy transzcendensek K felett.

Elemi kiterjesztések

Legyen L a K differenciáltest differenciálbővítése. Ha

$$L = K(\theta_1, \theta_2, \ldots, \theta_n)\,,$$

ahol $j = 1, 2, \ldots, n$-re θ_j logaritmikus, exponenciális vagy algebrai

$$K_{j-1} = K(\theta_1, \ldots, \theta_{j-1})$$

felett ($K_0 = K$), akkor azt mondjuk, hogy L a K ***elemi kiterjesztése.*** Ha $j = 1, 2, \ldots, n$-re θ_j transzcendens és logaritmikus vagy transzcendens és exponenciális K_{j-1} felett, akkor azt mondjuk, hogy L a K ***transzcendens elemi kiterjesztése.***

Legyen $C(x)$ a racionális függvények teste $C_{\mathbb{C}}$ konstans testtel és a szokásos differenciálással. Ennek egy elemi kiterjesztését ***elemi függvénytestnek***, transzcendens elemi kiterjesztését pedig ***transzcendens elemi függvénytestnek*** nevezzük.

2.21. példa. Az $f = \exp(x) + \exp(2x) + \exp(x/2)$ függvény $f = \theta_1 + \theta_2 + \theta_3 \in \mathbb{Q}(x, \theta_1, \theta_2, \theta_3)$ alakban írható, ahol $\theta_1 = \exp(x)$, $\theta_2 = \exp(2x)$, $\theta_3 = \exp(x/2)$. Nyilván θ_1 exponenciális $\mathbb{Q}(x)$ felett, θ_2 exponenciális $\mathbb{Q}(x, \theta_1)$ felett és θ_3 exponenciális $\mathbb{Q}(x, \theta_1, \theta_2)$ felett. Mivel $\theta_2 = \theta_1^2$, $\mathbb{Q}(x, \theta_1, \theta_2) = \mathbb{Q}(\theta_1)$, így f az egyszerűbb $f = \theta_1 + \theta_1^2 + \theta_3$ alakba írható. A θ_3 függvény nemcsak exponenciális $\mathbb{Q}(x, \theta_1)$ felett, hanem algebrai is, mivel $\theta_3^2 - \theta_1 = 0$, azaz $\theta_3 = \theta_1^{1/2}$. Így $f = \theta_1 + \theta_1^2 + \theta_1^{1/2} \in \mathbb{Q}(x, \theta_1, \theta_1^{1/2})$. De f még ennél egyszerűbb alakban is felírható:

$$f = \theta_3^2 + \theta_3^4 + \theta_3 \in \mathbb{Q}(x, \theta_3)\ .$$

2.22. példa. Az

$$f = \sqrt{\log(x^2 + 3x + 2)(\log(x + 1) + \log(x + 2))}$$

függvény felírható $f = \theta_4 \in \mathbb{Q}(x, \theta_1, \theta_2, \theta_3, \theta_4)$ alakban, ahol $\theta_1 = \log(x^2 + 3x + 1)$, $\theta_2 = \log(x + 1)$, $\theta_3 = \log(x+2)$, θ_4 pedig a $\theta_4^2 - \theta_1(\theta_2 + \theta_3) = 0$ algebrai egyenletnek tesz eleget, de sokkal egyszerűbben $f = \theta_1 \in \mathbb{Q}(x, \theta_1)$ alakban is.

2.23. példa. Az $f = \exp(\log(x)/2)$ függvény felírható $f = \theta_2 \in \mathbb{Q}(x, \theta_1, \theta_2)$ alakban, ahol $\theta_1 = \log(x)$ és $\theta_2 = \exp(\theta_1/2)$, így θ_1 logaritmikus $\mathbb{Q}(x)$ felett, θ_2 pedig exponenciális $\mathbb{Q}(x, \theta_1)$ felett. Azonban $\theta_2^2 - x = 0$, így θ_2 algebrai $\mathbb{Q}(x)$ felett, és $f(x) = x^{1/2}$.

Elemi függvények integrálása

Tetszőleges elemi függvénytestek elemeinek integrálját, ha elemi függvény, a Liouville-elv teljesen jellemezni fogja. Mindazonáltal az algebrai kiterjesztési lépések – ha nem csak a konstansok testét terjesztjük ki – nagy nehézségeket okoznak.

Itt csak a transzcendens elemi függvénytestek függvényeinek Risch-algoritmussal történő integrálásával foglalkozunk.

A gyakorlatban egy $\mathbb{Q}(\alpha_1, \ldots, \alpha_k)(x, \theta_1, \ldots, \theta_n)$ transzcendens elemi függvénytest valamely eleméről van szó, ahol $\alpha_1, \ldots, \alpha_k$ algebrai számok $\mathbb{Q}$ felett, az integrál pedig egy

$$\mathbb{Q}(\alpha_1, \ldots, \alpha_k, \ldots, \alpha_{k+h})(x, \theta_1, \ldots, \theta_n, \ldots, \theta_{n+m})$$

elemi függvénytest eleme lesz. Elvileg legegyszerűbb lenne a konstansok testét $\mathbb{C}$-nek választani, de mint a racionális függvények integrálásánál láttuk, ez nem lehetséges, mert csak bizonyos számtestekben, például algebrai számtestekben tudunk pontosan számolni, sőt, igyekeznünk kell az $\alpha_{k+1}, \ldots, \alpha_{k+h}$ algebrai számok számát és fokaikat a lehető legalacsonyabban tartani. Mindazonáltal a konstansok testének algebrai bővítéseit kezelhetjük dinamikusan, a szükségessé váló bővítésekről elképzelhetjük, hogy már előre elvégeztük őket, a gyakorlatban pedig mindig csak akkor hajtjuk végre a bővítést, ha szükségessé válik.

Miután a trigonometrikus és hiperbolikus függvények exponenciálisra, inverzeiknek pedig logaritmusra való konverzióját elvégeztük, az integrandust mint egy elemi függvénytest elemét kapjuk meg. A 2.21. és 2.22. példák mutatják, hogy bár lehet, hogy a függvény „első pillantásra” nem tűnik egy transzcendens elemi kiterjesztés elemének, mégis az, a 2.23. példa pedig azt, hogy lehet, hogy a függvény „első pillantásra” transzcendens elemi kiterjesztés elemének tűnik, mégsem az. Az első lépés tehát az, hogy a különböző exponenciális

és logaritmikus függvények közötti algebrai kapcsolatok vizsgálatával az integrandust mint transzcendens elemi függvénytest elemét állítjuk elő. Hogy ez hogyan lehetséges, azzal nem foglalkozunk. Hogy ez sikerült-e, az ellenőrizhető a Risch-től származó struktúra-tétel segítségével. A tétel bizonyítását elhagytuk. Szükségünk lesz egy definícióra.

Egy θ elem ***monomiális*** a K differenciáltest felett, ha transzcendens K felett, exponenciális vagy logaritmikus K felett, valamint K-ban és $K(\theta)$-ban ugyanazok a konstansok.

2.24. tétel (struktúra-tétel). *Legyen K a konstansok teste és $K_n = K(x, \theta_1, \ldots, \theta_n)$ egy differenciál-kiterjesztése $K(x)$-nek, amelyben a konstansok teste K. Tegyük fel, hogy minden θ_j vagy algebrai $K_{j-1} = K(x, \theta_1, \ldots, \theta_{j-1})$ felett, vagy $\theta_j = w_j$, ahol $w_j = \log(u_j)$ és $u_j \in K_{j-1}$, vagy $\theta_j = u_j$, ahol $u_j = exp(w_j)$ és $w_j \in K_{j-1}$. Ekkor*

1. *$g = \log(f)$, ahol $f \in K_n \setminus K$, pontosan akkor monomiális K_n felett, ha nincs olyan*

$$f^k \cdot \prod u_j^{k_j}, \quad k, k_j \in \mathbb{Z}, k \neq 0$$

   szorzat, amely K-beli;

2. *$g = exp(f)$, ahol $f \in K_n \setminus K$, pontosan akkor monomiális K_n felett, ha nincs olyan*

$$f + \sum c_j w_j, \quad c_j \in \mathbb{Q}$$

   lineáris kombináció, amely K-beli.

A szorzatképzés, illetve összegzés csak a logaritmikus és exponenciális lépésekre történik.

Az egész elmélet legfontosabb, klasszikus eredménye a következő tétel.

2.25. tétel (Liouville-elv). *Legyen K egy differenciáltest a C konstans testtel. Legyen L egy elemi differenciálkiterjesztése K-nak ugyanazzal a C konstans testtel. Tegyük fel, hogy $g' = f \in K$. Ekkor léteznek olyan $c_1, \ldots, c_m \in C$ konstansok és $v_0, v_1, \ldots, v_m \in K$ elemek, hogy*

$$f = v_0' + \sum_{j=1}^{m} c_j \frac{v_j'}{v_j} ,$$

azaz hogy

$$g = \int f = v_0 + \sum_{j=1}^{m} c_j \log(v_j) .$$

Figyeljük meg, hogy a helyzet hasonló, mint a racionális függvények integrálásánál.

A tételt nem bizonyítjuk. Bár a bizonyítás hosszadalmas, az ötlete könnyen elmondható. Először megmutatjuk, hogy egy transzcendens exponenciális bővítést differenciálással nem lehet „kiküszöbölni”, azaz egy racionális függvényét deriválva, abból az új elem nem tűnik el. Ez azon múlik, hogy a transzcendens exponenciális bővítő elem egy polinomját deriválva, a derivált újra az elem egy polinomja lesz, a polinom foka nem változik, és a derivált nem osztható az eredeti polinommal, kivéve, ha az monom volt. Utána megmutatjuk, hogy algebrai bővítésre nincs szükség az integrál kifejezéséhez. Ez lényegében azon múlik, hogy az algebrai bővítő elemet a minimálpolinomjába beírva nullát kapunk, és ezt az egyenletet differenciálva, a kapott egyenletből kifejezhető a bővítő elem deriváltja, mint az elem

racionális függvénye. Végül a transzcendens logaritmikus elemekkel való bővítéseket kell még megvizsgálnunk. Megmutatjuk, hogy egy ilyen bővítő elem differenciálással pontosan akkor küszöbölhető ki, ha egy konstans együtthatós elsőfokú polinomja szerepel. Ez azon múlik, hogy egy ilyen bővítő elem egy polinomját deriválva, annak egy polinomját kapjuk, aminek fokszáma vagy ugyanannyi, mint az eredetié, vagy eggyel kisebb, és ez utóbbi eset csak akkor fordulhat elő, ha a főegyüttható konstans.

Risch-algoritmus

Legyen K algebrai számtest $\mathbb{Q}$ felett, $K_n = K(x, \theta_1, \ldots, \theta_n)$ pedig transzcendens elemi függvénytest. Az algoritmus rekurzív n-ben: $\theta = \theta_n$ jelöléssel egy $f(\theta)/g(\theta) \in K_n = K_{n-1}(\theta)$ függvényt fogunk integrálni, ahol $K_{n-1} = K(x, \theta_1, \ldots, \theta_{n-1})$. (Az $n = 0$ eset a racionális függvények integrálása.) Feltehetjük, hogy f és g relatív prímek és g főpolinom. Az x szerinti differenciálás mellett használni fogjuk a θ szerinti deriválást is, ezt $d/d\theta$-val jelöljük. A továbbiakban csak az algoritmusokat ismertetjük.

Risch-algoritmus: logaritmikus eset

Az előző pont jelöléseivel, először feltesszük, hogy θ transzcendens és logaritmikus, $\theta' = u'/u$, $u \in K_{n-1}$. Maradékos osztással $f(\theta) = p(\theta)g(\theta) + h(\theta)$, ahonnan

$$\int \frac{f(\theta)}{g(\theta)} = \int p(\theta) + \int \frac{h(\theta)}{g(\theta)} \,.$$

A racionális függvények integrálásával ellentétben, most a polinomrész integrálása a nehezebb. Ezért a racionális rész integrálásával kezdjük.

Logaritmikus eset, racionális rész

Legyen $g(\theta) = g_1(\theta)g_2^2(\theta)\cdots g_m^m(\theta)$ a $g(\theta)$ négyzetmentes felbontása. Ekkor

$$\mathrm{lnko}\left((g_j(\theta), \frac{d}{d\theta}g_j(\theta)\right)) = 1$$

nyilván teljesül. Megmutatható, hogy a jóval erősebb $\mathrm{lnko}(g_j(\theta), g_j(\theta)') = 1$ feltétel is teljesül. Parciális törtekre bontással

$$\frac{h(\theta)}{g(\theta)} = \sum_{i=1}^{m}\sum_{j=1}^{i}\frac{h_{i,j}(\theta)}{g_i(\theta)^j} \,.$$

Hermite-redukciót fogunk használni: a bővített euklidészi algoritmussal kaphatunk olyan $s(\theta), t(\theta) \in K_{n-1}[\theta]$ polinomokat, amelyekre $s(\theta)g_i(\theta) + t(\theta)g_i(\theta)' = h_{i,j}(\theta)$ és $\deg s(\theta), \deg t(\theta) < \deg g_i(\theta)$. Innen parciális integrálással

$$\begin{aligned}
\int \frac{h_{i,j}(\theta)}{g_i^j(\theta)} &= \int \frac{t(\theta)\cdot g_i(\theta)'}{g_i(\theta)^j} + \int \frac{s(\theta)}{g_i(\theta)^{j-1}} \\
&= \frac{-t(\theta)}{(j-1)g_i(\theta)^{j-1}} + \int \frac{t(\theta)'}{(j-1)g_i(\theta)^{j-1}} + \int \frac{s(\theta)}{g_i(\theta)^{j-1}} \\
&= \frac{-t(\theta)}{(j-1)g_i(\theta)^{j-1}} + \int \frac{s(\theta) + t(\theta)'/(j-1)}{g_i(\theta)^{j-1}} \,.
\end{aligned}$$

Addig alkalmazva ezt az eljárást, amíg $j > 1$, azt kapjuk, hogy

$$\int \frac{h(\theta)}{g(\theta)} = \frac{c(\theta)}{d(\theta)} + \int \frac{a(\theta)}{b(\theta)} ,$$

ahol $a(\theta), b(\theta), c(\theta), d(\theta) \in K_{n-1}[\theta]$, $\deg a(\theta) < \deg b(\theta)$ és $b(\theta)$ négyzetmentes főpolinom.

Megmutatható, hogy az $\int a(\theta)/b(\theta)$ integrál kiszámítására a Rothstein–Trager-módszer alkalmazható. Számítsuk ki az

$$r(y) = \mathrm{res}_\theta(b(\theta), a(\theta) - y \cdot b(\theta)')$$

rezultánst. Megmutatható, hogy az integrál pontosan akkor elemi, ha $r(y) = r(y)s$ alakban írható, ahol $r(y) \in K[y]$ és $s \in K_{n-1}$. Ha tehát kiszámítjuk $r(y)$ primitív részét, ezt választjuk $r(y)$-nak, és $r(y)$ bármelyik együtthatója nem konstans, akkor nem létezik elemi integrál. Egyébként legyenek $c_1, \ldots, c_k$ az $r(y)$ különböző gyökei annak felbontási testében és legyen

$$v_i(\theta) = \mathrm{lnko}(b(\theta), a(\theta) - c_i b(\theta)') \in K_{n-1}(c_1, \ldots, c_k)[\theta] ,$$

ha $i = 1, \ldots, k$. Megmutatható, hogy

$$\int \frac{a(\theta)}{b(\theta)} = \sum_{i=1}^{k} c_i \log(v_i(\theta)) .$$

Tekintsünk néhány példát.

2.24. példa. Az $\int 1/\log(x)$ integrál integrandusa $1/\theta \in \mathbb{Q}(x, \theta)$, ahol $\theta = \log(x)$. Mivel

$$r(y) = \mathrm{res}_\theta(\theta, 1 - y/x) = 1 - y/x \in \mathbb{Q}(x)[y]$$

primitív polinom és van nem konstans együtthatója, az integrál nem elemi.

2.25. példa. Az $\int 1/(x\log(x))$ integrál integrandusa $1/(x\theta) \in \mathbb{Q}(x, \theta)$, ahol $\theta = \log(x)$. Itt

$$r(y) = \mathrm{res}_\theta(\theta, 1/x - y/x) = 1/x - y/x \in \mathbb{Q}(x)[y],$$

aminek primitív része $1 - y$. Ennek minden együtthatója konstans, így az integrál elemi, $c_1 = 1$, $v_1(\theta) = \mathrm{lnko}(\theta, 1/x - 1/x) = \theta$, így

$$\int \frac{1}{x\log(x)} = c_1 \log(v_1(\theta)) = \log(\log(x)) .$$

Logaritmikus eset, polinom rész

Marad a

$$p(\theta) = p_k\theta^k + p_{k-1}\theta^{k-1} + \cdots + p_0 \in K_{n-1}[\theta]$$

polinomrész integrálásának problémája. A Liouville-elv szerint $\int p(\theta)$ pontosan akkor elemi, ha

$$p(\theta) = v_0(\theta)' + \sum_{j=1}^{k} c_j \frac{v_j(\theta)'}{v_j(\theta)} ,$$

ahol $c_j \in K$ és $v_i \in K_{n-1}(\theta)$, ha $j = 0, 1, \ldots, m$, továbbá $K_{\mathbb{C}}$ a K valamely bővítése és $K_{n-1} = K(x, \theta_1, \ldots, \theta_{n-1})$. Meg fogjuk mutatni, hogy K lehet a K egy algebrai bővítése. Hasonlóan érvelve, mint a Liouville-elv bizonyításában, megmutatható, hogy $v_0(\theta) \in K_{n-1}[\theta]$ és $v_j(\theta) \in K_{n-1}$ (azaz független θ-tól), ha $j = 1, 2, \ldots, m$. Így

$$p(\theta) = v_0(\theta)' + \sum_{j=1}^{m} c_j \frac{v_j'}{v_j} .$$

A Liouville-elv bizonyításánál használt érveléssel azt is megkapjuk, hogy $v_0(\theta)$ foka legfeljebb $k+1$. így ha $v_0(\theta) = q_{k+1}\theta^{k+1} + q_k\theta^k + \cdots + q_0$, akkor

$$p_k\theta^k + p_{k-1}\theta^{k-1} + \cdots + p_0 = (q_{k+1}\theta^{k+1} + q_k\theta^k + \cdots + q_0)' + \sum_{j=1}^{m} c_j \frac{v_j'}{v_j} .$$

Innen a következő egyenletrendszert kapjuk:

$$\begin{aligned} 0 &= q_{k+1}' , \\ p_k &= (k+1)q_{k+1}\theta' + q_k' , \\ p_{k-1} &= kq_k\theta' + q_{k-1}' , \\ &\vdots \\ p_1 &= 2q_2\theta' + q_1' , \\ p_0 &= q_1\theta' + q_0' , \end{aligned}$$

ahol az utolsó egyenletben $q_0 = q_0 + \sum_{j=1}^{m} c_j \log(v_j)$. Az első egyenlet megoldása egyszerűen egy b_{k+1} konstans. Ezt visszahelyettesítve a következő egyenletbe és mindkét oldalt integrálva azt kapjuk, hogy

$$\int p_k = (k+1)b_{k+1} \cdot \theta + q_k .$$

Az integrálási eljárást rekurzív módon alkalmazva $p_k \in K_{n-1}$ integrálja kiszámítható, azonban ez az egyenlet csak akkor oldható meg, ha az integrál elemi, és legfeljebb egy logaritmikus bővítést használ és az éppen $\theta = \log(u)$. Ha ez nem teljesül, akkor $\int p(\theta)$ nem lehet elemi. Ha ez teljesül, akkor $\int p_k = c_k\theta + d_k$ valamely $c_k \in K$ és $d_k \in K_{n-1}$-el, ahonnan $b_{k+1} = c_{k+1}/(k+1) \in K$ és $q_k = d_k + b_k$ egy tetszőleges b_k integrációs konstanssal. Behelyettesítve q_k-t a következő egyenletbe és átrendezve

$$p_{k-1} - kd_k\theta' = kb_k\theta' + q_{k-1}' ,$$

vagyis mindkét oldalt integrálva

$$\int \left(p_{k-1} - kd_k\frac{u'}{u}\right) = kb_k\theta + q_{k-1}$$

adódik. A jobb oldalon az integrandus K_{n-1}-beli, így az integrációs eljárást rekurzív módon hívhatjuk. Ugyanúgy mint fent, az egyenlet csak akkor oldható meg, ha az integrál elemi, és legfeljebb egy logaritmikus bővítést használ és az éppen $\theta = \log(u)$. Tegyük fel, hogy ez

teljesül és

$$\int \left(p_{k-1} - k d_k \frac{u'}{u}\right) = c_{k-1}\theta + d_{k-1} \ ,$$

ahol $c_{k-1} \in K$ és $d_{k-1} \in K_{n-1}$. Ekkor a keresett megoldás $b_k = c_{k-1}/k \in K$ és $q_{k-1} = d_{k-1} + b_{k-1}$, ahol b_{k-1} egy tetszőleges integrációs konstans. Az eljárást folytatva, az utolsó előtti egyenlet megoldása $b_2 = c_1/2 \in K$ és $q_1 = d_1 + b_1$ valamely b_1 integrációs konstanssal. Behelyettesítve q_1-et az utolsó egyenletbe, átrendezve, majd integrálva

$$\int \left(p_0 - d_1 \frac{u'}{u}\right) = b_1\theta + q_0 \ .$$

Ez alkalommal csak az a feltétel, hogy az integrál elemi függvény legyen. Ha elemi függvény, mondjuk

$$\int \left(p_0 - d_1 \frac{u'}{u}\right) = d_0 \in K_{n-1} \ ,$$

akkor $b_1 \in K$ a $\theta = \log(u)$ együtthatója d_0-ban és $q_0 = d_0 - b_1 \log(u)$, az eredmény pedig

$$\int p(\theta) = b_{k+1}\theta^{k+1} + q_k\theta^k + \cdots + q_1\theta + q_0 \ .$$

Nézzünk néhány példát.

2.26. példa. Az $\int \log(x)$ integrál integrandusa $\theta \in \mathbb{Q}(x, \theta)$, ahol $\theta = \log(x)$. Ha az integrál elemi, akkor

$$\int \theta = b_2\theta^2 + q_1\theta + q_0$$

és $0 = b_2'$, $1 = 2b_2\theta' + q_1'$, $0 = q_1\theta' + q_0'$. Az ismeretlen b_2 konstanssal a második egyenletből $\int 1 = 2b_2\theta + q_1$. Mivel $\int 1 = x + b_1$, azt kapjuk, hogy $b_2 = 0$, $q_1 = x + b_1$. A harmadik egyenletből $-x\theta' = b_1\theta' + q_0'$. Mivel $\theta' = 1/x$, integrálva $\int -1 = b_1\theta + q_0$, és $\int -1 = -x$, azt kapjuk, hogy $b_1 = 0$, $q_0 = -x$, így $\int \log(x) = x\log(x) - x$.

2.27. példa. Az $\int \log(\log(x))$ integrál integrandusa $\theta_2 \in \mathbb{Q}(x, \theta_1, \theta_2)$, ahol $\theta_1 = \log(x)$ és $\theta_2 = \log(\theta_1)$. Ha az integrál elemi, akkor

$$\int \theta_2 = b_2\theta_2^2 + q_1\theta_2 + q_0$$

és $0 = b_2'$, $1 = 2b_2\theta_2' + q_1'$, $0 = q_1\theta_2' + q_0'$. Az ismeretlen b_2 konstanssal a második egyenletből $\int 1 = 2b_2\theta + q_1$. Mivel $\int 1 = x + b_1$, azt kapjuk, hogy $b_2 = 0$, $q_1 = x + b_1$. A harmadik egyenletből $-x\theta_2' = b_1\theta_2' + q_0'$. Mivel $\theta_2' = \theta_1'/\theta_1 = 1/(x\log(x))$, teljesülnie kell az

$$\int \frac{-1}{\log(x)} = b_1\theta_2 + q_0$$

egyenlőségnek, azonban a 2.24. példából tudjuk, hogy a bal oldalon álló integrál nem elemi.

Risch-algoritmus: exponenciális eset

Most feltesszük, hogy θ transzcendens és exponenciális, $\theta'/\theta = u'$, $u \in K_{n-1}$. Maradékos osztással $f(\theta) = q(\theta)g(\theta) + h(\theta)$, ahonnan

$$\int \frac{f(\theta)}{g(\theta)} = \int q(\theta) + \int \frac{h(\theta)}{g(\theta)} \ .$$

Tervünk az, hogy a racionális részre Hermite módszerét fogjuk alkalmazni. Kellemetlen meglepetés ér azonban bennünket, mert a négyzetmentes felbontásban szereplő $g_j(\theta)$ függvényekre bár

$$\mathrm{lnko}\left(g_j(\theta), \frac{d}{d\theta}g_j(\theta)\right) = 1$$

nyilván teljesül, a jóval erősebb $\mathrm{lnko}(g_j(\theta), g_j(\theta)') = 1$ feltétel már nem. Például ha $g_j(\theta) = \theta$, akkor

$$\mathrm{lnko}(g_j(\theta), g_j(\theta)') = \mathrm{lnko}(\theta, u'\theta) = \theta\ .$$

Megmutatható azonban, hogy ez a kellemetlen jelenség nem lép fel, ha $\theta \nmid g_j(\theta)$, ekkor már $\mathrm{lnko}(g_j(\theta), g_j(\theta)') = 1$. Elég tehát, ha a θ tényezőt eltávolítjuk a nevezőből. Legyen $g(\theta) = \theta^\ell g(\theta)$, ahol már $\theta \nmid g(\theta)$, és keressünk olyan $h(\theta), s(\theta) \in K_{n-1}[\theta]$ polinomokat, amelyekre $h(\theta)\theta^\ell + t(\theta)g(\theta) = h(\theta)$, $\deg h(\theta) < \deg g(\theta)$ és $\deg s(\theta) < \ell$. Mindkét oldalt osztva $g(\theta)$-val azt kapjuk, hogy

$$\frac{f(\theta(}{g(\theta)} = q(\theta) + \frac{t(\theta)}{\theta^l} + \frac{h(\theta)}{g(\theta)}\ .$$

A $p(\theta) = q(\theta) + t(\theta)/\theta^l$ jelöléssel $p(\theta)$ véges Laurent-sor, ennek integrálása azonban semmivel sem lesz nehezebb, mint egy polinom integrálása. Ez nem meglepő, ha meggondoljuk, hogy $\theta^{-1} = \exp(-u)$. Még így is, itt is a „polinomrész” integrálása a nehezebb. A másikkal kezdjük.

Exponenciális eset, racionális rész

Legyen $g(\theta) = g_1(\theta)g_2^2(\theta)\cdots g_m^m(\theta)$ a $g(\theta)$ négyzetmentes felbontása. Ekkor $\theta \nmid g_j(\theta)$ miatt $\mathrm{lnko}(g_j(\theta), g_j(\theta)') = 1$. Parciális törtekre bontással

$$\frac{h(\theta)}{g(\theta)} = \sum_{i=1}^{m}\sum_{j=1}^{i}\frac{h_{i,j}(\theta)}{g_i(\theta)^j}\ .$$

A Hermite-redukció ugyanúgy megy, mint a logaritmikus résznél. Azt kapjuk, hogy

$$\int \frac{h(\theta)}{g(\theta)} = \frac{c(\theta)}{d(\theta)} + \int \frac{a(\theta)}{b(\theta)}\ ,$$

ahol $a(\theta), b(\theta), c(\theta), d(\theta) \in K_{n-1}[\theta]$, $\deg a(\theta) < \deg b(\theta)$ és $b(\theta)$ négyzetmentes főpolinom, $\theta \nmid b(\theta)$.

Megmutatható, hogy az $\int a(\theta)/b(\theta)$ integrál kiszámítására a Rothstein–Trager-módszer alkalmazható. Számítsuk ki a

$$r(y) = \mathrm{res}_\theta(b(\theta), a(\theta) - y\cdot b(\theta)')$$

rezultánst. Megmutatható, hogy az integrál pontosan akkor elemi, ha $r(y) = r(y)s$ alakban írható, ahol $r(y) \in K[y]$ és $s \in K_{n-1}$. Ha tehát kiszámítjuk $r(y)$ primitív részét, ezt választjuk $r(y)$-nak, és $r(y)$ bármelyik együtthatója nem konstans, akkor nem létezik elemi integrál. Egyébként legyenek $c_1, \ldots, c_k$ az $r(y)$ különböző gyökei annak felbontási testében és legyen

$$v_i(\theta) = \mathrm{lnko}(b(\theta), a(\theta) - c_i b(\theta)') \in K_{n-1}(c_1, \ldots, c_k)[\theta]\ ,$$

ha $i = 1, \ldots, k$. Megmutatható, hogy

$$\int \frac{a(\theta)}{b(\theta)} = -\left(\left(\sum_{i=1}^{k} c_i \deg v_i(\theta) \right) \right) + \sum_{i=1}^{k} c_i \log(v_i(\theta)) \ .$$

Nézzünk néhány példát.

2.28. példa. Az $\int 1/(1+\exp(x))$ integrál integrandusa $1/(1+\theta) \in \mathbb{Q}(x,\theta)$, ahol $\theta = \exp(x)$. Mivel

$$r(y) = \mathrm{res}_\theta(\theta + 1, 1 - y\theta) = -1 - y \in \mathbb{Q}(x)[y]$$

primitív polinom és csak konstans együtthatói vannak, az integrál elemi, $c_1 = -1$, $v_1(\theta) = \mathrm{lnko}(\theta + 1, 1 + \theta) = 1 + \theta$, így

$$\int \frac{1}{1+\exp(x)} = -c_1 x \deg v_1(\theta) + c_1 \log(v_1(\theta)) = x - \log(\exp(x) + 1) \ .$$

2.29. példa. Az $\int x/(1+\exp(x))$ integrál integrandusa $x/(1+\theta) \in \mathbb{Q}(x,\theta)$, ahol $\theta = \exp(x)$. Mivel

$$r(y) = \mathrm{res}_\theta(\theta + 1, x - y\theta) = -x - y \in \mathbb{Q}(x)[y]$$

primitív polinom, amelynek van nem konstans együtthatója, az integrál nem elemi.

Exponenciális eset, polinom rész

Marad a

$$p(\theta) = \sum_{i=-\ell}^{k} p_i \theta^i \in K_{n-1}(\theta)$$

„polinomrész” integrálásának problémája. A Liouville-elv szerint $\int p(\theta)$ pontosan akkor elemi, ha

$$p(\theta) = v_0(\theta)' + \sum_{j=1}^{m} c_j \frac{v_j(\theta)'}{v_j(\theta)} \ ,$$

ahol $c_j \in K$ és $v_j \in K_{n-1}(\theta)$, ha $j = 0, 1, \ldots, m$, továbbá $K_{\mathbb{C}}$ a K valamely bővítése és $K_{n-1} = K(x, \theta_1, \ldots, \theta_{n-1})$. Megmutatható, hogy K lehet a K egy algebrai bővítése. Hasonlóan érvelve, mint a Liouville-elv bizonyításában, megmutatható, hogy az általánosság megszorítása nélkül feltehetjük: $v_j(\theta)$ vagy K_{n-1} eleme (azaz független θ-tól) vagy pedig főpolinom és irreducibilis $K_{n-1}[\theta]$-ban, ha $j = 1, 2, \ldots, m$. Továbbá az is belátható, hogy $v_0(\theta)$ nevezőjében nem lehet nem monom tényező, mivel egy ilyen tényező megmaradna a derivált nevezőjében is. Hasonlóan $v_j(\theta)$-nak ($j = 1, 2, \ldots, m$) sem lehet nem monom tényezője. Így azt kapjuk, hogy $v_j(\theta)$ vagy K_{n-1} eleme, vagy pedig $v_j(\theta) = \theta$, mivel ez az egyetlen irreducibilis monom, amely főpolinom. Ha azonban $v_j(\theta) = \theta$, akkor a megfelelő tag az összegben $c_j v_j(\theta)'/v_j(\theta) = c_j u'$, ami beolvasztható $v_0(\theta)'$-be. Ebből azt kapjuk, hogy ha $p(\theta)$-nak van elemi integrálja, akkor

$$p(\theta) = \left(\left(\sum_{j=-\ell}^{k} q_j \theta^j \right) \right)' + \sum_{j=1}^{m} c_j \frac{v_j'}{v_j} \ ,$$

ahol $q_j, v_j \in K_{n-1}$ és $c_j \in K$; az, hogy az összegzési határok az első összegben meg kell egyezzenek a $p(\theta)$ előállításában szereplő összegzési határokkal, következik abból, hogy

$$(q_j\theta^j)' = (q_j' + ju'g_j)\theta^j \ .$$

Az együtthatók összehasonlításával a

$$\begin{array}{rcll} p_j & = & q_j' + ju'q_j, & \text{ha } -\ell \le j \le k,\ j \neq 0\ , \\ p_0 & = & q_0' \ , & \end{array}$$

egyenletrendszert kapjuk, ahol $q_0 = q_0 + \sum_{j=1}^{m} c_j \log(v_j)$. A $p_0 = q_0'$ egyenlet megoldása egyszerűen $q_0 = \int p_0$; ha ez az integrál nem elemi, akkor $\int p(\theta)$ sem elemi, ha viszont elemi, akkor meghatároztuk q_0-at. A $j \neq 0$ esetben egy differenciálegyenletet kell megoldanunk q_j meghatározásához, az úgynevezett ***Risch-differenciálegyenletet.*** A differenciálegyenlet $y' + fy = g$ alakú, ahol az adott f, g függvények K_{n-1} elemei és K_{n-1}-beli megoldásokat keresünk. Első pillantásra úgy tűnik, hogy az integrálás problémáját egy nehezebb problémával helyettesítettük, azonban az, hogy az egyenletek lineárisak, a megoldásnak pedig K_{n-1}-ben kell lenni, sokat segít. Ha valamelyik Risch-differenciálegyenletnek nincs K_{n-1}-beli megoldása, akkor $\int p(\theta)$ nem elemi, egyébként

$$\int p(\theta) = \sum_{j\neq 0} q_j\theta^j + q_0 \ .$$

A Risch-differenciálegyenlet algoritmussal megoldható, ezt nem részletezzük.

Tekintsünk néhány példát.

2.30. példa. Az $\int \exp(-x^2)$ integrál integrandusa $\theta \in \mathbb{Q}(x,\theta)$, ahol $\theta = \exp(-x^2)$. Ha az integrál elemi, akkor $\int \theta = q_1\theta$, ahol $q_1 \in \mathbb{C}(x)$. Nem nehéz belátni, hogy a differenciálegyenletnek nincs racionális megoldása, így $\int \exp(-x^2)$ nem elemi.

2.31. példa. Az $\int x^x$ integrál integrandusa $\exp((x\log(x)) = \theta_2 \in \mathbb{Q}(x,\theta_1,\theta_2)$, ahol $\theta_1 = \log(x)$ és $\theta_2 = \exp(x\theta_1)$. Ha az integrál elemi, akkor $\int \theta_2 = q_1\theta_2$, ahol $q_1 \in \mathbb{C}(x,\theta_1)$. Mindkét oldalt differenciálva $\theta_2 = q_1'\theta_2 + q_1(\theta_1+1)\theta_2$, ahonnan $1 = q_1' + (\theta_1+1)q_1$. Mivel θ_1 transzcendens $\mathbb{C}(x)$ felett, az együtthatók összehasonlításával $1 = q_1' + q_1$ és $0 = q_1$, aminek nincs megoldása. így $\int x^x$ nem elemi.

2.32. példa. Az

$$\int \frac{(4x^2+4x-1)(\exp(x^2)+1)(\exp(x^2)-1)}{(x+1)^2}$$

integrál integrandusa

$$f(\theta) = \frac{4x^2+4x-1}{(x+1)^2}(\theta^2-1) \in \mathbb{Q}(x,\theta)\ ,$$

ahol $\theta = \exp(x^2)$. Ha az integrál elemi, akkor $\int f(\theta) = q_2\theta^2 + q_0$ alakú, ahol

$$q_2' + 4xq_2 \quad = \quad \frac{4x^2+4x-1}{(x+1)^2}\ ,$$

$$q_0' = -\frac{4x^2+4x-1}{(x+1)^2} .$$

A második egyenlet integrálható, és q_0 elemi. Az első egyenletnek megoldása $q_2 = 1/(1+x)$. Innen

$$\int f(\theta) = \frac{1}{x+1}\exp^2(x^2) - \frac{(2x+1)^2}{x+1} + 4\log(x+1) .$$

Gyakorlatok

2.4-1. Alkalmazzuk a Hermite-redukciót az alábbi $f(x) \in \mathbb{Q}(x)$ függvényre:

$$f(x) = \frac{441x^7 + 780x^6 - 286x^5 + 4085x^4 + 769x^3 + 3713x^2 - 43253x + 24500}{9x^6 + 6x^5 - 65x^4 + 20x^3 + 135x^2 - 154x + 49} .$$

2.4-2. Számítsuk ki az $\int f$ integrált, ahol

$$f(x) = \frac{36x^6 + 126x^5 + 183x^4 + 13807/6x^3 - 407x^2 - 3242/5x + 3044/15}{(x^2 + 7/6x + 1/3)^2(x - 2/5)^3} \in \mathbb{Q}(x) .$$

2.4-3. Alkalmazzuk a Risch-algoritmust az alábbi integrál kiszámítására:

$$\int \frac{x(x+1)\{(x^2e^{2x^2} - \log(x+1)^2) + 2xe^{3x^2}(x - (2x^3+2x^2+x+1)\log(x+1))\}}{((x+1)\log^2(x+1) - (x^3+x^2)e^{2x^2})^2} dx .$$

2.5. Elmélet és gyakorlat

A fejezet eddigi részében arra törekedtünk, hogy néhány lényeges szimbolikus algoritmus bemutatásán keresztül érzékeltessük a komputeralgebra tudományterületének algoritmustervezési problémáit. A következőkben az érdeklődő Olvasó áttekintést kaphat a szimbolikus algoritmusok kutatásának szélesebb világából.

2.5.1. Egyéb szimbolikus algoritmusok

A fejezetben bemutatott rezultánsmódszer és a Gröbner-bázis elmélet mellett létezik algoritmus nemlineáris egyenletek és egyenlőtlenségek *valós* szimbolikus gyökeinek meghatározására is (Collins).

Figyelemre méltó algoritmusok születtek differenciálegyenletek szimbolikus megoldásainak vizsgálata során. A Risch-algoritmushoz hasonló döntési eljárás létezik racionálisfüggvény-együtthatójú homogén másodrendű közönséges differenciálegyenletek zárt alakú megoldásainak kiszámítására. Magasabb rendű lineáris esetben az Abramov-eljárás a polinomegyüttható egyenletek zárt racionális megoldásait, a Bronstein-algoritmus pedig az $\exp(\int f(x)dx)$ alakú megoldásokat határozza meg. Parciális differenciálegyenletek esetében a Lie-féle szimmetria-módszerek állnak rendelkezésre. Létezik algoritmus formális hatványsorok és racionális függvények feletti lineáris differenciáloperátorok faktorizálására is.

A faktorizáláson alapuló eljárások komoly jelentőséggel bírnak a komputeralgebrai algoritmusok kutatásában. Olyannyira igaz ez a megállapítás, hogy sokan az egész tudományterület születését Berlekamp azon publikációjától számítják, amelyben hatékony algoritmust ad kis p karakterisztikájú véges testek feletti egyhatározatlanú polinomok faktorizációjára. Berlekamp eredményét később nagyobb p karakterisztikákra is kiterjesztette. Azért, hogy hasonlóan jó futási időt kapjon, az algoritmusába véletlen elemeket illesztett. A mai komputeralgebra rendszerek nagyobb véges testekre is rutinszerűen alkalmazzák Berlekamp eljárását talán anélkül, hogy a felhasználók többségének az algoritmus valószínűségi eredetéről tudomása lenne. A módszer könyvünk második kötetében kerül ismertetésre. Megjegyezzük, hogy véges testek feletti polinomok faktorizálására számos algoritmus létezik.

Nem sokkal azután, hogy a véges testek feletti polinomfaktorizáció megoldódott, Zassenhaus van der Waerden 1936-os *Moderne Algebra* könyvét alapul véve a p-adikus számok aritmetikájának ún. Hensel-lemmáját alkalmazta a faktorok kiterjesztésére. A „Hensel-lifting" – ahogy ma az eljárását nevezik – általános megközelítési módszer arra, hogyan rekonstruáljuk a faktorokat moduláris képeikből. Az interpolációval ellentétben, ami több képpont meglétét követeli meg, a Hensel-lifting kiindulásul csak egyetlen képpontot igényel. Az egész együtthatós polinomok faktorizációjára adott Berlekamp–Zassenhaus-féle algoritmus alapvető jelentőségű, mégis rejteget magában két csapdát. Az egyik, hogy bizonyos fajta polinomokra az algoritmus futási ideje exponenciális. Sajnos, az algebrai számtestekre épített polinomok faktorizációjánál sok ilyen „rossz" polinom kerül elő. A második, hogy többváltozós polinomokra hasonló reprezentációs probléma lép fel, mint amilyet ritka mátrixok Gauss-eliminációjánál tapasztalunk. Az első problémát egy, a számok geometriáján alapuló diofantoszi optimalizálás, a Lenstra–Lenstra–Lovász nevével fémjelzett ún. rácsredukciós algoritmus oldotta meg, amit a Berlekamp-módszerrel együtt használnak. Ezen polinomiális algoritmus mellé még egy olyan eljárás társul, amely arról gondoskodik, hogy a Hensel-lifting „jó" moduláris képről induljon és „megfelelő időben" véget érjen. A többváltozós polinomfaktorizáció említett reprezentációs problémájára is születtek megoldások. Ez a második olyan terület, ahol a véletlenítés kritikus szerepet kap a hatékony algoritmusok tervezésében. Megjegyezzük, hogy a gyakorlatban a Berlekamp–Zassenhaus–Hensel-féle algoritmus hatékonyabban működik, mint a Lenstra–Lenstra–Lovász-féle eljárás. Összevetésképpen a polinomfaktorizálás problémája tehát polinomiális időben megoldható, míg az N egész faktorizálására a bizonyítottan legjobb algoritmikus korlát $\widetilde{O}(N^{1/4})$ (Pollard és Strassen) a determinisztikus, és $L(N)^{1+o(1)}$ (Lenstra és Pomerance) a valószínűségi esetben, ahol $L(N) = e^{\sqrt{\ln N \ln\ln N}}$.

Valójában a heurisztikus, valószínűségi módszereknek egy új elmélete van születőben a komputeralgebrában egyrészt az számítási tárrobbanás elkerülésére, másrészt a determinisztikusan nagy futási idejű algoritmusok hatékonyságának növelésére. A valószínűségi algoritmusok esetében a nem megfelelő működésnek is van pozitív valószínűsége, ami vagy az esetleges rossz válaszban nyilvánul meg (Monte Carlo csoport), vagy, habár sohasem kapunk rossz választ (Las Vegas csoport), elképzelhető, hogy polinomiális időben nem kapunk semmit. A már említetteken kívül heurisztikákkal szép eredményeket értek el többek között polinomazonosság tesztelésénél, polinomok irreducibilitásának vizsgálatánál, mátrixok normálformáinak (Frobenius, Hilbert, Smith) meghatározásánál. Szerepük minden valószínűség szerint a jövőben is növekedni fog.

A fejezet eddigi részében a lényegesebb szimbolikus algoritmusokat tekintettük át. Már a bevezetőben említettük, hogy a komputeralgebra-rendszerek többsége képes numerikus

számítások elvégzésére is: a hagyományostól eltérően a számítási pontosságot a felhasználó határozhatja meg. Gyakran előfordul, hogy a szimbolikus és numerikus számításokat együtt célszerű használni. Tekintsük például egy differenciálegyenlet szimbolikusan kiszámolt hatványsor megoldását. A csonkított hatványsort ezután bizonyos pontokban a szokásos lebegőpontos aritmetikával kiértékelve a differenciálegyenlet megoldásának numerikus approximációját kapjuk. Amennyiben a megoldandó probléma valamilyen valós fizikai probléma approximációja, a szimbolikus számítások vonzó lehetőségei gyakran érvényüket vesztik; egyszerűen mert túl bonyolultak vagy túl lassúak ahhoz, hogy hasznosak vagy szükségesek legyenek, hiszen a megoldást is numerikusan keressük. Más esetekben, ha a probléma szimbolikusan kezelhetetlen, az egyetlen lehetséges út a numerikus approximációs módszerek használata. Ilyen akkor fordulhat elő, ha a létező szimbolikus algoritmus nem talál zárt megoldást (pl. nem elemi függvények integrálja stb.), vagy nem létezik a problémát kezelni tudó szimbolikus algoritmus. Ámbár egyre több numerikus algoritmusnak létezik szimbolikus megfelelője, a numerikus eljárások nagyon fontosak a komputeralgebrában. Gondoljunk csak a differenciál- és integrálszámítás területére: bizonyos esetekben a hagyományos algoritmusok – integráltranszformációk, hatványsor-approximációk, perturbációs módszerek – lehetnek a legcélravezetőbbek.

A komputeralgebra algoritmusok tervezésénél a jövőben egyre nagyobb szerepet kapnak a párhuzamos architektúrájú számítógépek. Habár sok meglévő algoritmus „ránézésre” párhuzamosítható, nem biztos, hogy a jó szekvenciális algoritmusok párhuzamos architektúrákon is a legjobbak lesznek: az optimum talán egy teljesen különböző eljárással érhető el.

2.5.2. A komputeralgebra-rendszerek áttekintése

A komputeralgebra-rendszerek fejlődésének története szoros kapcsolatban áll az informatika és az algoritmikus matematika fejlődésével. A számítástechnika kezdeti időszakában a különböző tudományterületek művelői szimbolikus számítási igényeik megkönnyítése és felgyorsítása érdekében kezdték el fejleszteni az első komputeralgebra-rendszereket, amelyek átdolgozva, folyamatosan megújulva és a sokadik változatban napjainkban is jelen vannak. A hetvenes években jelentek meg az ***általános célú*** komputeralgebra-rendszerek, amelyeket beépített adatstruktúrák, matematikai függvények és algoritmusok széles tárháza jellemez, minél nagyobb felhasználói területet próbálva meg lefedni. A jelentős számítógépes erőforrásigény miatt robbanásszerű elterjedésük a nyolcvanas évek elejére, a mikroprocesszor alapú munkaállomások megjelenésének idejére tehető. A jobb hardver környezet, a hatékonyabb erőforrás-gazdálkodás, rendszerfüggetlen, magasszintű alapnyelv használata, és nem utolsósorban a társadalmi-gazdasági igények fokozatosan piaci termékké érlelték az általános célú komputeralgebra-rendszereket, ami viszont erős javulást hozott a felhasználói felület és dokumentumkészítés terén.

Az alábbiakban a legismertebb általános és speciális célú komputeralgebra rendszereket, könyvtárakat soroljuk fel.

- Általános célú komputeralgebra rendszerek: Axiom, Derive, Form, GNU-calc, Jacal, Macsyma, Maxima, Maple, Distributed Maple, MathCAD, Matlab Symbolic Math Toolbox, Scilab, MAS, Mathematica, Mathview, Mock-Mma, MuPAD, Reduce, Risa.

- Algebra és számelmélet: Bergman, CoCoA, Felix, Fermat, GRB, Kan, Macaulay, Magma, Numbers, Pari, Simath, Singular.
- Algebrai geometria: Casa, Ganith.
- Csoportelmélet: Gap, LiE, Magma, Schur.
- Tenzor analízis: Cartan, FeynCalc, GRG, GRTensor, MathTensor, RedTen, Ricci, TTC.
- Komputeralgebra könyvtárak: Apfloat, BigNum, GNU MP, Kant, LiDiA, NTL, Saclib, Ubasic, Weyl, Zen.

Az általános célú komputeralgebra rendszerek többsége olyan tulajdonságokkal bír, mint

- interaktivitás,
- matematikai tények ismerete,
- matematikai objektumok kezelésére képes deklaratív[2], magas szintű programozási nyelv funkcionális programozási lehetőségekkel,
- az operációs rendszer és más programok felé való kiterjeszthetőség,
- szimbolikus és numerikus számítások integrálása,
- automatikus (optimalizált) C és Fortran kódszegmens generálása,
- grafikus felhasználói környezet,
- 2 és 3 dimenziós grafika, animáció,
- szövegszerkesztési lehetőség és automatikus LaTeX konverzió,
- on-line súgó.

A komputeralgebra-rendszereket ***matematikai szakértői rendszereknek*** is nevezik. Napjainkban az általános célú komputeralgebra-rendszerek szédítő iramú fejlődésének lehetünk szemtanúi, elsősorban tudásuknak és széles körű alkalmazhatóságuknak köszönhetően. Mégis hiba volna alábecsülni a speciális rendszereket, amelyek egyrészt igen fontos szerepet játszanak sok tudományterületen, másrészt sok esetben könnyebben kezelhetők és hatékonyabbak, jelölésrendszerük és algoritmusaik alacsony szintű programnyelvi implementációja miatt. Nagyon lényeges, hogy egy adott probléma megoldásához az arra legalkalmasabb komputeralgebra-rendszert válasszuk ki.

Feladatok

2-1. Maradékos osztás maradéksorozata együtthatói hosszának vizsgálata
Generáljunk két ($n = 10$)-ed fokú pszeudovéletlen polinomot $\mathbb{Z}[x]$-ben $l = 10$ decimális jegyből álló együtthatókkal. Hajtsunk végre egyetlen maradékos osztást ($\mathbb{Q}[x]$-ben) és számítsuk ki a maradék polinom és az eredeti polinom (λ függvénnyel meghatározott) maximális együtthatóhosszának arányát. Ismételjük meg a számításokat ($t = 20$)-szor és számítsunk átlagot. Mennyi lesz a kapott érték? Ismételjük meg a fenti kísérletet $l = 100, 500, 1000$ esetén.

[2] A deklaratív programozási nyelvek a kívánt eredményt specifikálják és nem a hozzájuk vezető utat, mint ahogy az imperatív nyelvek teszik.

2-2.* Moduláris-lnko-kisprímek *algoritmus szimulációs vizsgálata
Szimulációs vizsgálattal adjunk becslést a Moduláris-lnko-kisprímek algoritmusban az n változó optimális értékére. Használjunk véletlen polinomokat különböző fokszám és együttható-nagyság esetén.

2-3. Módosított pszeudo-maradékos osztás
Legyen $f, g \in \mathbb{Z}[x]$, $\deg f = m \geq n = \deg g$. A pszeudo-maradékos osztást módosítsuk oly módon, hogy a

$$g_n^s f = gq + r$$

egyenletnél az $s = m - n + 1$ kitevő helyett a lehető legkisebb olyan $s \in \mathbb{N}$ érték szerepeljen, amellyel $q, r \in \mathbb{Z}[x]$. Az így származtatott spquo() és sprem() eljárásokkal helyettesítve a Primitív-Euklidesz algoritmus pquo() és prem() eljárásait véletlen polinomokat alkalmazva vessük össze az algoritmusok tárigényét.

2-4. Redukált Gröbner-bázis konstrukciója
Tervezzünk algoritmust, amely adott G Gröbner-bázisból redukált Gröbner-bázist állít elő.

2-5. A Hermite-redukció megvalósítása
Valósítsuk meg a Hermite-redukciót valamilyen választott komputeralgebra nyelven.

2-6. Racionális törtfüggvény integrálása
Írjunk programot a racionális törtfüggvények integrálására.

Megjegyzések a fejezethez

A Klasszikus-Euklidesz és Bővített-Euklidesz algoritmusok nemnegatív egész bemenet esetén működő változata megtalálható [7]-ben. A rezultánsok elméletének természetes folytatásaként juthatunk el a szubrezultánsokhoz, melynek segítségével a Bővített-Euklidesz algoritmus során látott együttható-növekedés jól kordában tartható (lásd például [10, 9]).

A Gröbner-bázisokat B. Buchberger vezette be 1965-ben [2]. Polinomideálok bázisaival több szerző is foglalkozott ezt megelőzően. A legismertebb talán Hironaka, aki 1964-ben $\mathbb{C}$ feletti szingularitások feloldására hatványsorok ideáljainak bázisát használta. Munkájáért Fields-érmet kapott. Azonban Hironaka módszere nem volt konstruktív. A Gröbner-bázisokat az utóbbi két évtizedben számos algebrai struktúrára általánosították.

A differenciálalgebra alapjait J. F. Ritt fektette le 1948-ban [26]. A szimbolikus integrálás során használt négyzetmentes felbontás algoritmusa megtalálható például [9, 10] könyvekben. A Hermite-redukcióban a testbővítés foka lehető legkisebbre választásának fontosságát igazolja a [10]-ben található 11.11. példa, amelyben a felbontási test rendkívül magas fokú, míg az integrál egy másodfokú testbővítésben felírható. A Rothstein–Trager-féle integráló algoritmus bizonyítása megtalálható [9]-ben (22.8. tétel). Megjegyezzük, hogy az algoritmust Rothstein és Trager egymástól függetlenül találták. A Lazard–Rioboo–Trager-formula helyességének bizonyítása, a Logaritmikus-rész-integrál algoritmus futási idejének elemzése, az algebrai kiterjesztési lépések nehézségének kezelésével kapcsolatos eljárások vázlatos ismertetése, a $C(x)$ felett hiperexponenciális elem hiperexponenciális integráljának meghatározása (ha ilyen létezik), a Liouville-elv bizonyítása, a Risch-algoritmussal kapcsolatos állítások bizonyításai megtalálhatók a [9] könyvben.

A komputeralgebráról és a kapcsolódó tudományterületekről számos publikáció és könyv született. Magyar nyelven [7], [12] és [21] érhető el. Angolul az érdeklődőknek az

összefoglaló matematikai jellegű munkák közül felsorolunk néhányat: Caviness [3], Davenport és társai [8], von zur Gathen és társai [9], Geddes és társai [10], Knuth [17, 18, 19], Mignotte [22], Mishra [23], Pavelle és társai [25], Winkler [28].

A komputeralgebra informatikai oldaláról az érdeklődő Olvasó további információt találhat az alábbiakban: Christensen [4], Gonnet és Gruntz [11], Harper és társai [14], valamint a világhálón.

Az alkalmazásokról könyvek és cikkek nagyon széles választéka áll rendelkezésre, pl. Akritas [1], Cohen és társai (ed.) [5, 6], Grossman (ed.) [13], Hearn (ed.) [15], Kovács [20] és Odlyzko [24].

A komputeralgebra-rendszerek oktatásban betöltött szerepéről lásd pl. Karian [16] és Uhl [27] munkáját.

Konferencia kiadványok: AAECC, DISCO, EUROCAL, EUROSAM, ISSAC és SYMSAC.

Komputeralgebra folyóiratok: *Journal of Symbolic Computation* – Academic Press, *Applicable Algebra in Engineering, Communication and Computing* – Springer-Verlag, SIGSAM *Bulletin* – ACM Press.

Az Eötvös Loránd Tudományegyetem Informatikai Karának Komputeralgebra Tanszéke az oktatásban a [10, 22, 9, 28] munkákat veszi alapul.

Irodalomjegyzék

[1] A. G. Akritas. *Elements of Computer Algebra with Applications*. John Wiley & Sons, 1989. 93

[2] B. Buchberger. Ein algorithmus zum auffinden der basiselemente des restklassenringes nach einem nulldimensionalen polynomideal, 1965. PhD disszertáció, Leopold-Franzens-Universität, Innsbruck. 92

[3] B. F. Caviness. Computer algebra: past and future. *Journal of Symbolic Computations*, 2:217–263, 1986. 93

[4] S. M. Christensen. Resources for computer algebra. *Computers in Physics*, 8:308–315, 1994. 93

[5] A. M. Cohen, L. van Gasten, S. Lunel (szerkesztők). *Computer Algebra for Industry 2, Problem Solving in Practice*. John Wiley & Sons, 1995. 93

[6] A. M. Cohen (szerkesztő). *Computer Algebra for Industry: Problem Solving in Practice*. John Wiley & Sons, 1993. 93

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. *Introduction to Algorithms*. The MIT Press/McGraw-Hill, 2004 (Második kiadás ötödik, javított utánnyomása. Magyarul: *Új algoritmusok*. Scolar Kiadó, 2003). 92

[8] J. Davenport, Y. Siret, E. Tournier, E.. *Computer Algebra: Systems and Algorithms for Algebraic Computation*. Academic Press, 2000. 93

[9] J. Gathen, von zur. *Modern Computer Algebra*. Cambridge University Press, 2003. 92, 93

[10] K. O. Geddes S. Czapor, G. Labahhn. *Algorithms for Computer Algebra*. Kluwer Academic Publishers, 1992. 92, 93

[11] G. Gonnet, D. Gruntz, L. Bernardin Computer algebra systems. In A. Ralston, E. D. Reilly, D. Hemmendinger (szerkesztők), *Encyclopedia of Computer Science*, 287–301. o. Nature Publishing Group, 4. kiadás, 2000. 93

[12] R. L. Graham, D. E. Knuth, O. Patashnik. *Concrete Mathematics*. Addison-Wesley, 1994 (2. kiadás. Magyarul: *Konkrét matematika*, Műszaki Könyvkiadó, 1998). 92

[13] R. Grossman. *Symbolic Computation: Applications to Scientific Computing*, *Frontiers in Applied Mathematics* 5. kötete. SIAM, 1989. 93

[14] D. Harper, C. Wooff D. Hodginson. *A Guide to Computer Algebra Systems*. John Wiley & Sons, 1991. 93

[15] A. C. Hearn. *Future Directions for Research in Symbolic Computation*. SIAM Reports on Issues in the Mathematical Sciences. SIAM, 1990. 93

[16] Z. Karian, A. Starrett. Use of symbolic computation in probability and statistics. In Z. Karian (szerkesztő), *Symbolic Computation in Undergraduate Mathematics Education*, number24 in Notes of Mathematical Association of America. Mathematical Association of America, 1992. 93

[17] D. E. Knuth. *Fundamental Algorithms*, *The Art of Computer Programming* 1. kötete. Addison-Wesley, 1968 (3., javított kiadás 1997. Magyarul: *A számítógép-programozás művészete. 1. kötet. Alapvető algoritmusok*, Műszaki Könyvkiadó, 1993, 2. kiadás.). 93

[18] D. E. Knuth. *Seminumerical Algorithms*, *The Art of Computer Programming* 2. kötete. Addison-Wesley, 1969 (3. javított kiadás 1998. Magyarul: *A számítógép-programozás művészete. 2. kötet. Szeminumerikus algoritmusok*, Műszaki Könyvkiadó, 1993, 2. kiadás.). 93

[19] D. E. Knuth. *Sorting and Searching*, *The Art of Computer Programming* 3. kötete. Addison-Wesley, 1973 (3., javított kiadás 1997. Magyarul: *A számítógép-programozás művészete. 3. kötet. Keresés és rendezés*, Műszaki Könyvkiadó, 1994, 2. kiadás.). 93

[20] A. Kovács. Komputer algebra a tudományokban és a gyakorlatban (Computer algebra in science and practice). *Alkalmazott Matematikai Lapok*, 18:181–202, 1994-98. 93

[21] A. Kovács. Computer algebra: Impact and perspectives. *Nieuw Archief voor Wiskunde*, 17(1):29–55, 1999. 92

[22] M. E. Mignotte. *Mathematics for Computer Algebra*. Springer, 1992. 93

[23] B. E. Mishra. *Algorithmic Algebra*. Springer, 1993. 93

[24] A. Odlyzko. *Applications of Symbolic Mathematics to Mathematics*. Kluwer Academic Publishers, 1985. 93

[25] R. Pavelle, M. Rothstein. Computer algebra. *Scientific American*, 245(12):102–113, 1981. 93

[26] J. Ritt. *Integration in Finite Terms*. Columbia University Press, 1948. 92

[27] J. J. Uhl. MATHEMATICA and Me. *Notices of AMS*, 35:1345–1345, 1988. 93

[28] F. Winkler. *Polynomial Algorithms in Computer Algebra*. Springer-Verlag, 1990. 93

Tárgymutató

Névmutató

Tartalomjegyzék