DOI: 10.15476/ELTE.2021.151

Eötvös Loránd Tudományegyetem
Informatikai Kar
Komputeralgebra Tanszék

Algebrai számokkal kapcsolatos algoritmusok és számítások

A doktori értekezés tézisei

Uray Marcell János

Témavezető:
Burcsi Péter
(egyetemi docens)

Doktori iskola: Informatika Doktori Iskola

Doktori iskola vezetője: Dr. Csuhaj Varjú Erzsébet
(tanszékvezető egyetemi tanár)

Doktori program: Numerikus és Szimbolikus Számítások

Programvezető: Dr. Weisz Ferenc (egyetemi tanár)

Budapest, 2021

A doktori értekezés algebrai számokkal kapcsolatos különféle algoritmusokról, valamint ezekhez kapcsolódó elméleti kérdésekről szól. *Algebrai szám*nak nevezzük az olyan számot, amely egy egész együtthatós polinom gyöke, például a $\sqrt{2}$ az $x^2 - 2$ gyöke.

A vizsgált algoritmusok fontos jellemzője, hogy nem *numerikusan*, azaz közelítésekkel, hanem *egzaktul* számolnak. A $\sqrt{2}$ algebrai szám így sosem 1,4142, hanem mindig pontosan a $\sqrt{2}$ matematikai objektum. Ezzel elkerüljük a numerikus számolások jellemző problémáját, a kerekítési hibákat, viszont cserébe másfajta problémák jönnek elő: az egzakt algoritmusok általában lassabbak, mint a numerikus megfelelőjük, és előfordulhat, hogy az egzakt ábrázoláshoz használt egész számok (együtthatók) menet közben túl gyorsan, akár exponenciális mértékben növekednek. Ezt hívjuk *együtthatórobbanás*nak. Ennek elkerülése és a futási idő kordában tartása (polinomiális idő garantálása) fontos szempont az egzakt algoritmusok tervezése során.

1. Tézis: Műveletek hatékonysága algebrai számtestekben. *Kielemezzük algebrai számtestekben a fontosabb műveletek hatékonyságát egzakt számolás esetén: megvizsgáljuk, hogy az ábrázoláshoz szükséges együtthatók mennyire növekedhetnek a műveletvégzés során, valamint kiszámítjuk a műveletek futási idejét, figyelembe véve az együtthatónövekedést is.*

Algebrai számokról és a kapcsolódó algoritmusokról a szakirodalomban rengeteg leírás található (pl. [6, 7]), ezek hatékonyságáról azonban már kevesebb szó esik. Ennek precíz elemzése nem könnyű feladat, ha az együtthatónövekedésre is figyelünk. Van erről néhány korábbi eredmény, pl. [8], de ezek nem kellően általánosak. Ezekből egy-két eredményt és ötletet használunk, de egyébként – legjobb tudomásunk szerint – a dolgozatban leírtakhoz hasonló általános eredményeket mások még nem számoltak ki, és egyes speciális műveleteket még egyáltalán nem vizsgáltak.

Algebrai számokat hatékonyan lehet ábrázolni *algebrai számtest*ben: rögzítünk egy θ algebrai számot, és az abból alapműveletekkel kifejezhető algebrai számok körében dolgozunk. Ezek mindig kifejezhetők $a_0 + a_1\theta + a_2\theta^2 + \ldots + a_{m-1}\theta^{m-1}$ alakban, ahol a_i-k racionális számok, és m a θ fokszáma. A dolgozatban feltételezzük, hogy előre felszoroztunk a racionális együtthatók nevezőivel, így az együtthatók egész számok.

Bevezetjük az $\mathrm{s}(\alpha)$ jelölést az α algebrai szám egész együtthatóinak maximális hosszára, azaz a fenti jelöléssel $\mathrm{s}(\alpha) := \log \max_i |a_i|$. Ekkor könnyen látható, hogy például

$\mathrm{s}(\alpha+\beta) \leq \max(\mathrm{s}(\alpha), \mathrm{s}(\beta)) + \log 2$. Kevésbé triviális, de megmutatjuk például azt is, hogy $\mathrm{s}(\alpha\beta) \leq \mathrm{s}(\alpha) + \mathrm{s}(\beta) + C$, ahol C az algebrai számtesttől függ (a dolgozatban konkrétan megadjuk). Ehhez hasonló eredményeket adunk mind a négy alapműveletre, valamint kiszámítjuk azt is, hogy $|\alpha|$ mekkora lehet legalább és legfeljebb az $\mathrm{s}(\alpha)$ függvényében.

Megvizsgáljuk továbbá az algebrai számtestbeli műveletek futási idejét is. Az általánosság kedvéért bevezetjük a következő jelölést: két legfeljebb A-bites egész szám összeszorzásához szükséges gépi aritmetikai műveletek számát jelöljük $\mathrm{Mul}(A)$-val. A hagyományos szorzás esetén $\mathrm{Mul}(A) = O(A^2)$, de nagy egész számokra hatékonyabb pl. a Schönhage–Strassen-szorzás, amelyre $\mathrm{Mul}(A) = O(A \log A \log\log A)$. Minden eredményünket ezen $\mathrm{Mul}(\cdot)$ függvény segítségével írjuk fel. Például könnyen látható, hogy egy α algebrai szám és egy B-bites egész szám összeszorzása legfeljebb $m\,\mathrm{Mul}(\mathrm{s}(\alpha), B)$ ideig tart. Ezenkívül kiszámítjuk az algebrai számok összeadásának, kivonásának, szorzásának és osztásának futási idejét.

A dolgozatban ennél tovább megyünk, és megvizsgálunk néhány kifejezetten valós algebrai számokra vonatkozó műveletet is. Az egyik ilyen az összehasonlítás (pl. $\alpha < \beta$). A legtöbb esetben ez természetesen könnyen eldönthető közelítéssel, de ha a két szám nagyon közel van egymáshoz, akkor akár a számítógép legpontosabb lebegőpontos számtípusa sem tudja érzékelni a különbséget. A dolgozatban megvizsgálunk egy lehetséges módszert a kérdés egzakt eldöntésére, és kiszámítjuk a futási idejét a legrosszabb esetben. Ehhez hasonlóan megvizsgálunk egy másik műveletet is, az egészre kerekítést. Ezeket a műveleteket használjuk a következő pontban.

2. Tézis: LLL-algoritmus algebrai számtestben. *A rácsredukciós LLL-algoritmust algebrai számtestekre alkalmazzuk, és bebizonyítjuk, hogy egzakt számolás esetén is polinomiális a futási ideje.*

Az LLL-algoritmus egy rácsredukciós algoritmus, amely egy rács egy bázisát ugyanannak a rácsnak egy redukált bázisává alakítja. Az algoritmust Lenstra, Lenstra és Lovász írták meg [9]-ben, és kielemezték a hatékonyságát egész vektorokra ($\mathbb{Z}^n$-ben). Azóta többen vizsgálták $\mathbb{R}^n$-ben is (numerikusan). Mi ehelyett azzal foglalkozunk, hogy mi történik, ha valós algebrai számtestben számolunk egzaktul.

Felmerülhet a kérdés, hogy mi értelme van az LLL-algoritmust egzaktul számolni, hiszen a legtöbb alkalmazásban nincs szükség az egzakt eredményre, csak egy „eléggé

redukált" bázisra vagy egy elég rövid vektorra, és a numerikus számolás sokkal gyorsabb. Azért foglalkozunk ezzel mégis, mert egyrészt elméleti szempontból érdekes kérdés, másrészt vannak olyan alkalmazások is, ahol kifejezetten az egzakt eredményre van szükség, például [10]-ben.

Az algoritmus egy nagy while-ciklusban módosítja lépésenként a bemeneti vektorokat. Ránézésre már az sem triviális kérdés, hogy a ciklus hány lépést tesz meg, vagy hogy egyáltalán megáll-e valaha. A [9] szerzői bebizonyították, hogy az algoritmus véges lépésben terminál, és $\mathbb{Z}^n$-re kiszámítottak egy korlátot a futási időre. A dolgozatban ezt általánosítjuk algebrai egészekre. Ez egyáltalán nem könnyű feladat, mert [9] sokszor kihasználja $\mathbb{Z}$-nek azt a tulajdonságát, hogy egy pozitív szám biztosan legalább 1, ami algebrai egészekre nem teljesül, sőt, azok tetszőlegesen közel lehetnek a 0-hoz.

A dolgozatban elemezzük az LLL-algoritmust algebrai számtestekben, és kiszámítjuk a futási idejét. Ezt három összetevőből rakjuk össze: adunk egy becslést a while-ciklus lépéseinek számára ($\mathbb{Z}^n$-nél általánosabban), kiszámítunk egy korlátot a változók menet közbeni méretére a fentiekben bevezetett $\mathrm{s}(\cdot)$ függvény segítségével, valamint az egyes műveletek alapos elemzésével megbecsüljük egy while-iteráció futási idejét. Ebből a három összetevőből megkapjuk a teljes algoritmus futási idejét. Mindeközben támaszkodunk az előző pontban felvázolt eredményekre az algebrai számtest műveleteire vonatkozóan, és az eredményeket az ott bevezetett általános $\mathrm{Mul}(\cdot)$ jelöléssel fejezzük ki.

3. Tézis: Expanzív polinomok leírása determinánsokkal. *Adunk egy egzakt és hatékony módszert annak eldöntésére, hogy egy egész együtthatós polinom expanzív-e. Az algoritmus garantáltan polinomiális futási idejű, figyelembe véve az együtthatók növekedését is.*

Olyan algebrai számokat vizsgálunk, amelyek 1-nél nagyobb abszolút értékűek, és a definiáló polinomjuk (amelynek gyöke) összes többi gyöke is ilyen. A komplex számsíkon ábrázolva ez azt jelenti, hogy az összes gyök az egységkörön kívül van. Az ilyen polinomokat *expanzív polinom*oknak nevezzük.

A kérdés az, hogy egy egész együtthatós polinomról hogyan dönthetjük el, hogy expanzív-e (anélkül, hogy ki kelljen számítani a gyökeit). Erre a szakirodalomban vannak módszerek, például a Schur–Cohn-teszt [11], amely a komplex gyökkeresésre használt Lehmer–Schur-algoritmus kulcsfontosságú komponense. A Schur–Cohn-teszt

generál a polinomból egy csökkenő fokszámú polinomsorozatot egy egyszerű szabállyal, és mindegyikre megvizsgál egy egyszerű feltételt, nevezetesen hogy a konstans tagjuk nagyobb abszolút értékű-e, mint a főegyütthatójuk. Az eredeti polinom pontosan akkor expanzív, ha a fenti feltétel mindegyikre teljesül. Például a $2x^4 + x^3 - 14x^2 - 4x + 24$ polinomból a következő sorozatot generálja:

$$\begin{aligned} 2x^4 + x^3 - 14x^2 - 4x + 24, \\ 32x^3 - 308x^2 - 98x + 572, \\ -173040x^2 - 46200x + 326160, \\ -23063040000x + 76437504000, \\ 5310788203708416000000. \end{aligned}$$

A teszt láthatóan teljesül, tehát a polinom expanzív.

A példából látható még valami: a Schur–Cohn-teszt együtthatórobbanást okozhat. A rekurzív szabály, amit alkalmaz, minden lépésben megduplázhatja az együtthatók hosszát, így azok akár exponenciális mértékben növekedhetnek. Az algoritmust általában lebegőpontosan szokták számolni, és akkor ez nem jelent problémát, de akkor a nagy számok utolsó számjegyei elveszhetnek, ezért nem garantált a pontos eredmény.

A dolgozatban bevezetünk egy új egzakt módszert az expanzivitás eldöntésére, amely elkerüli az együtthatórobbanást. Ehhez speciális struktúrájú determinánsokat definiálunk a polinom együtthatóiból egy egyszerű szabállyal. Például egy negyedfokú polinomra, $f(x) = ax^4 + bx^3 + cx^2 + dx + e$-re a következő determinánsokat definiáljuk:

$$\begin{vmatrix} e-a \end{vmatrix} \qquad \begin{vmatrix} e-b & d-a \\ -a & e \end{vmatrix} \qquad \begin{vmatrix} e-c & d-b & c-a \\ -b & e-a & d \\ -a & & e \end{vmatrix}$$

$$\begin{vmatrix} e+a \end{vmatrix} \qquad \begin{vmatrix} e+b & d+a \\ a & e \end{vmatrix} \qquad \begin{vmatrix} e+c & d+b & c+a \\ b & e+a & d \\ a & & e \end{vmatrix}$$

A módszerünk szerint a polinom akkor és csak akkor expanzív, ha minden ilyen determináns pozitív (és még bizonyos triviális feltételek teljesülnek). A korábbi példát folytatva, ha $f(x) = 2x^4 + x^3 - 14x^2 - 4x + 24$, akkor a fenti determinánsok értéke rendre 22, 540, 19200, 26, 604, 6960, és mivel ez mind pozitív, ezért a polinom expanzív. Látható, hogy ezek a számok sokkal kisebbek, mint amik a Schur–Cohn-teszt során jönnek elő.

A dolgozatban precízen bebizonyítjuk a módszerünk helyességét általánosan, részben felhasználva a Schur–Cohn-tesztet. Ezenkívül kiszámítjuk a futási idejét, amely a legrosszabb esetben is garantáltan polinomiális.

4. Tézis: Alsó korlát az expanzivitási hézagra. *Megvizsgáljuk, hogy egy egész együtthatós expanzív polinom gyökei mennyire lehetnek közel az egységkörhöz, és erre különféle alsó korlátokat bizonyítunk a fokszám és az együtthatók méretének függvényében.*

Egy expanzív polinom *expanzivitási hézag*ja a legkisebb gyök egységkörtől való távolsága. Azt a kérdést vizsgáljuk, hogy ez mennyire lehet kicsi, ha az együtthatók egészek.

Ha nem mondunk további feltételt, akkor persze tetszőlegesen kicsi lehet, például az $Ax-(A+1)$ lineáris polinom expanzivitási hézagja $1/A$. Ezért a kérdést úgy fogalmazzuk meg, hogy rögzített fokszám mellett, az együtthatók méretét korlátozva milyen kicsi lehet az expanzivitási hézag.

A dolgozatban erre többféle választ is adunk, attól függően, hogy hogyan mérjük az együtthatók méretét. Korlátozhatjuk a polinom *magasság*át, azaz a legnagyobb abszolút értékű együtthatót, vagy a polinom *hosszúság*át, azaz az együtthatók abszolút értékének összegét, vagy kiválaszthatunk egy-két speciális együtthatót, például a konstans tagot és a főegyütthatót, és csak azokat korlátozzuk. Négy különböző esetet vizsgálunk, és mindegyikben kiszámítunk egy explicit alsó korlátot az expanzivitási hézagra a fokszám és az együtthatók választott mérőszámának függvényében.

A számításokhoz többféle módszert használunk. Az egyik ilyen a Liouville-egyenlőtlenség, amely algebrai számok közötti távolság becslésére való. Ennek a segítségével bebizonyítjuk például, hogy egy n-edfokú, a_0 konstans tagú egész expanzív polinom expanzivitási hézagja nem lehet kisebb, mint $1/(2^{\binom{n}{2}}|a_0|^{n-1}+1)$. Ezenkívül használjuk az előző pontban bemutatott determinánsokat is, amelyek segítségével további alsó korlátokat vezetünk le. A korlátjainkban közös, hogy az együtthatókat mérő mennyiség (a fenti példában $|a_0|$) mindig az $n-1$-edik hatványon van. Ennek az n-től függő szorzója azonban mindegyik esetben más és más (a fenti példában a legnagyobb).

A dolgozat végén mutatunk egy alkalmazást a kiszámított korlátokra: segítségével megbecsüljük az általánosított számrendszerek eldöntési algoritmusának futási idejét.

5. Tézis: Alig expanzív polinomcsalád. *Konstruálunk egy olyan egész együtthatós expanzív polinomcsaládot, amelyek gyökei nagyon közel vannak az egységkörhöz, és ezzel megmutatjuk, hogy az előző pontban kiszámított alsó korlátok közel élesek.*

Az előző pontban bemutatott alsó korlátoknál felmerül, hogy ezek vajon javíthatók-e tovább, vagy élesek. Az élességet úgy lehet bizonyítani, ha mutatunk olyan polinomokat, amelyek expanzivitási hézagja éppen a kívánt nagyságrendű.

A dolgozatban definiálunk minden n fokszámra és minden elég nagy A számra egy-egy polinomot, és kiszámítjuk ezek expanzivitási hézagjának nagyságrendjét. Azt találjuk, hogy ez kb. $1/2A^{n-1}$, ami hasonlít az előző pontban kiszámított korlátokra, csak ott a 2 helyett nagyobb, n-től függő szorzó szerepel. Ez azt mutatja, hogy rögzített n esetén a korlátjaink nagyságrendje éles.

A definiált polinomok az első néhány n-re a következőképpen néznek ki:

$$
\begin{aligned}
n = 2: &\quad (A-1)x^2 + (A-1)x + A,\\
n = 3: &\quad -x^3 + (A-1)x^2 + (A-2)x + A,\\
n = 4: &\quad -x^4 - 2x^3 + (A-2)x^2 + (A-2)x + A,\\
n = 5: &\quad -x^5 - 3x^4 - 5x^3 + (A-4)x^2 + (A-3)x + A,\\
n = 6: &\quad -x^6 - 4x^5 - 9x^4 - 12x^3 + (A-9)x^2 + (A-5)x + A,\\
n = 7: &\quad -x^7 - 5x^6 - 14x^5 - 25x^4 - 30x^3 + (A-21)x^2 + (A-10)x + A.
\end{aligned}
$$

A dologzatban bebizonyítjuk, hogy ezek a polinomok valóban expanzívak (elég nagy A-ra). Ebben és az expanzivitási hézag kiszámításában kulcsfontosságú szerepet játszanak az expanzív polinomokra megalkotott determinánsos kritériumaink.

Publikációk

[1] Uray M.J.: Algebraic number fields and the LLL algorithm. *Elbírálás alatt*, kézirat: `https://arxiv.org/abs/1810.01634`

[2] Uray M.J.: Characterization of expansive polynomials by special determinants. *Publ. Math. Debrecen* **98/3-4** (2021), 379–399. o.

[3] Uray M.J.: A family of barely expansive polynomials. *Annales Univ. Sci. Budapest., Sect. Math.*, elfogadott

[4] Bóka D., Burcsi P., Uray M.J.: On the algorithmic complexity of some numeration-related problems. *Publ. Math. Debrecen* **98/1-2** (2021), 65–81. o.

[5] Uray M.J.: On proving inequalities by cylindrical algebraic decomposition. *Annales Univ. Sci. Budapest., Sect. Comp.* **51** (2020), 231–252. o.

Irodalomjegyzék

[6] Cohen, H.: A Course in Computational Algebraic Number Theory. *Springer-Verlag Berlin Heidelberg*, 1996

[7] Geddes, K.O., Czapor, S.R., Labahn, G.: Algorithms for Computer Algebra. *Kluwer Academic Publishers*, 1992

[8] Biasse, J.-F., Fieker, C., Hofmann, T.: On the computation of the HNF of a module over the ring of integers of a number field. *Journal of Symbolic Computation* **80** (2017), 581–615. o.

[9] Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring Polynomials with Rational Coefficients. *Mathematische Annalen* **261** (1982), 515–534. o.

[10] Pethő A., Pohst, M.E., Bertók, Cs.: On multidimensional Diophantine approximation of algebraic numbers. *Journal of Number Theory* **171** (2017), 422–448. o.

[11] Cohn, A.: Über die Anzahl der Wurzeln einer algebraischen Gleichung in einem Kreise. *Mathematische Zeitschrift* **14** (1922), 110–148. o.