Lietuvos matematikos rinkinys, 2019
Lietuvos matematikų draugijos darbai, serija B
https://doi.org/10.15388/LMR.B.2019.15234

ISSN 0132-2818
eISSN 2335-898X
60 tomas, 39–45

Tiesioginis ir atvirkštinis skaičių skaidymo algoritmai

Grigorijus Melničenko

Vytauto Didžiojo universitetas, Švietimo akademija
K. Donelaičio g. 58, LT-44248 Kaunas
E. paštas: gmelnicenko@gmail.com

Santrauka. Natūraliųjų skaičių skaidymas pirminiais dauginamaisiais yra sunkus skaičiavimo uždavinys. Šio uždavinio sprendimo sudėtingumas yra vieno iš žinomiausių kriptografijos metodų RSA saugumo pagrindas. Klasikinis bandomosios dalybos algoritmas dalija duotą skaičių N į visus daliklius, pradedant nuo 2 ir baigiant skaičiumi $\sqrt{N}$. Todėl šį algoritmą galima vadinti tiesioginiu bandomosios dalybos algoritmu. Pateikiame atvirkštinį bandomosios dalybos algoritmą, kuris dalija duotą skaičių N į visus daliklius, pradedant nuo skaičiaus $\sqrt{N}$ sveikosios dalies iki 2.

Raktiniai žodžiai: pirminiai skaičiai, bandomoji dalyba, Ferma skaidymo algoritmas.

1 Įvadas

Sudėtinio natūraliojo skaičiaus skaidymas pirminiais dauginamaisiais (faktorizacija) yra svarbus kriptografijos uždavinys, be to gerokai sunkesnis, nei patikrinimas ar natūralusis skaičius yra pirminis. Siūlomi algoritmai faktorizacijos uždavinį sprendžia gaudami, kad sudėtinis natūralusis skaičius N turi pavidalą $N = p{\cdot}q$, čia p ir q yra natūralieji skaičiai, o priešingu atveju, jei toks pavidalas yra negalimas, skaičius N yra pirminis.

Toliau užrašas $\lfloor\sqrt{N}\rfloor$ žymi, kad apvaliname iki artimiausio mažesnio už $\sqrt{N}$ sveikojo skaičiaus, (apvalinimas į mažesnę pusę, t. y. skaičiuojame skaičiaus sveikąją dalį), o užrašas $\lceil\sqrt{N}\rceil$ žymi, kad apvaliname iki artimiausio didesnio už $\sqrt{N}$ sveikojo skaičiaus (apvalinimas į didesnę pusę).

Mūsų tyrinėjimai remiasi išimtinai elementariąja aritmetika. Autoriaus nuomone, tokie tyrinėjimai gali būti gera medžiaga įvairiems projektiniams darbams su studentais ir magistrantais.

2 Tiesioginiai skaidymo algoritmai

Sakykime, kad natūralųjį skaičių N skaidome pirminiais dauginamaisiais. Bandomosios dalybos algoritmas (*trial division*) remiasi paprasta idėja: imame visus skaičius

nuo 2 iki $\sqrt{N}$ ir patikriname, ar kiekvienas jų dalija skaičių N [1, 2, 3, 5]. Pateikiame tiesioginio dalybos algoritmą patogia tolimesniam dėstymui forma.

2.1 Tiesioginis bandomojo dalybos skaidymo algoritmas

Bandomosios dalybos algoritmas pagrįstas nuosekliu duotojo skaičiaus N dalijimu iš skaičių, neviršijančių skaičiaus $\sqrt{N}$, kadangi teisingas lengvai įrodomas teiginys, kad mažiausias pirminis skaičiaus N daliklis p tenkina nelygybę $p \leqslant \sqrt{N}$ [3, 18 p.]. Taigi tiesioginės dalybos algoritmo taikymo rezultate skaičius N išskaidomas į du daugi-namuosius $p{\cdot}q$, čia skaičius p – mažiausias pirminis skaičiaus p daliklis, tenkinantis sąlygą $p \leqslant \sqrt{N}$.

Įvestis: natūralusis skaičius $N \geqslant 2$.
Išvestis: skaičius N išskaidomas į du daugiklius $p{\cdot}q$, čia $p > 1$ yra mažiausias pirminis skaičiaus N daliklis, tenkinantis sąlygai $p \leqslant \sqrt{N}$, o q – kitas daliklis, arba N yra pirminis skaičius.

1. $p \leftarrow 2$;
2. **while** $(N$ **mod** $p \neq 0)$ {
3. $\quad p \leftarrow p + 1$;
4. }
5. **if** $(p < N)$ {
6. $\quad q \leftarrow N/p$;
7. $\quad$ report „Skaičius N išskaidomas $p{\cdot}q$, čia $p > 1$ yra mažiausias skaičiaus
8. $\quad N$ pirminis daliklis, tenkinantis sąlygą $p \leqslant \sqrt{N}$, o q – kitas daliklis“;
9. }
10. **else** report „Skaičius N yra pirminis“.

2.2 Tiesioginis bandomosios dalybos skaidymo algoritmas nelyginiams skaičiams

Nagrinėdami tiesioginį skaidymo algoritmą, matome, kad jis gali būti gerokai paspartintas, nes visi nelyginio skaičiaus dalikliai irgi yra nelyginiai skaičiai. Todėl pakanka pradėti algoritmą nuo skaičiaus 3, o po to imti bandomuosius daliklius visus nelyginius skaičius.

Įvestis: nelyginis natūralusis skaičius $N \geqslant 3$.
Išvestis: nelyginis natūralusis skaičius N išskaidomas į du daugiklius $p{\cdot}q$, čia $p > 1$ yra mažiausias pirminis skaičiaus N daliklis, tenkinantis sąlygą $p \leqslant \sqrt{N}$, o q – kitas daliklis, arba N yra pirminis skaičius.

1. $p \leftarrow 3$;
2. **while** $(N$ **mod** $p \neq 0)$ {
3. $\quad p \leftarrow p + 2$;
4. }
5. **if** $(p < N)$ {
6. $\quad q \leftarrow N/p$;
7. $\quad$ report „Nelyginis skaičius N išskaidomas $p{\cdot}q$, čia $p > 1$ yra
8. $\quad$ mažiausias skaičiaus N pirminis daliklis, tenkinantis sąlygą
9. $\quad p \leqslant \sqrt{N}$, o q – kitas daliklis“;
10. }
11. **else** report „Skaičius N yra pirminis“.

1 lentelė. Algoritmų palyginimas.

Skaidomas skaičius	**Atvirkštinis skaidymo algoritmas**	**Ferma algoritmas**
63018038201 pirminis skaičius	Laikas: 0,0021309 ms Iteracijų kiekis: 125517	**long** tipo perpildymas (*overflow error*)
1071306649417 = 17 * 63018038201	Laikas: 0,0021309 ms Iteracijų kiekis: 125517	**long** tipo perpildymas (*overflow error*)
493285479548767 = 3997859 * 123387413	Laikas: 0,1368027 ms Iteracijų kiekis: 9106087	Laikas: 02,9632556 ms Iteracijų kiekis: 41482605
506116755157199 = 54018521 * 9369319	Laikas: 0,0997659 ms Iteracijų kiekis: 6563860	Laikas: 0,6644017 ms Iteracijų kiekis: 9196882
1838485518786809 = 42643801 * 43112609	Laikas: 0,0018679 ms Iteracijų kiekis: 116882	Laikas: 0,0001429 ms Iteracijų kiekis: 641
12362351254304321 = 111181111 * 111191111	Laikas: 0,0001304 ms Iteracijų kiekis: 2500	Laikas: 0,0000972 ms Iteracijų kiekis: 1
18018188954915833 = 2097593 * 8589935681	Laikas: 0,9850275 ms Iteracijų kiekis: 66067128	**long** tipo perpildymas (*overflow error*)
20000273725560971 = 54018521 * 370248451	Laikas: 0,6559598 ms Iteracijų kiekis: 43701902	Laikas: 5,0658707 ms Iteracijų kiekis: 70711163
42139523531366663 = 1299953 * 32416190071	Laikas: 1,5251772 ms Iteracijų kiekis: 101989592	**long** tipo perpildymas (*overflow error*)
116629353995915777 = 1299953 * 32416190071	Laikas: 1,5251772 ms Iteracijų kiekis: 101989592	**long** tipo perpildymas (*overflow error*)
159999926400005863 = 399999857 * 399999959	Laikas: 0,0001229 ms Iteracijų kiekis: 26	Laikas: 0,0010159 ms Iteracijų kiekis: 1
251937231184211659 = 3997859 * 63018038201	Laikas: 3,7889441 ms Iteracijų kiekis: 248967817	**long** tipo perpildymas (*overflow error*)
590436102659355119 = 9369319 * 63018038201	Laikas: 5,7878456 ms Iteracijų kiekis: 379514541	**long** tipo perpildymas (*overflow error*)
1532092723613038223 = 1237777331 * 1237777333	Laikas: 0,0001017 ms Iteracijų kiekis: 1	Laikas: 0,0326802 ms Iteracijų kiekis: 460416

3 Atvirkštiniai skaidymo algoritmai

Sakykime, skaičių N išreiškiame dviejų dauginamųjų sandauga $N = p{\cdot}q$, be to skaičius q turi būti didžiausias jo daliklis (nebūtinai pirminis) tenkintų sąlygą $p \leqslant \sqrt{N}$, o p – kitas daliklis. Žinoma, šiam tikslui yra taikomas gerai žinomas Ferma algoritmas [1, 2, 3, 4, 5]. Bet šis algoritmas turi trūkumą – kiekviename žingsnyje jis reikalauja kvadratinės šaknies apskaičiavimo, o tai yra darbui ir laikui imli operacija.

Ferma algoritmas labai efektyvus, kai dauginamaisiais skaidomi skaičiai, kurių pavidalas yra $N = p{\cdot}q$ ir daugikliai p, q yra gana artimi skaičiui $\sqrt{N}$ [2, 4]. Šia savybe pasižymi siūlomi atvirkštiniai algoritmai. Pvz., skaidymui $1532092723613038223 = 1237777331 \cdot 1237777333$, užtenka atlikti po vieną kiekvieno iš atvirkštinių algoritmų iteraciją (1 lentelė). Į šį faktą reikia atsižvelgti, parenkant RSA kriptosistemos šifrų raktus.

3.1 Atvirkštinis bandomosios dalybos skaidymo algoritmas

Bandomosios dalybos algoritmu skaidant skaičių N dauginamaisiais, jis dalijamas iš visų bandomųjų daliklių, pradedant 2 ir baigiant sveikąja skaičiaus dalimi $\lfloor\sqrt{N}\rfloor$. Todėl šis algoritmas straipsnyje vadinamas tiesioginiu bandomosios dalybos algoritmu. Pateikiame atvirkštinį bandomosios dalybos skaidymo algoritmą, kurio esmė ta, kad bandomieji dalikliai pradedami nagrinėti nuo skaičiaus sveikosios dalies $\lfloor\sqrt{N}\rfloor$ iki 2. Todėl pirmasis surastas skaičiaus N daliklis yra didžiausias jo daliklis, neviršijantis skaičiaus $\sqrt{N}$.

Įvestis: natūralusis skaičius $N \geqslant 2$.
Išvestis: natūralusis skaičius N išskaidomas į du daugiklius $p{\cdot}q$, čia $q > 1$ yra didžiausias (nebūtinai pirminis) skaičiaus N daliklis, tenkinantis sąlygą $q \leqslant \sqrt{N}$, o p – kitas daliklis, arba N yra pirminis skaičius.

1. $q \leftarrow \lfloor\sqrt{N}\rfloor$;
2. **while**(N **mod** $q \neq 0$) {
3. $q \leftarrow q - 1$;
4. }
5. **if** ($q > 1$) {
6. $p \leftarrow N/q$;
7. report „Skaičius N išskaidomas $p{\cdot}q$, čia $q > 1$ yra didžiausias skaičiaus
8. N, daliklis tenkinantis sąlygą $q \leqslant \sqrt{N}$, o p – kitas daliklis“;
9. }
10. **else** report „Skaičius N yra pirminis“.

3.2 Atvirkštinis bandomosios dalybos skaidymo algoritmas nelyginiams skaičiams

Nagrinėdami atvirkštinį skaidymo algoritmą pastebime, kad jį galima žymiai pagreitinti, taikant nelyginiam skaičiui N, pradedant jį nuo skaičiaus $\lfloor\sqrt{N}\rfloor$ (skaičiaus $\sqrt{N}$ sveikosios dalies), jei šio skaičiaus sveikoji dalis yra nelyginė, ir vienetu mažiausio skaičiaus priešingu atveju. Po to naudojami bandomieji nelyginiai dalikliai q.

Įvestis: natūralusis skaičius $N \geqslant 3$.
Išvestis: natūralusis skaičius N išskaidomas į du daugiklius $p{\cdot}q$, čia $q > 1$ yra didžiausias (nebūtinai pirminis) skaičiaus N daliklis, tenkinantis sąlygą $q \leqslant \sqrt{N}$, o p – kitas daliklis, arba N yra pirminis skaičius.

1. $q \leftarrow \lfloor\sqrt{N}\rfloor$;
2. **if** (q **mod** $2 = 0$) {
3. $q \leftarrow q - 1$;
4. }
5. **while** (N **mod** $q \neq 0$) {
6. $q \leftarrow q - 2$;
7. }
8. **if** ($q > 1$) {
9. $p \leftarrow N/q$;
10. report „Nelyginis skaičius N išskaidomas $p{\cdot}q$, čia $q > 1$
11. yra didžiausias skaičiaus N daliklis, tenkinantis
12. sąlygą $q \leqslant \sqrt{N}$, o p – kitas daliklis“;

13. }
14. **else** report „Skaičius N yra pirminis“.

4 Ferma skaidymo algoritmas nelyginiams skai-čiams

Nesudėtingai galima išskaidyti dideli nelygini skaičių N, kai jis turi pavidalą $p{\cdot}q$. Tam naudojamas Ferma metodas, kuris remiasi lygybe $a^2 - b^2 = (a-b)(a+b)$. Iš lygybės seka, kad ieškant dauginamųjų p ir q, galime ieškoti kvadratų a^2, b^2, kurių skirtumas lygus N. Kvadrato paieška prasideda nuo skaičiaus $a = \lceil\sqrt{N}\rceil + 1$ (mažiausio skaičiaus, tenkinančio sąlygą $a^2 - N \geqslant 0$) ir tikrinama, ar skaičius $a^2 - N$ yra sveikojo skaičiaus kvadratas. Jei nėra, a padidinamas per 1 ir vėl tikrinama. Jei po keleto žingsniu k gaunama, kad $(\lceil\sqrt{N}\rceil + k)^2 - N$ yra sveikojo skaičiaus kvadratas, t. y. $a^2 - N = (\lceil\sqrt{N}\rceil + k)^2 - N = b^2$, tai gaunamas išskaidymas $N = a^2 - b^2 = (a-b)(a+b) = p{\cdot}q$, čia $a = (\lceil\sqrt{N}\rceil + k)$. Jei $a - b = 1$, tai išskaidymas yra trivialus ir skaičius N yra pirminis.

Jei nelyginis skaičių N išskaidomas į du daugiklius $p{\cdot}q$, tai turime teiginį [2, 206 psl.], kad pirmiausias netrivialus daliklis q rastas Ferma algoritmu, yra didžiausias skaičiaus N daliklis, tenkinantis sąlygą $q \leqslant \sqrt{N}$, ir jis nebūtinai yra pirminis.

Įvestis:: nelyginis skaičius $N \geqslant 3$.
Išvestis: nelyginis skaičius N išskaidomas į du daugiklius $p{\cdot}q$, čia $q > 1$ yra didžiausias (nebūtinai pirminis) skaičiaus N daliklis, tenkinantis sąlygą $q \leqslant \sqrt{N}$, o p – kitas daliklis, arba N yra pirminis skaičius.
1. $a \leftarrow \lceil\sqrt{N}\rceil + 1$;
2. $b2 \leftarrow a * a - N$;
3. $b \leftarrow \lceil\sqrt{b2}\rceil$;
4. **while** $(b * b \neq b2)$ {
5. $\quad a \leftarrow a + 1$;
6. $\quad b2 \leftarrow a * a - N$;
7. $\quad b \leftarrow \lceil\sqrt{b2}\rceil$;
8. }
9. $q \leftarrow a - b$;
10. **if** $(q > 1)$ {
11. $\quad p \leftarrow a + b$;
12. $\quad$ report „Nelyginis skaičius N išskaidomas $p{\cdot}q$, čia $q > 1$
13. $\quad$ yra didžiausias skaičiaus N daliklis, tenkinantis
14. $\quad$ sąlygą $q \leqslant \sqrt{N}$, o p – kitas daliklis“;
15. }
16. **else** report „Skaičius N yra pirminis“.

5 Išvados ir tolesni tyrinėjimai

Tiesioginių algoritmų rezultatas yra mažiausias pirminis skaičiaus N daliklis p, neviršijantis skaičiaus $\sqrt{N}$. Atvirkštinių algoritmų rezultatas yra didžiausias skaičiaus N daliklis q (nebutinai pirminis), neviršijantis skaičiaus $\sqrt{N}$, ką nesunku pastebėti, nagrinėjant šį algoritmą. Šią savybę turi ir Ferma algoritmas.

Atvirkštinio algoritmo palyginimui su Ferma algoritmu panaudotas kompiuteris: 1) operacinė sistema – Windows 10 × 64; 2) procesorius – Intel(R) Core i5-2500 K CPU @3.300 GHg; 3) RAM – 8 GB. Algoritmų programos buvo parašytos *C#* kalba.

Anksčiau pažymėta, kad atvirkštinis ir Ferma algoritmai nelyginiams skaičiams praneša vienodą rezultatą. Iš 1 lentelės matome, kad atvirkštinis bandomosios dalybos skaidymo algoritmas daugelių atvejų pirmauja: veikimo laiko trukmė mažesnė ir iteracijų kiekis mažesnis.

Ferma algoritmas turi didelį trukumą, kai vyksta kėlimas kvadratu 6-ame algoritmo žingsnyje. Gali atsitikti sveikojo tipo **long** atminties ląstelės perpildymas (*overflow error*), į kurią siunčiamas 6-to algoritmo žingsnio aritmetinės operacijos rezultatas. Kas ir atsitiko, kai skaidomi skaičiai 63018038201, 1071306649417, 18018188954915833, 42139523531366663, 251937231184211659 ir 590436102659355119 (1 lentelė).

Šiame darbe nenagrinėjamas pateiktųjų algoritmų asimptotinis sudėtingumas, nes tai yra studentų ir magistrantų projektinių darbų tematika.

Tiesioginis bandomosios dalybos algoritmas nelyginiams skaičiams gerai tinka, kai nagrinėjamas skaičiaus N turi mažus pirminius daliklius [1, 2, 3, 5]. Todėl tiesioginis dalybos algoritmas paprastai taikomas, pradedant skaičiumi 3 iki pasirinktos konstantos B, taip atmetant mažiausius daliklius. Jei gauname, kad nagrinėjamas skaičius N neturi mažų pirminių daliklių, tai taikome atvirkštinį skaidymo algoritmą, bet atliekame jį iki pasirinktos konstantos B.

Viena iš tolimesnių tyrinėjimų krypčių su studentais ir magistrantais lygiagretieji skaičiavimai. Pvz., galima paleisti lygiagrečiai dirbti tiesioginį ir atvirkštinį algoritmą. Taip pat galima papildomai lygiagrečiai paleisti vieną iš žinomų patikrinimo, ar skaičius yra pirminis, algoritmų [1, 2, 3, 4, 5]. Šių algoritmų skaičiavimo laiko sąnaudos yra gerokai mažesnės, nei skaidymo algoritmų.

Praktinėse uždaviniuose skaičiavimo veiksmai atliekami su dideliais skaičiais, turinčiais šimtus ar tūkstančius skaitmenų. Pvz. RSA algoritmui yra naudojami skaičiai, kurie sudaryti iš daugiau nei 2028 bitų (617 dešimtainių skaitmenų).

Sveikieji duomenų tipai daugelyje programavimo kalbų turi apribotą reikšmių diapazoną. Tam, kad būtų galima atlikti skaičiavimus su sveikaisiais skaičiais bet kuriame diapazone, į *Java* ir *C#* sudėtį įvedama *Biglnteger* klasė. Planuojama pritaikyti *Biglnteger* klasę algoritmų programavimui projektiniuose darbuose su studentais ir magistrantais.

Literatūra

[1] R. Crandall and C. Pomerance. *Prime Numbers: A Computational Perspective* (second edition). Springer Science+Business Media, Inc., New York, 2005.

[2] M.M. Gluhov, I.A Kruglov, A.B. Pichkur and A.V. CHeryomushkin. *Vvedenie v Teoretiko-Chislovye Metody Kriptografii.* Lan, Saint-Petersburg, 2011 (in Russian).

[3] A.Ju. Nesterenko. *Teoretiko-Chislovye Metody v Kriptografii. Uchebnoe Posobie.* MIEM, Moscow, 2012 (in Russian).

[4] P. Šarka ir J. Šiurys. *Kodavimas ir kriptografija. Paskaitų konspektas*, 2011. VU, Vilnius. Adresas internete: http://web.vu.lt/mif/j.siurys/files/2017/02/kknotes.pdf (žiūrėta 2019-10-01).

[5] O.N. Vasilenko. *Number-Theoretic Algorithms in Cryptography. Translations of Mathematical Monographs,* Vol. 232. American Mathematical Society, Rhode Island, USA, 2007.

SUMMARY

Direct and inverse factorization algorithms of numbers

G. Melnichenko

The factoring natural numbers into factors is a complex computational task. The complexity of solving this problem lies at the heart of RSA security, one of the most famous cryptographic methods. The classical trial division algorithm divides a given number N into all divisors, starting from 2 and to integer part of $\sqrt{N}$. Therefore, this algorithm can be called the direct trial division algorithm. We present the inverse trial division algorithm, which divides a given number N into all divisors, starting from the integer part of $\sqrt{N}$ to 2.

Keywords: prime numbers, trial division, Fermats factorization algorithm.