ISSN 1648-8776
JAUNŲJŲ MOKSLININKŲ DARBAI. Nr. 47 (1). 2017 Doi: 10.21277/jmd.v47i1.133

ILGOSIOS ARITMETIKOS ALGORITMŲ GREITAVEIKOS TYRIMAS

Robert Dzisevič, Dmitrij Šešok
Vilniaus Gedimino technikos universitetas
El. p.: robert.dzisevic@gmail.com, dmitrij.sesok@vgtu.lt

Įvadas

Daugumoje programų aritmetiniai skaičiavimai atliekami naudojant duomenų formatus, turinčius iš anksto apibrėžtą tikslumą [1]. Tikslumą riboja procesoriaus registrai, dažniausiai leidžiantys dirbti su ne didesniais nei 64 bitų, o kartais ir su 128 bitų dydžio skaičiais [2], [3]. Tokio dydžio skaičių formato pakanka norint išspręsti didžiąją dalį matematinių uždavinių, tačiau yra tokių užduočių (pavyzdžiui, kriptografinės užduotys, geometriniai skaičiavimai), kur veiksmai atliekami su dideliais skaičiais, turinčiais šimtus ar tūkstančius skaitmenų [1]. Minėti uždaviniai išsprendžiami realizuojant programas, kuriose būna taikomos papildomos ilgosios aritmetikos bibliotekos, susidedančios iš funkcijų, duomenų struktūrų, pritaikytų dirbti su itin dideliais skaičiais [1].

Tikslas – atlikti ilgosios aritmetikos algoritmų realizacijų našumo tyrimus, taikant skirtingas ilgosios aritmetikos bibliotekas.

Uždaviniai: 1) išnagrinėti egzistuojančias ilgosios aritmetikos bibliotekas; 2) realizuoti algoritmus, taikant skirtingas ilgosios aritmetikos bibliotekas; 3) atlikti skirtingų algoritmų našumo tyrimus.

Tyrimo metodai: literatūros analizė, skaitiniai eksperimentai.

Ilgosios aritmetikos bibliotekos

Visos populiariausios programavimo kalbos turi standartinius duomenų tipus, naudojamus skaičiams saugoti (*integer*, *float*, *double*), tačiau tik kelios kalbos suteikia galimybę atlikti veiksmus su daug skaitmenų turinčiais skaičiais (*Lisp*, *Java*, *Perl*) [4]. Pavyzdžiui, programavimo kalba „Java“ savo standartiniuose paketuose turi aprašytą klasę „BigInteger“, leidžiančią atlikti skaičiavimus su dideliais sveikaisiais skaičiais [5], [6]. Dauguma likusių programavimo kalbų turi sukurtų papildomų bibliotekų, žinomų dar kaip ilgosios aritmetikos bibliotekos. Kadangi tokių bibliotekų sukurta ganėtinai daug, didesnis dėmesys bus skiriamas populiariausioms C/C++ programavimo kalbos bibliotekoms: NTL (angl. *Number Theory Library*), FLINT (angl. *Fast Library for Number Theory*), CLN (angl. *Class Library for Numbers*). Minėtų bibliotekų funkcijos, duomenų struktūros yra realizuojamos skirtingai, tačiau visos kaip pagrindą skaičiavimams atlikti gali naudoti GMP biblioteką (angl. *GNU Multiple Precision Library*) [7], [8], [9]. GMP – tai C programavimo kalba aprašyta ilgosios aritmetikos biblioteka, skirta dideliems skaičiams greitai apdoroti [6], [10], [11].

Ilgosios aritmetikos algoritmų našumo įvertinimo metodika

Siekiant įvertinti skirtingų ilgosios aritmetikos bibliotekų našumą, tyrimui buvo pasirinktos trys užduotys, pagal kurias buvo realizuojami algoritmai. Dvi užduotys yra bendro pobūdžio (apima daugybą ir dalybą) – faktorialo apskaičiavimas ir daugyba, kartu atliekant dalybą moduliu, ir dar viena konkreti užduotis – Proth'o skaičiaus ieškojimas, taikant Proth'o formulę ir Proth'o teoremą (apima skaičiaus kėlimą laipsniu ir dalybą moduliu) (1, 2, 3, 4).

Faktorialo formulė:

$$n! = \prod_{z=1}^{n} z = 1\times2\times3\times \ldots \times(n-2)\times(n-1)\times n. \quad (1)$$

Daugyba, kartu atliekant ir dalybą moduliu:

$$\prod_{i=3}^{n}(a_{i-2}\times a_{i-1}\times i)\ mod\ n,\ \ kai\ a_1 = 1, a_2 = 2. \quad (2)$$

Proth'o skaičiaus formulė:

$$P = k\times2^n + 1, kai\ 2^n > k \quad (3)$$

Proth'o teorema:

$$a^{(P-1)/2} = -1 \, mod \, P, kai \, a - bet \, kuris \, skaičius \quad (4)$$

Tyrimo metu fiksuojamas kodo vykdymo laikas. Šiam parametrui nustatyti prieš testuojamo kodo dalį fiksuojamas dabartinis laikas, o baigus vykdyti kodą, vėl fiksuojamas tas pats parametras ir nustatomas gautų reikšmių skirtumas.

Kadangi visos ilgosios aritmetikos bibliotekos palaiko „Unix" aplinką, visi algoritmai buvo realizuojami „Ubuntu 16.04" aplinkoje. Programos buvo sukompiliuotos naudojant „GCC 6.3 G++" ir „javac 1.8" kompiliatorius. Algoritmai buvo realizuojami taikant keturias C/C++ kalbos ilgosios aritmetikos bibliotekas (GMP 6.1.2, NTL 10.3.0, FLINT 2.5.2, CLN 1.3.4) ir standartiniame pakete „Java" esančią klasę („BigInteger Java 1.8").

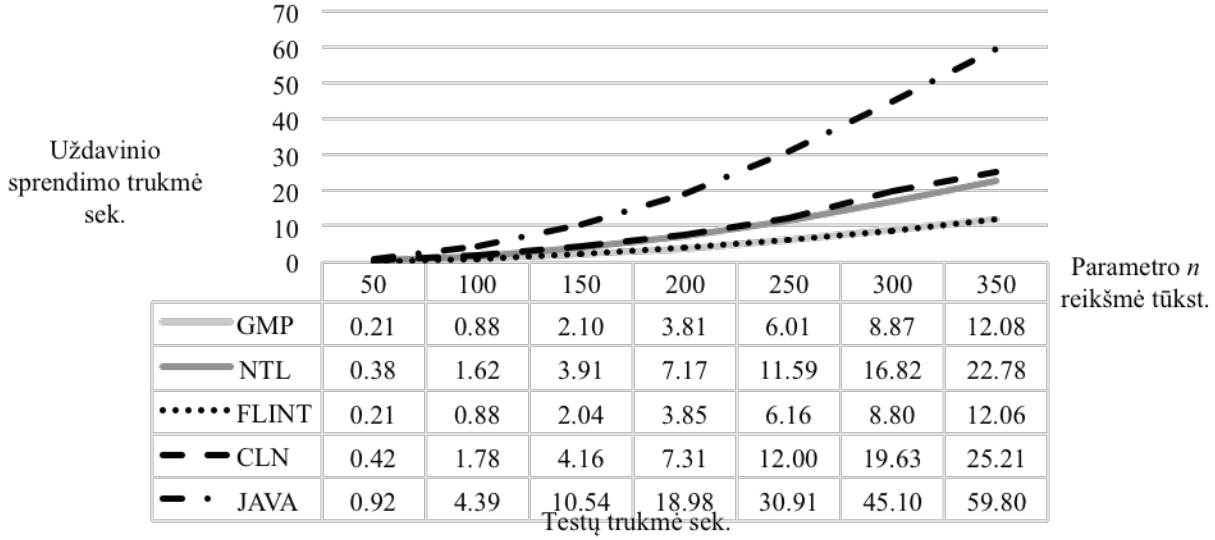
Bandymų metu testai yra atliekami septynis kartus, didinant parametro *n* reikšmę (1, 2). Siekiant gauti tikslesnius rezultatus, kiekvienas testas iš minėtų septynių yra atliekamas penkis kartus. Rezultatai, pernelyg išsiskiriantys iš daugumos testų rezultatų, yra atmetami kaip netinkami, kadangi jie yra paveikti kitų procesų, apkraunančių kompiuterio darbą. Atrinkti rezultatai yra panaudojami vidutiniam testo rezultatui apskaičiuoti.

Testavimo metu didinama uždavinio parametro *n* reikšmė kas 50 000 nuo 50 000 iki 350 000 (50 000, 100 000, 150 000, 200 000 ir t. t.). Tokio paskirstymo įvesties duomenys leidžia įvertinti bibliotekos našumą dirbant su skirtingo dydžio skaičiais (50 000 faktorialas turi daugiau nei 2,5 mln. skaitmenų, o 350 000 faktorialas turi daugiau nei 21,3 mln. skaitmenų).

Apskaičiuojant pirminį Proth'o skaičių, be *n* parametro, reikia dar dviejų parametrų: *k* ir *a*. Šie parametrai bandymų metu buvo fiksuoti, t. y. nekito jų nustatytos reikšmės. Esant skirtingoms minėtų parametrų reikšmėms, kinta tikimybė rasti pirminį Proth'o skaičių. Kadangi bandymų tikslas yra ne rasti pirminius skaičius, o nustatyti didelių Proth'o skaičių apdorojimo trukmę, parametrų *k* ir *a* reikšmės buvo sugeneruotos atsitiktinai (*k* = 147131, *a* = 3).

Ilgosios aritmetikos algoritmų našumo tyrimų rezultatai

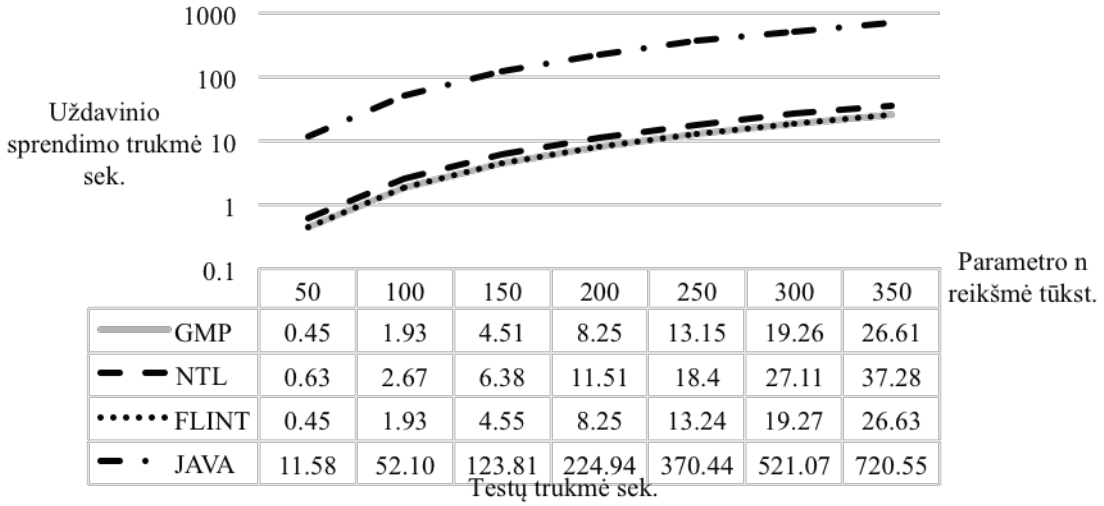
Nagrinėjant algoritmų našumą, apskaičiuojant faktorialą, pastebėta, kad geriausius rezultatus demonstruoja algoritmai, realizuojami naudojant GMP ir FLINT biblioteką. CLN ir NTL bibliotekų realizacijos pagal savo našumą yra beveik du kartus lėtesnės. Vykdant tą pačią užduotį, blogiausius rezultatus demonstruoja algoritmo realizacija taikant „Java BigInteger" klasę, kuri yra beveik penkis kartus lėtesnė nei taikant GMP ar FLINT bibliotekas (žr. 1 pav.).



	50	100	150	200	250	300	350
GMP	0.21	0.88	2.10	3.81	6.01	8.87	12.08
NTL	0.38	1.62	3.91	7.17	11.59	16.82	22.78
FLINT	0.21	0.88	2.04	3.85	6.16	8.80	12.06
CLN	0.42	1.78	4.16	7.31	12.00	19.63	25.21
JAVA	0.92	4.39	10.54	18.98	30.91	45.10	59.80

1 pav. Faktorialo apskaičiavimo greičio palyginimas

Analizuojant algoritmų našumą, atliekant daugybos ir dalybos moduliu operacijas, pastebėta, kad vėl geriausius rezultatus demonstruoja algoritmai, realizuojami naudojant GMP ir FLINT bibliotekas (žr. 2 pav.). Šių testų metu nebuvo nagrinėjama realizacija taikant CLN biblioteką, kadangi ši biblioteka neaprašo dalybos moduliu operacijų su sveikaisiais skaičiais.

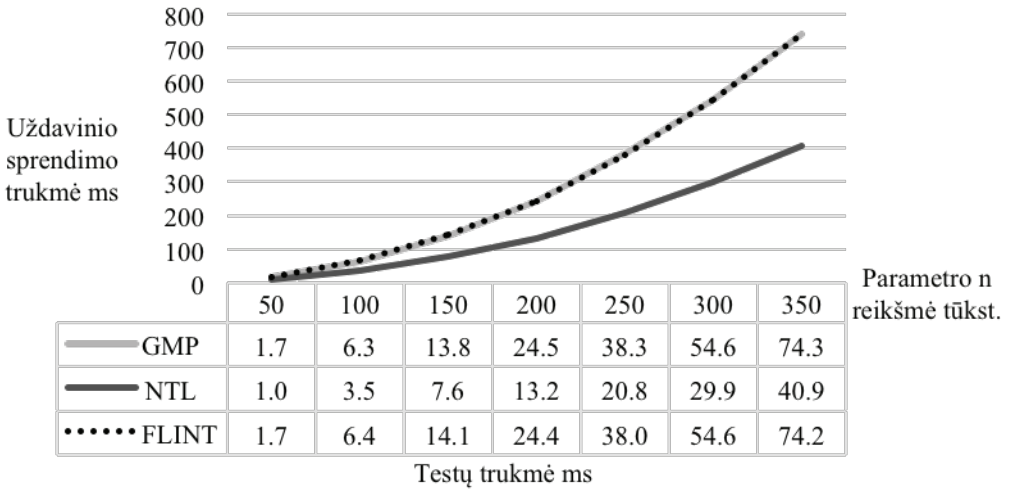


	50	100	150	200	250	300	350
GMP	0.45	1.93	4.51	8.25	13.15	19.26	26.61
NTL	0.63	2.67	6.38	11.51	18.4	27.11	37.28
FLINT	0.45	1.93	4.55	8.25	13.24	19.27	26.63
JAVA	11.58	52.10	123.81	224.94	370.44	521.07	720.55

2 pav. Daugybos ir dalybos moduliu operacijų greičio palyginimas logaritminėje skalėje

Šių testų metu pastebėta, jog algoritmo realizacija, taikant klasę „Java BigInteger“, pasirodė 27 kartus lėtesnė nei vieną iš geriausių rezultatų pademonstravusi FLINT biblioteka (žr. 2 pav.).

Nagrinėjant algoritmų našumą, apskaičiuojant Proth'o skaičius, pastebėta, kad geriausius rezultatus parodė algoritmas, realizuojamas taikant NTL biblioteką. Minėtas algoritmas, apdorojant Proth'o skaičius, ne mažesnius nei $2^{350\,000}$ (pagal Proth'o skaičiaus formulę), veikia 1,8 karto greičiau nei realizacija taikant GMP ar FLINT bibliotekas (žr. 3 pav.). Kadangi klasės „Java BigInteger“ rezultatai per dalybos moduliu operacijas buvo labai prasti, realizacija nebuvo nagrinėjama testų metu (žr. 2 pav.). Bandymų metu dėl jau minėto funkcionalumo (dalybos moduliu) trūkumo nebuvo nagrinėjamas ir algoritmas, taikant CLN biblioteką.



	50	100	150	200	250	300	350
GMP	1.7	6.3	13.8	24.5	38.3	54.6	74.3
NTL	1.0	3.5	7.6	13.2	20.8	29.9	40.9
FLINT	1.7	6.4	14.1	24.4	38.0	54.6	74.2

3 pav. Proth'o skaičiaus apskaičiavimo palyginimas

Atlikus bandymus nustatyta, kad algoritmų realizacijos, taikant „Java“ standartinio paketo klasę „BigInteger“, pagal našumą neprilygsta C/C++ ilgosios aritmetikos bibliotekoms. Geriausius rezultatus demonstruoja FLINT ir GMP bibliotekos; CLN ir NTL bibliotekos pasižymi mažesniu našumu. Šiuos skirtumus gali nulemti tai, kad GMP bibliotekos branduolys yra aprašytas asemblerio programavimo kalba. Dauguma C/C++ ilgosios aritmetikos bibliotekų pateikia patogią sąsają ir kaip pagrindą naudoja GMP, tačiau dėl bibliotekose implementuotos sąsajos naudojami algoritmai užtrunka ilgiau inicijuojant, valdant, trinant klasės objektus. Atsižvelgiant į tyrimų rezultatus galima teigti, kad FLINT biblioteka našumu prilygsta GMP bibliotekai. Šio panašumo priežastis gali būti tai, kad FLINT bibliotekos aprašytos funkcijos ir duomenų struktūros ne per daugiausia skiriasi nuo GMP bibliotekos taikomų funkcijų ir struktūrų. Algoritmų našumas priklauso ir nuo uždavinio pobūdžio: tuo galima įsitikinti apžvelgiant Victor'o Shoup'o atliktą tyrimą, kuriame plačiau tarpusavyje lyginamos NTL ir FLINT bibliotekos [12].

Išvados

1. Palyginus tarpusavyje skirtingas algoritmų realizacijas buvo nustatyta, kad nėra ilgosios aritmetikos bibliotekos, kuri savo greitaveika pranoktų kitas bibliotekas. Realizuotų algoritmų greitaveika priklauso nuo sprendžiamo uždavinio. Atliekant daugybos ir dalybos operacijas, geriausius rezultatus demonstruoja algoritmai, realizuojami naudojant GMP arba FLINT bibliotekas, tačiau, sprendžiant konkretesnį uždavinį (Proth'o skaičių paieška), geriausią rezultatą rodo NTL biblioteka.
2. Visos algoritmų realizacijos, kuriose buvo taikomos C/C++ ilgosios aritmetikos bibliotekos, savo greitaveika buvo pranašesnės nei algoritmų realizacijos, kuriose buvo taikoma klasė „Java BigInteger".
3. Nepaisant to, kad CLN, NTL ir FLINT bibliotekos kaip pagrindą naudoja tą pačią GMP biblioteką, savo našumu FLINT biblioteka pranoksta likusias bibliotekas ir veikia tokiu pačiu greičiu kaip ir GMP biblioteka. Kadangi FLINT ir GMP bibliotekos demonstruoja itin panašius rezultatus, sprendžiant skirtingo pobūdžio uždavinius galima daryti prielaidą, kad šių dviejų bibliotekų realizacijos yra labai panašios.

Literatūra

1. Langer B., 2015, *Arbitrary-Precision Arithmetics on the GPU*. Prieiga per internetą: http://old.cescg.org/CESCG-2015/papers/Langer-Arbitrary-Precision_Arithmetics_on_the_GPU.pdf [žiūrėta 2017-06-01].
2. Bailey D. H., Borwein J. M., 2015, *High-Precision Arithmetic in Mathematical Physics*. Prieiga per internetą: www.mdpi.com/2227-7390/3/2/337/pdf [žiūrėta 2017-06-01].
3. Bailey D. H., 2017, *A Thread-Safe Arbitrary Precision*. Prieiga per internetą: http://www.davidhbailey.com/dhbpapers/mpfun2015.pdf [žiūrėta 2017-06-02].
4. Primi I., 2010, *High Precision Arithmetic Library*. Prieiga per internetą: http://www.nongnu.org/hpalib/ [žiūrėta 2017-06-02].
5. Cornell G., Horstmann C. S., *Fundamental Programming Structures in Java*. Prieiga per internetą: http://www.informit.com/articles/article.aspx?p=101766&seqNum=9 [žiūrėta 2017-06-02].
6. Seacord R. C., *Arbitrary Precision Arithmetic*. Prieiga per internetą: https://www.us-cert.gov/bsi/articles/knowledge/coding-practices/arbitrary-precision-arithmetic [žiūrėta 2017-06-03].
7. Shoup V., *NTL: A Library for doing Number Theory*. Prieiga per internetą: http://www.shoup.net/ntl/ [žiūrėta 2017-06-02].
8. Hart W., Johansson F., Pancratz S., 2012, *FLINT*. Prieiga per internetą: http://www.flintlib.org/flint-2.3.pdf [žiūrėta 2017-06-02].
9. Haible B., Kreckel R. B., Sheplyakov A., *CLN, a Class Library for Numbers*. Prieiga per internetą: https://www.ginac.de/CLN/cln.html [žiūrėta 2017-06-02].
10. Salimova D., *Introduction to the GNU GMP Library*. Prieiga per internetą: http://www-oldurls.inf.ethz.ch/personal/fukudak/lect/mssemi/reports/01_rep_Diyora-Salimova.pdf [žiūrėta 2017-06-03].
11. Granlund T., 2014, *GNU MP*. Prieiga per internetą: https://gmplib.org/gmp-man-6.0.0a.pdf [žiūrėta 2017-06-03].
12. Shoup V., 2016, *NTL vs FLINT*. Prieiga per internetą: http://www.shoup.net/ntl/benchmarks.pdf [žiūrėta 2017-06-03].

Summary

RESEARCH ON THE PERFORMANCE OF ARBITRARY-PRECISION ARITHMETIC ALGORITHMS

R. Dzisevič, D. Šešok

Arbitrary-precision arithmetic libraries are analysed and computing experiments were examined in this article. During the experiments, the efficiency of different arbitrary-precision arithmetic libraries (*GMP*, *NTL*, *FLINT*, *CLN*, *Java BigInteger*) solving computing tasks which involve the operations of modular multiplication, division and exponentiation were examined.

Keywords: arbitrary-precision arithmetic, arbitrary-precision arithmetic library, C/C++, Java, factorial, Proth primes.

Santrauka

ILGOSIOS ARITMETIKOS ALGORITMŲ GREITAVEIKOS TYRIMAS

R. Dzisevič, D. Šešok

Straipsnyje analizuojamos ilgosios aritmetikos bibliotekos ir aprašomi atlikti skaitiniai eksperimentai. Eksperimentų metu išnagrinėjamas skirtingų ilgosios aritmetikos bibliotekų (GMP, NTL, FLINT, CLN, „Java BigInteger“) darbo našumas sprendžiant uždavinius, apimančius daugybos, dalybos moduliu ir kėlimo laipsniu operacijas.

Prasminiai žodžiai: ilgoji aritmetika, ilgosios aritmetikos biblioteka, C/C++, kalba „Java“, faktorialas, Proth’o pirminiai skaičiai.

Įteikta 2017-06-02
Priimta 2017-06-23