Научна конференция „Иновационни ИКТ в научните изследвания и обучението: математика, информатика и информационни технологии“, 29-30 ноември 2018 г., Пампорово, България
Scientific Conference “Innovative ICT in Research and Education: Mathematics, Informatics and Information Technologies”, 29-30 November 2018, Pamporovo, Bulgaria

THE FASTER EUCLIDEAN ALGORITHM FOR COMPUTING POLYNOMIAL MULTIPLICATIVE INVERSE

Pavel Kyurkchiev, Viktor Matanski

Abstract: Here we will give optimized solution for computing polynomial multiplicative inverse. Our solution is based on results in [12]–[23]. New iteration scheme [12] gives better computational results because it economize some operations. For other results that concern Euclidean algorithm and its wide usage the reader can explore the sources [3]–[11], [24]–[34].

Keywords: *extended Euclidean greatest common divisor for polynomials, polynomial multiplicative inverse, Euclidean algorithm for polynomials, reduced number of iterations*

1. Introduction

Euclidean algorithm for computing polynomial multiplicative inverse is known (see [33], [34]):

Algorithm 1.

INPUT: two polynomials $a(x)$!= 0 and $b(x)$!= 0.

OUTPUT: polynomials $s(x)$, $t(x)$ which satisfy

$$s(x)a(x) + t(x)b(x) = \mathrm{gcd}(a(x), b(x)) = d(x) = 1,$$

where $s(x)$ is the inverse of $a(x)$ (mod $b(x)$) and $t(x)$ is the inverse of $b(x)$ (mod $a(x)$).

1. Set $s2(x) \leftarrow 1$, $s1(\mathrm{x}) \leftarrow 0$, $t2(\mathrm{x}) \leftarrow 0$, and $t1(\mathrm{x}) \leftarrow 1$.

2. While $b(x)$!= 0 do the following:

2.1 $q(x) \leftarrow a(x)$ div $b(x)$, and $r(x) \leftarrow a(x) - b(x)q(x)$.

2.2 $s(x) \leftarrow s2(x) - q(x)s1(x)$, and $t(x) \leftarrow t2(x) - q(x)t1(x)$.

2.3 $a(x) \leftarrow b(x)$, and $b(x) \leftarrow r(x)$.

Дата на получаване: 03.12.2018 г.
Дата на рецензиране: 29.03.2019 г.
Дата на публикуване: 17.05.2019 г.

2.4 $s2(x) \leftarrow s1(x)$, $s1(x) \leftarrow s(x)$, $t2(x) \leftarrow t1(x)$, and $t1(x) \leftarrow t(x)$.

3. Set $d(x) \leftarrow a(x)$, $s(x) \leftarrow s2(x)$, and $t(x) \leftarrow t2(x)$.

4. Return($d(x)$,$s(x)$,$t(x)$).

2. Main results

Using the new approach given in [15] we receive the following optimized algorithm or computing polynomial multiplicative inverse:

Algorithm 2.

INPUT: two polynomials $a(x)$!= 0 and $b(x)$!= 0.

OUTPUT: polynomials $s(x)$, $t(x)$ which satisfy

$$s(x)a(x) + t(x)b(x) = \gcd(a(x), b(x)) = d(x) = 1,$$

where $s(x)$ is the inverse of $a(x)$ (mod $b(x)$) and $t(x)$ is the inverse of $b(x)$ (mod $a(x)$).

1. Set $ao(x) = a(x)$, and $bo(x) = b(x)$.

2a. If degree of $a(x)$ is greater than degree of $b(x)$. Set $s2(x) \leftarrow 1$,

and $s1(\mathrm{x}) \leftarrow 0$. While (true) do the following:

2a.1 $q(x) \leftarrow a(x)$ div $b(x)$, and $r(x) \leftarrow a(x) - b(x)q(x)$.

2a.2 $s(x) \leftarrow s2(x) - q(x)s1(x)$, $s2(x) \leftarrow s1(x)$, and $s1(x) \leftarrow s(x)$.

2a.3 If $r(x) = 0$ then set $d(x) \leftarrow b(x)$, $s(x) \leftarrow s2(x)$,

$t(x) \leftarrow (b(x) – s(x)ao(x))bo\{-1\}(x)$, and break.

2a.4 $q(x) \leftarrow b(x)$ div $r(x)$, and $r1(x) \leftarrow b(x) - r(x)q(x)$.

2a.5 $s(x) \leftarrow s2(x) - q(x)s1(x)$, $s2(x) \leftarrow s1(x)$, and $s1(x) \leftarrow s(x)$.

2a.6 If $r1(x) = 0$ then set $d(x) \leftarrow a(x)$, $s(x) \leftarrow s2(x)$,

$t(x) \leftarrow (a(x) – s(x)ao(x))bo\{-1\}(x)$, and break.

2a.7 $a(x) \leftarrow r(x)$, and $b(x) \leftarrow r1(x)$.

2b. [else] If degree of $b(x)$ is greater than or equal to the degree of $a(x)$.

Set $s2(x) \leftarrow 0$, and $s1(x) \leftarrow 1$. While (true) do the following:

2b.1 $q(x) \leftarrow b(x)$ div $a(x)$, and $r(x) \leftarrow b(x) - a(x)q(x)$.

2b.2 $s(x) \leftarrow s2(x) - q(x)s1(x)$, $s2(x) \leftarrow s1(x)$, and $s1(x) \leftarrow s(x)$.

2b.3 If $b(x) = 0$ then set $d(x) \leftarrow a(x)$, $s(x) \leftarrow s2(x)$,

$t(x) \leftarrow (a(x) – s(x)ao(x))bo\{-1\}(x)$, and break.

2b.4 $q(x) \leftarrow a(x)$ div $r(x)$, and $r1(x) \leftarrow a(x) - r(x)q(x)$.

2b.5 $s(x) \leftarrow s2(x) - q(x)s1(x)$, $s2(x) \leftarrow s1(x)$, and $s1(x) \leftarrow s(x)$.

2b.6 If $a(x) = 0$ then set $d(x) \leftarrow b(x)$, $s(x) \leftarrow s2(x)$,

$t(x) \leftarrow (b(x) - s(x)ao(x))bo\{-1\}(x)$, and break.

2b.7 $b(x) \leftarrow r(x)$, and $a(x) \leftarrow r1(x)$.

3. [Make monic] Set c !=0 as the leading coefficient of $d(x)$.

$(d(x), s(x), t(x)) = (c\{-1\}d(x), c\{-1\}s(x), c\{-1\}t(x))$.

4. Return($d(x)$, $s(x)$, $t(x)$).

Algorithm 2. uses algorithmic technique "divide and conquer" [15] with respect to degree of polynomials $a(x)$ and $b(x)$.

Acknowledgments

This work has been supported by the project FP17-FMI008 of Department for Scientific Research, Paisii Hilendarski University of Plovdiv.

References

[1] A. Akritas, A new method for computing polynomial greatest common divisors and polynomial remainder sequences, *Numerische Mathematik*, 52 (1988), 119–127.

[2] A. Akritas, G. Malaschonok, P. Vigklas, On the Remainders Obtained in Finding the Greatest Common Divisor of Two Polynomials, *Serdica Journal of Computing*, 9 (2015), 123–138.

[3] L. Ammeraal, *Algorithms and Data Structures in C++*, John Wiley & Sons Inc., New York (1996).

[4] D. Bressoud, *Factorization and primality testing*, Springer-Verlag, New York (1989).

[5] F. Chang, Factoring a Polynomial with Multiple-Roots, *World Academy of Science, Engineering and Technology*, 47 (2008), 492–495.

[6] Th. Cormen, Ch. Leiserson, R. Rivest, Cl. Stein, Introduction to Algorithms, 3rd ed., The MIT Press, Cambridge (2009).

[7] A. Drozdek, *Data Structures and Algorithms in C++*, 4th ed., Cengage Learning (2013).

[8] K. Garov, A. Rahnev, *Textbook-notes on programming in BASIC for facultative training in mathematics for 9.–10. Grade of ESPU*, Sofia (1986). (in Bulgarian)

[9] S. Goldman, K. Goldman, *A Practical Guide to Data Structures and Algorithms Using JAVA*, Chapman & Hall/CRC, Taylor & Francis Group, New York (2008).

[10] A. Golev, *Textbook on algorithms and programs in C#*, University Press "Paisii Hilendarski", Plovdiv (2012).

[11] M. Goodrich, R. Tamassia, D. Mount, *Data Structures and Algorithms in C++*, 2nd ed., John Wiley & Sons Inc., New York (2011).

[12] A. Iliev, N. Kyurkchiev, A Note on Knuth's Implementation of Euclid's Greatest Common Divisor Algorithm, *International Journal of Pure and Applied Mathematics*, 117 (2017), 603–608.

[13] A. Iliev, N. Kyurkchiev, A. Golev, A Note on Knuth's Implementation of Extended Euclidean Greatest Common Divisor Algorithm, *International Journal of Pure and Applied Mathematics*, 118 (2018), 31–37.

[14] A. Iliev, N. Kyurkchiev, A. Rahnev, A Note on Adaptation of the Knuth's Extended Euclidean Algorithm for Computing Multiplicative Inverse, *International Journal of Pure and Applied Mathematics*, 118 (2018), 281–290.

[15] A. Iliev, N. Kyurkchiev, A Note on Euclidean and Extended Euclidean Algorithms for Greatest Common Divisor for Polynomials, *International Journal of Pure and Applied Mathematics*, 118 (2018), 713–721.

[16] A. Iliev, N. Kyurkchiev, A Note on Least Absolute Remainder Euclidean Algorithm for Greatest Common Divisor, *International Journal of Scientific Engineering and Applied Science*, 4(3) (2018), 31–34.

[17] A. Iliev, N. Kyurkchiev, A Note on Knuth's Algorithm for Computing Extended Greatest Common Divisor using SGN Function, *International Journal of Scientific Engineering and Applied Science*, 4(3) (2018), 26–29.

[18] A. Iliev, N. Kyurkchiev, *New Trends in Practical Algorithms: Some Computational and Approximation Aspects*, LAP LAMBERT Academic Publishing, Beau Bassin (2018).

[19] A. Iliev, N. Kyurkchiev, 80th Anniversary of the birth of Prof. Donald Knuth, *Biomath Communications*, 5 (2018), 7 pp.

[20] A. Iliev, N. Kyurkchiev, New Realization of the Euclidean Algorithm, *Proc. of Eleventh National Conference with International Participation Education and Research in the Information Society*, Plovdiv, ADIS, June 1–2, (2018), 180–185. (in Bulgarian)

[21] A. Iliev, N. Kyurkchiev, New Organizing of the Euclid's Algorithm and one of its Applications to the Continued Fractions, *Proc. of conference "Mathematics. Informatics. Information Technologies. Application in Education"*, Pamporovo, Bulgaria, October 10–12, 2018, (to appear).

[22] A. Iliev, N. Kyurkchiev, The faster Euclidean algorithm, *Proc. of Scientific Conference "Innovative ICT in Research and Education: Mathematics, Informatics and Information Technologies"*, Pamporovo, Bulgaria, November 28–30, 15–20, 2018.

[23] A. Iliev, N. Kyurkchiev, The faster extended Euclidean algorithm, *Proc. of Scientific Conference "Innovative ICT in Research and Education: Mathematics, Informatics and Information Technologies"*, Pamporovo, Bulgaria, November 28–30, 21–26, 2018.

[24] A. Iliev, N. Valchanov, T. Terzieva, Generalization and Optimization of Some Algorithms, *Proc. of National Conference "Education in Information Society"*, Plovdiv, ADIS, May 12–13, (2009), 52–58, (in Bulgarian).

[25] D. Knuth, *The Art of Computer Programming*, Vol. 2, Seminumerical Algorithms, 3rd ed., Addison-Wesley, Boston (1998).

[26] Hr. Krushkov, *Programming in C#*, Koala press, Plovdiv (2017). (in Bulgarian)

[27] P. Nakov, P. Dobrikov, *Programming=++Algorithms*, 5th ed., Sofia (2015). (in Bulgarian)

[28] A. Rahnev, K. Garov, O. Gavrailov, *Textbook for extracurricular work using BASIC*, MNP Press, Sofia (1985). (in Bulgarian)

[29] A. Rahnev, K. Garov, O. Gavrailov, *BASIC in examples and tasks*, Government Press "Narodna prosveta", Sofia (1990). (in Bulgarian)

[30] D. Schmidt, Euclid's GCD Algorithm (2014).

[31] R. Sedgewick, K. Wayne, *Algorithms*, 4th ed., Addison-Wesley, Boston (2011).

[32] A. Stepanov, *Notes on Programming* (2007).

[33] R. Crandall, C. Pomerance, *Prime Numbers: A Computational Perspective, Springer-Verlag*, New York (2005).

[34] A. Menezes, P. Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, 5th ed., CRC Press LLC, New York (2001).

Faculty of Mathematics and Informatics,

University of Plovdiv "Paisii Hilendarski",

24 Tzar Asen Str., 4000 Plovdiv, Bulgaria,

e-mails: pkyurkchiev@uni-plovdiv.bg, viktor_matanski@yahoo.com

ПО-БЪРЗИЯТ АЛГОРИТЪМ НА ЕВКЛИД ЗА ИЗЧИСЛЯВАНЕ НА МУЛТИПЛИКАТИВ ИНВЪРС ЗА ПОЛИНОМИ

Павел Кюркчиев, Виктор Матански

Резюме: Тук даваме оптимизирано решение за изчисляване на мултипликатив инвърс за полиноми. Нашето решение се базира на резултатите в [12]–[23]. Новата итерационна схема [12] дава по-добри изчислителни резултати, защото икономисва някои операции. За други резултати касаещи алгоритъма на Евклид и неговото широко използване читателят може да разгледа източниците [3]–[11], [24]–[34].