International Journal of Differential Equations and Applications
Volume 22, No. 1 (2023), pages: 47-56
ISSN (Print): 1311-2872; ISSN (Online): 1314-6084;
url: https://www.ijdea.eu

ijdea.eu

EXTENDED BASED ON GENERALIZED TEMBHURNE-SATHE ALGORITHM USING SGN FUNCTION

Anton Iliev[1,2], Nikolay Kyurkchiev[1,2],
Asen Rahnev[1] and Todorka Terzieva[1]

[1]Faculty of Mathematics and Informatics
University of Plovdiv Paisii Hilendarski
24, Tzar Asen Str., 4000 Plovdiv, BULGARIA

[2]Institute of Mathematics and Informatics
Bulgarian Academy of Sciences
Acad. G. Bonchev Str., Bl. 8, 1113 Sofia, BULGARIA

ABSTRACT: Our aim is to develop new extension of our recent results in [28]. As s results we obtain algorithms for arbitrary integer numbers $a \neq 0$ and $b \neq 0$.

AMS Subject Classification: 11A05, 68W01

Key Words: extended Euclidean algorithm, hybrid extended algorithm, sgn function

Received: April 28, 2023 **Revised:** June 5, 2023
Published: June 8, 2023 **doi:** 10.12732/ijdea.v22i1.4
Academic Publications, Ltd. https://acadpubl.eu

1. INTRODUCTION

For integer numbers $a \neq 0$ and $b \neq 0$ we will construct new algorithm for seeking of integer numbers x and y such that $x * a + y * b = gcd$, where gcd is a Greatest Common Divisor of a and b. In many research papers and book can be seen the basis of Euclidean algorithms [1]–[9] and [31]–[48]. New directions in producing more computational effective novel algorithms as well as modern realizations of this kind computational schemes can be seen in [10]–[30].

For testing purposes we will use the following computer: processor – Intel(R) Core(TM) i7-6700HQ CPU 2.60GHz, 2592 Mhz, 4 Core(s), 8 Logical Processor(s), RAM 16 GB, Microsoft Windows 10 Enterprise x64, Microsoft Visual C# 2017 x64.

2. MAIN RESULTS

Using sgn function we give new hybrid extended optimized iterative

Algorithm 1.

```
int g = 0;
x2 = 0; y1 = 0;
if (a > 0) x1 = 1; else if (a < 0) x1 = -1;
if (b > 0) y2 = 1; else if (b < 0) y2 = -1;
sng = x1 * y2;
b = Math.Abs(b); a = Math.Abs(a);
if ((a & 1) == 0 && (b & 1) == 0)
do { a >>= 1; b >>= 1; g++; }
while ((a & 1) == 0 && (b & 1) == 0);
u = a; v = b;
do
if (u > v)
{
q = u / v; u %= v;
x1 -= q * y1; x2 -= q * y2;
if (u < 1) { x = y1; y = y2; gcd = v << g; break; }
v -= u; y1 -= x1; y2 -= x2;
if (u == v) { x = x1; y = x2; gcd = u << g; break; }
if ((v & 1) == 0) do
{
v >>= 1;
if ((y1 & 1) == 0 && (y2 & 1) == 0) { y1 >>= 1; y2 >>= 1; }
else if (sng > 0) { y1 = (y1 + b) >> 1; y2 = (y2 - a) >> 1; }
else { y1 = (y1 - b) >> 1; y2 = (y2 - a) >> 1; }
if (v == 1) { x = x1; y = x2; gcd = 1 << g; break; }
} while ((v & 1) == 0);
```

```
if ((u & 1) == 0) do
{
if (u == 0) { x = y1; y = y2; gcd = v << g; break; }
u >>= 1;
if ((x1 & 1) == 0 && (x2 & 1) == 0) { x1 >>= 1; x2 >>= 1; }
else if (sng > 0) { x1 = (x1 + b) >> 1; x2 = (x2 - a) >> 1; }
else { x1 = (x1 - b) >> 1; x2 = (x2 - a) >> 1; }
} while ((u & 1) == 0);
}
else
{
q = v / u; v %= u;
y1 -= q * x1; y2 -= q * x2;
if (v < 1) { x = x1; y = x2; gcd = u << g; break; }
u -= v; x1 -= y1; x2 -= y2;
if (u == v) { x = y1; y = y2; gcd = v << g; break; }
if ((u & 1) == 0) do
{
u >>= 1;
if ((x1 & 1) == 0 && (x2 & 1) == 0) { x1 >>= 1; x2 >>= 1; }
else if (sng > 0) { x1 = (x1 + b) >> 1; x2 = (x2 - a) >> 1; }
else { x1 = (x1 - b) >> 1; x2 = (x2 - a) >> 1; }
if (u == 1) { x = y1; y = y2; gcd = 1 << g; break; }
} while ((u & 1) == 0);
if ((v & 1) == 0) do
{
if (v == 0) { x = x1; y = x2; gcd = u << g; break; }
v >>= 1;
if ((y1 & 1) == 0 && (y2 & 1) == 0) { y1 >>= 1; y2 >>= 1; }
else if (sng > 0) { y1 = (y1 + b) >> 1; y2 = (y2 - a) >> 1; }
else { y1 = (y1 - b) >> 1; y2 = (y2 - a) >> 1; }
} while ((v & 1) == 0);
}
while (true);
```

and its recursive version as

Algorithm 2.

```
static long Euclid(long u, long v, long a, long b,
ref long x, ref long y, long x1, long x2, long y1, long y2, int g, long sng)
{
long q;
if (u > v)
{
q = u / v; u %= v;
x1 -= q * y1; x2 -= q * y2;
if (u < 1) { x = y1; y = y2; return v << g; }
v -= u; y1 -= x1; y2 -= x2;
if (u == v) { x = x1; y = x2; return u << g; }
if ((v & 1) == 0)
{
if ((y1 & 1) == 0 && (y2 & 1) == 0) { y1 >>= 1; y2 >>= 1; }
else if (sng > 0) { y1 = (y1 + b) >> 1; y2 = (y2 - a) >> 1; }
else { y1 = (y1 - b) >> 1; y2 = (y2 - a) >> 1; }
if (v == 1) { x = x1; y = x2; return 1 << g; }
return Euclid(u, v >> 1, a, b, ref x, ref y, x1, x2, y1, y2, g, sng);
}
if ((u & 1) == 0)
{
if (u == 0) { x = y1; y = y2; return v << g; }
if ((x1 & 1) == 0 && (x2 & 1) == 0) { x1 >>= 1; x2 >>= 1; }
else if (sng > 0) { x1 = (x1 + b) >> 1; x2 = (x2 - a) >> 1; }
else { x1 = (x1 - b) >> 1; x2 = (x2 - a) >> 1; }
return Euclid(u >> 1, v, a, b, ref x, ref y, x1, x2, y1, y2, g, sng);
}
}
else
{
q = v / u; v %= u;
y1 -= q * x1; y2 -= q * x2;
if (v < 1) { x = x1; y = x2; return u << g; }
u -= v; x1 -= y1; x2 -= y2;
if (u == v) { x = y1; y = y2; return v << g; }
```

```
if ((u & 1) == 0)
{
if ((x1 & 1) == 0 && (x2 & 1) == 0) { x1 >>= 1; x2 >>= 1; }
else if (sng > 0) { x1 = (x1 + b) >> 1; x2 = (x2 - a) >> 1; }
else { x1 = (x1 - b) >> 1; x2 = (x2 - a) >> 1; }
if (u == 1) { x = y1; y = y2; return 1 << g; }
return Euclid(u >> 1, v, a, b, ref x, ref y, x1, x2, y1, y2, g, sng);
}
if ((v & 1) == 0)
{
if (v == 0) { x = x1; y = x2; return u << g; }
if ((y1 & 1) == 0 && (y2 & 1) == 0) { y1 >>= 1; y2 >>= 1; }
else if (sng > 0) { y1 = (y1 + b) >> 1; y2 = (y2 - a) >> 1; }
else { y1 = (y1 - b) >> 1; y2 = (y2 - a) >> 1; }
return Euclid(u, v >> 1, a, b, ref x, ref y, x1, x2, y1, y2, g, sng);
}
}
return Euclid(u, v, a, b, ref x, ref y, x1, x2, y1, y2, g, sng);
}
```

The recursive function can be called by:

```
x2 = 0; y1 = 0;
if (a > 0) x1 = 1; else if (a < 0) x1 = -1;
if (b > 0) y2 = 1; else if (b < 0) y2 = -1;
sng = x1 * y2;
b = Math.Abs(b); a = Math.Abs(a);
int g = 0;
if ((a & 1) == 0 && (b & 1) == 0)
do { a >>= 1; b >>= 1; g++; }
while ((a & 1) == 0 && (b & 1) == 0);
u = a; v = b;
gcd = Euclid(u, v, a, b, ref x, ref y, x1, x2, y1, y2, g, sng);
```

Numerical Example.

For testing of Algorithms 1 and 2 we will use the following main function:

```
long a, b, gcd, d1 = 0, x = 0, y = 0;
long x1 = 0, x2, y1, y2 = 0, q, u, v, sng;
for (int i = 1; i < 100000001; i++) { a = i; b = 200000002 - i;
//here are placed the source code of algorithm 1
//as well as calling of recursive algorithm 2
d1 += gcd;
}
Console.WriteLine(d1);
```

CPU time results are:

CPU time of Algorithm 1 is: **49.812 seconds.**

CPU time of Algorithm 2 is: **70.377 seconds.**

If we compare new results to the results obtained in [29] we will see that the time for iterative Algorithm 1 is comparable to time **49.753 seconds** from [29] and the time for recursive implementation of Algorithm 2 from [29] **90.674 seconds** is worst.

3. Conclusion

We argue the new hybrid algorithms which work correctly and with sufficiently high computational speed.

Acknowledgement

This work has been accomplished with the financial support by the Grant No BG05M2OP001-1.001-0003, financed by the Science and Education for Smart Growth Operational Program (2014-2020) and co-financed by the European Union through the European structural and Investment funds.

REFERENCES

[1] T. Moore, On the Least Absolute Remainder Euclidean Algorithm, *Fibonacci Quarterly*, **30** (1992), 161–165.

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

[3] J. Tembhurne, S. Sathe, New Modified Euclidean and Binary Greatest Common Divisor Algorithm, *IETE Journal of Research*, 62, No. 6 (2016), 852–858.

[4] 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)

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

[6] T. Terzieva, *Introduction to web programming*, University Press "Paisii Hilendarski", Plovdiv (2021), ISBN 978-619-202-623-3. (in Bulgarian)

[7] T. Terzieva, *Development of algorithmic thinking in the Informatics Education*, University Press "Paisii Hilendarski", Plovdiv (2021), ISBN 978-619-202-622-6. (in Bulgarian)

[8] T. Terzieva, *Educational tools for teaching in digital environment*, University Press "Paisii Hilendarski", Plovdiv (2021). (in Bulgarian)

[9] S. Enkov, *Programming in Arduino Environment*, University Press "Paisii Hilendarski", Plovdiv (2017). (in Bulgarian)

[10] 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.

[11] 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.

[12] 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.

[13] 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** No. 3 (2018), 26–29.

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

[15] A. Iliev, N. Kyurkchiev, The faster extended Euclidean algorithm, *Collection of scientific works from conference*, Pamporovo, Bulgaria, 28–30 November 2018, (2019), 21–26.

[16] A. Iliev, N. Kyurkchiev, A. Rahnev, A New Improvement of Least Absolute Remainder Algorithm for Greatest Common Divisor. III, *Neural, Parallel, and Scientific Computations*, **27** No. 1 (2019), 1–9.

[17] A. Iliev, N. Kyurkchiev, A. Rahnev, *Nontrivial Practical Algorithms: Part 2*, LAP LAMBERT Academic Publishing, Beau Bassin (2019).

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

[19] A. Iliev, N. Kyurkchiev, A. Rahnev, New Extended Algorithm for Finding Greatest Common Divisor, *Neural, Parallel, and Scientific Computations*, 28 No. 1 (2020), 89–95.

[20] A. Iliev, N. Kyurkchiev, A. Rahnev, Efficient Binary Algorithm for Kronecker Symbol, *Communications in Applied Analysis*, 25 No. 1 (2021), 11–21.

[21] A. Iliev, N. Kyurkchiev, A. Rahnev, Efficient Algorithm for Kronecker Symbol, *International Electronic Journal of Pure and Applied Mathematics*, 15 No. 1 (2021), 23–30.

[22] A. Iliev, N. Kyurkchiev, A. Rahnev, New Extended Algorithm Using Least Absolute Remainder, *International Journal of Differential Equations and Applications*, **21** No. 1 (2022), 85–92.

[23] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, Efficient Binary Extended Algorithm Using SGN Function, *International Journal of Differential Equations and Applications*, **20**, No. 2 (2021), 179–186.

[24] A. Iliev, N. Kyurkchiev, A. Rahnev, A Refinement of the Knuth's Extended Euclidean Algorithm for Computing Modular Multiplicative Inverse, (2021), *Communications in Applied Analysis*, 25 No. 1 (2021), 23–37.

[25] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, New Hybrid Extended Algorithm, *Communications in Applied Analysis* **27**, No. 1 (2023), 15–25.

[26] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, New Refined Enhanced Hybrid Extended Algorithm, *Communications in Applied Analysis*, 26 No. 1 (2022), 99–109.

[27] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, Refined Enhanced Hybrid Extended Algorithm Using SGN Function, (2023). (preprint)

[28] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, New Extended Based on Generalization of Tembhurne-Sathe Algorithm, *International Journal of Differential Equations and Applications*, **21**, No. 2 (2022), 89–100.

[29] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, Extended Based on Generalized Harris Algorithm Using SGN Function, *Communications in Applied Analysis*, **27**, No. 1 (2023), 1–14.

[30] A. Iliev, N. Kyurkchiev, A. Rahnev, A Refinement of the Extended Euclidean Algorithm, (2021), *International Electronic Journal of Pure and Applied Mathematics*, 15 No. 1 (2021), 33–44.

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

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

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

[34] N. Kasakliev, *C# Programming Guide*, University Press "Paisii Hilendarski", Plovdiv (2016). (in Bulgarian)

[35] A. Rahnev, N. Pavlov, N. Valchanov, T. Terzieva, *Object Oriented Programming*, Lightning Source UK Ltd., London (2014).

[36] D. Rachmawati, M. Budiman, On Using The First Variant of Dependent RSA Encryption Scheme to Secure Text: A Tutorial, *J. Phys.: Conf. Ser.*, (2020), 1542 012024.

[37] J. A. Erho, J. I. Consul, B. R. Japheth, Juggling Versus Three-Way-Reversal Sequence Rotation Performance Across Four Data Types, *International Journal of Scientific Research in Computer Science and Engineering*, **7** No. 6 (2019), 10–18.

[38] J. L. Butar-butar, F. Sinuhaji, Faktorisasi Polinomial Square-Free dan bukan Square-Free atas Lapangan Hingga Zp, *Jurnal Teori dan Aplikasi Matematika*, **3** No. 2 (2019), 132–142.

[39] L. Akcay, B. Ors, Comparison of RISC-V and transport triggered architectures for a post-quantum cryptography application, *Turk J Elec Eng & Comp Sci*, **29**, (2021), 321–333.

[40] C. Falcon Rodriguez, M. Cruz, C. Falcon, Full Euclidean Algorithm by Means of a Steady Walk, *Applied Mathematics*, **12** (2021), 269–279.

[41] P. Thapar, U. Batra, Implementation of Elliptical Curve Cryptography Based Diffie-Hellman Key Exchange Mechanism in Contiki Operating System for Internet of Things, *International Journal of Electrical and Electronics Research (IJEER)*, **10** No. 2 (2022), 335–340.

[42] J. L. Butar-Butar, Y. B. P. Siringoringo, Kode Siklik Berulang Dari Kode Linear Fp Atas Lapangan Hingga F Pl Dengan L Bilangan Prima Tertentu, *Barekeng: J. Il. Mat. & Ter.*, **15**, No. 02 (2021), 231–240.

[43] V. Matanski, An Efficient Binary Algorithm for Solving Equation $GCD * 2^{|J-K|} = X * A0 + Y * B0$, Proceedings of Anniversary International Scientific Conference "Computer Technologies and Applications", 15-17 September 2021, Pamporovo, Bulgaria, *Plovdiv University Press*, 79–86, ISBN: 978-619-202-702-5.

[44] H. Gyulyustan, A Note on Euclidean Sequencing Algorithm, Proceedings of the Scientific Conference "Innovative ICT for Digital Research Space in Mathematics, Informatics and Educational Pedagogy", Pamporovo, 7-8.11.2019, *Plovdiv University Press*, (2020), 57–63, ISBN 978-619-202-572-4.

[45] P. Kyurkchiev, V. Matanski, The Faster Euclidean Algorithm for Computing Polynomial Multiplicative Inverse, Proceedings of the Scientific Conference Innovative ICT in Research and Education: Mathematics, Informatics and Information Technologies, Pamporovo, 29-30 November 2018, (2019), 43–48, ISBN: 978-619-202-439-0.

[46] V. Matanski, P. Kyurkchiev, The Faster Lehmer's Greatest Common Divisor Algorithm, Proceedings of the Scientific Conference Innovative ICT in Research and Education: Mathematics, Informatics and Information Technologies, Pamporovo, 29-30 November 2018, (2019), 37–42, ISBN: 978-619-202-439-0.

[47] Z. Ibran, E. Aljatlawi, A. Awin, On Continued Fractions and Their Applications, *Journal of Applied Mathematics and Physics*, **10** (2022), 142–159.

[48] Y. Fan, G. Chen, M. Cui, Formalization of Finite Field GF(2^n) Based on COQ, *Computer Science*, **47** No. 12 (2020), 311–318.