



REFINED ENHANCED HYBRID EXTENDED
ALGORITHM USING SGN FUNCTION

Anton Iliev^{1,2}, Nikolay Kyurkchiev^{1,2},
Asen Rahnev¹ and Todorka Terzieva¹

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

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

Abstract: In this paper we generalize our recent results in [25]. The aim of this research is to construct algorithms which will work 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

1. Introduction

Let $a \neq 0$ and $b \neq 0$ are integer numbers. We will show how it can be found integer numbers x and y such that $x * a + y * b = gcd$, where gcd is a Greatest Common Divisor of a and b . The Euclidean algorithm is subject for investigations in many sources [1]–[8] and [28]–[45]. Natural algorithmic way for these algorithms is object of our research in [9]–[27]. The new theoretical approach given by us leads to faster computational process, which is independent of computer hardware and software implementation.

Received: May 18, 2023

Revised: June 7, 2023

Published: June 8, 2023

© 2023 Academic Publications, Ltd.

url: <https://www.e.ijpam.eu>

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 will introduce 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;
while ((u & 1) == 0)
{
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 ((v & 1) == 0)
{
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; }
}
do
if (u > v)
{
q = u / v; u %= v;
if (u < 1) { gcd = v << g; x = y1; y = y2; break; }
x1 -= q * y1; x2 -= q * y2;
while ((u & 1) == 0)

```

```

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

```

and its recursive version as

Algorithm 2.

```

static long Euclid(long a0, long b0, long a, long b, ref long x,
ref long y, long x1, long x2, long y1, long y2, long sng)
{
long q;
if (a > b)
{
q = a / b; a %= b;
if (a < 1) { x = y1; y = y2; return b; }
x1 -= q * y1; x2 -= q * y2;
if ((a & 1) == 0)
{
if ((x1 & 1) == 0 && (x2 & 1) == 0) { x1 >>= 1; x2 >>= 1; }
else if (sng > 0) { x1 = (x1 + b0) >> 1; x2 = (x2 - a0) >> 1; }
else { x1 = (x1 - b0) >> 1; x2 = (x2 - a0) >> 1; }
return Euclid(a0, b0, a >> 1, b, ref x, ref y, x1, x2, y1, y2, sng);
}
}
b -= a; y1 -= x1; y2 -= x2;
if ((b & 1) == 0)
{
if ((y1 & 1) == 0 && (y2 & 1) == 0) { y1 >>= 1; y2 >>= 1; }
else if (sng > 0) { y1 = (y1 + b0) >> 1; y2 = (y2 - a0) >> 1; }
else { y1 = (y1 - b0) >> 1; y2 = (y2 - a0) >> 1; }
return Euclid(a0, b0, a, b >> 1, ref x, ref y, x1, x2, y1, y2, sng);
}
}
else
{
q = b / a; b %= a;
if (b < 1) { x = x1; y = x2; return a; }
y1 -= q * x1; y2 -= q * x2;
if ((b & 1) == 0)
{
if ((y1 & 1) == 0 && (y2 & 1) == 0) { y1 >>= 1; y2 >>= 1; }
else if (sng > 0) { y1 = (y1 + b0) >> 1; y2 = (y2 - a0) >> 1; }
else { y1 = (y1 - b0) >> 1; y2 = (y2 - a0) >> 1; }
return Euclid(a0, b0, a, b >> 1, ref x, ref y, x1, x2, y1, y2, sng);
}
}
a -= b; x1 -= y1; x2 -= y2;
if ((a & 1) == 0)
{
if ((x1 & 1) == 0 && (x2 & 1) == 0) { x1 >>= 1; x2 >>= 1; }
else if (sng > 0) { x1 = (x1 + b0) >> 1; x2 = (x2 - a0) >> 1; }
}
}

```

```

else { x1 = (x1 - b0) >> 1; x2 = (x2 - a0) >> 1; }
return Euclid(a0, b0, a >> 1, b, ref x, ref y, x1, x2, y1, y2, sng);
}
}
return Euclid(a0, b0, a, b, ref x, ref y, x1, x2, y1, y2, sng);
}

```

The recursive function can be called by:

```

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;
while ((u & 1) == 0)
{
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 ((v & 1) == 0)
{
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; }
}
gcd = Euclid(a, b, u, v, ref x, ref y, x1, x2, y1, y2, sng) << g;

```

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: **60.750 seconds**.

CPU time of Algorithm 2 is: **68.290 seconds**.

We can compare our time results to the time results for the same numerical experiment from [26]. We can conclude that we have not benefits for iterative Algorithm 1 (**49.812 seconds**) from [26], but there some advantages for recursive implemented Algorithm 2 (**70.377 seconds**) [26].

3. Conclusion

We obtained new algorithms, which work in more general case for arbitrary integer numbers $a \neq 0$ and $b \neq 0$.

Acknowledgement

This study is financed by the project No FP23-FMI-002 “Intelligent software tools and applications in research in Mathematics, Informatics, and Pedagogy of Education” of the Scientific Fund of the Paisii Hilendarski University of Plovdiv, Bulgaria.

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

- [5] T. Terzieva, *Introduction to web programming*, University Press "Paisii Hilendarski", Plovdiv (2021), ISBN 978-619-202-623-3. (in Bulgarian)
- [6] T. Terzieva, *Development of algorithmic thinking in the Informatics Education*, University Press "Paisii Hilendarski", Plovdiv (2021), ISBN 978-619-202-622-6. (in Bulgarian)
- [7] T. Terzieva, *Educational tools for teaching in digital environment*, University Press "Paisii Hilendarski", Plovdiv (2021). (in Bulgarian)
- [8] S. Enkov, *Programming in Arduino Environment*, University Press "Paisii Hilendarski", Plovdiv (2017). (in Bulgarian)
- [9] 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.
- [10] 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.
- [11] 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.
- [12] 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.
- [13] A. Iliev, N. Kyurkchiev, *New Trends in Practical Algorithms: Some Computational and Approximation Aspects*, LAP LAMBERT Academic Publishing, Beau Bassin (2018).
- [14] A. Iliev, N. Kyurkchiev, The faster extended Euclidean algorithm, *Collection of scientific works from conference*, Pamporovo, Bulgaria, 28–30 November 2018, (2019), 21–26.
- [15] 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.
- [16] A. Iliev, N. Kyurkchiev, A. Rahnev, *Nontrivial Practical Algorithms: Part 2*, LAP LAMBERT Academic Publishing, Beau Bassin (2019).
- [17] 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)
- [18] 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.
- [19] A. Iliev, N. Kyurkchiev, A. Rahnev, Efficient Binary Algorithm for Kronecker Symbol, *Communications in Applied Analysis*, 25 No. 1 (2021), 11–21.
- [20] 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.
- [21] 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.
- [22] 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.
- [23] 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.
- [24] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, New Hybrid Extended Algorithm, *Communications in Applied Analysis*, 27, No. 1 (2023), 15–25.
- [25] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, New Refined Enhanced Hybrid Extended Algorithm, *Communications in Applied Analysis*, 26 No. 1 (2022), 99–109.
- [26] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, Extended Based on Generalized Temburne-Sathe Algorithm Using SGN Function, (2023). (preprint)
- [27] 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.
- [28] D. Knuth, *The Art of Computer Programming, Vol. 2, Seminumerical Algorithms*, 3rd ed., Addison-Wesley, Boston (1998).
- [29] A. Rahnev, K. Garov, O. Gavrailov, *Textbook for extracurricular work using BASIC*, MNP Press, Sofia (1985). (in Bulgarian)

- [30] A. Rahnev, K. Garov, O. Gavrailov, *BASIC in examples and tasks*, Government Press "Narodna prosveta", Sofia (1990). (in Bulgarian)
- [31] N. Kasakliev, *C# Programming Guide*, University Press "Paisii Hilendarski", Plovdiv (2016). (in Bulgarian)
- [32] A. Rahnev, N. Pavlov, N. Valchanov, T. Terzieva, *Object Oriented Programming*, Lightning Source UK Ltd., London (2014).
- [33] 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.
- [34] 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.
- [35] J. L. Butar-butur, F. Sinuhaji, Faktorisasi Polinomial Square-Free dan bukan Square-Free atas Lapangan Hingga Z_p , *Jurnal Teori dan Aplikasi Matematika*, **3** No. 2 (2019), 132–142.
- [36] 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.
- [37] C. Falcon Rodriguez, M. Cruz, C. Falcon, Full Euclidean Algorithm by Means of a Steady Walk, *Applied Mathematics*, **12** (2021), 269–279.
- [38] 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.
- [39] J. L. Butar-Butur, Y. B. P. Siringoringo, Kode Siklik Berulang Dari Kode Linear F_p Atas Lapangan Hingga F_{p^l} Dengan l Bilangan Prima Tertentu, *Barekeng: J. Il. Mat. & Ter.*, **15**, No. 02 (2021), 231–240.
- [40] 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.

- [41] 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, *Plodiv University Press*, (2020), 57–63, ISBN 978-619-202-572-4.
- [42] 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.
- [43] 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.
- [44] Z. Ibran, E. Aljatlawi, A. Awin, On Continued Fractions and Their Applications, *Journal of Applied Mathematics and Physics*, **10** (2022), 142–159.
- [45] Y. Fan, G. Chen, M. Cui, Formalization of Finite Field $GF(2^n)$ Based on COQ, *Computer Science*, **47** No. 12 (2020), 311–318.