International Electronic Journal of Pure and Applied Mathematics
Volume 15 No. 1 2021, 33-44
ISSN: 1314-0744
url: https://www.e.ijpam.eu
doi: 10.12732/iejpam.v15i1.4

A REFINEMENT OF THE EXTENDED EUCLIDEAN ALGORITHM

Anton Iliev[1,2], Nikolay Kyurkchiev[1,2], Asen Rahnev[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: We present new extended algorithms, which are based mainly on "remainder" and "sum" operations. Additionally we use the variable for counting the number of the sign changing. These algorithms are quite appropriate for regular as well as long numbers.

AMS Subject Classification: 11A05, 68W01
Key Words: extended algorithm, greatest common divisor, reduced number of operations

1. Introduction

In series of papers we give many new algorithms. As an example below are new recursive algorithms for finding greatest common divisor [31].

```
static long Euclid(long a, long b)
{ if (a == b) return a; else
if(a > b){ if ((a %= b) == 0) return b; else return Euclid(a, b - a); }
else { if ((b %= a) == 0) return a; else return Euclid(a - b, b); }
}
```

and its calling

Received: February 26, 2021
Revised: May 14, 2021
Published: May 18, 2021

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 15, No. 1 (2021)

```
gcd = Euclid(a, b);
```

and new improved Tembhurne–Sathe algorithm [27] –

```
static long Euclid(long a, long b)
{
if ((a & 1) == 0 && (b & 1) == 0)
return Euclid(a >> 1, b >> 1) << 1;
else
if (a == b) return a;
else
if (a > b)
{
a %= b; b = b - a;
if ((b & 1) == 0) { b >>= 1; if (b == 1) return 1; }
if ((a & 1) == 0) { if (a == 0) return b; a >>= 1; }
b %= a; a = a - b;
if ((a & 1) == 0) { a >>= 1; if (a == 1) return 1; }
if ((b & 1) == 0) { if (b == 0) return a; b >>= 1; }
return Euclid(a, b);
}
else
{
b %= a; a = a - b;
if ((a & 1) == 0) { a >>= 1; if (a == 1) return 1; }
if ((b & 1) == 0) { if (b == 0) return a; b >>= 1; }
a %= b; b = b - a;
if ((b & 1) == 0) { b >>= 1; if (b == 1) return 1; }
if ((a & 1) == 0) { if (a == 0) return b; a >>= 1; }
return Euclid(a, b);
}
}
```

and its calling

```
gcd = Euclid(a, b);
```

Now we set the following task – for two natural numbers a and b we search the integer numbers x and y for which $x*a+y*b = gcd$, where gcd is the greatest common divisor. We give new computational way for conducting this iteration process [10]–[44]. Some new ideas, which concern these algorithms can be found in [52]–[56]. The classical treatment of this topic can be found in [1]–[6] and [45]–[51].

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 15, No. 1 (2021)

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 classical approach we propose the following:

Algorithm 1.

```
b0 = b; a0 = a; x1 = 1; x2 = 0; iter = 1;
while (b > 0)
{
q = a / b; t1 = a % b; t = x1 + q * x2;
x1 = x2; x2 = t; a = b; b = t1; iter = -iter;
}
if (iter < 0) x = b0 - x1; else x = x1;
gcd = a; y = (a - x * a0) / b0;
```

its recursive implementation as

Algorithm 2.

```
static long Euclid(long a, long b, ref long x,
ref long y, ref long iter)
{
if (b < 1) { x = 1; y = 0; return a; }
long q = a / b; long r = a % b; iter = -iter;
long d = Euclid(b, r, ref y, ref x, ref iter);
y += q * x;
return d;
}
```

and its calling

```
iter = 1;
if (a > b)
{
gcd = Euclid(a, b, ref x, ref y, ref iter); if (iter < 0) x = b - x;
}
else
```

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 15, No. 1 (2021)

```
{
gcd = Euclid(b, a, ref y, ref x, ref iter); if (iter > 0) x = b - x;
}
y = (gcd - x * a) / b;
```

Using our approach to this task [10]–[44] we provide the following:

Algorithm 3.

```
b0 = b; a0 = a; iter = 1; x1 = 1; x2 = 0;
if (a > b)
do
{
q = a / b; a %= b; t = x1 + q * x2; x1 = x2; x2 = t; iter = -iter;
if (a < 1)
{
if (iter < 0) x = b0 - x1; else x = x1;
gcd = b; y = (b - x * a0) / b0; break;
}
q = b / a; b %= a; t = x1 + q * x2; x1 = x2; x2 = t; iter = -iter;
if (b < 1)
{
if (iter < 0) x = b0 - x1; else x = x1;
gcd = a; y = (a - x * a0) / b0; break;
} }
while (true);
else
do
{
q = b / a; b %= a; t = x2 + q * x1; x2 = x1; x1 = t; iter = -iter;
if (b < 1)
{
if (iter < 0) x = x2; else x = b0 - x2;
gcd = a; y = (a - x * a0) / b0; break;
}
q = a / b; a %= b; t = x2 + q * x1; x2 = x1; x1 = t; iter = -iter;
if (a < 1)
{
if (iter < 0) x = x2; else x = b0 - x2;
gcd = b; y = (b - x * a0) / b0; break;
} }
while (true);
```

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 15, No. 1 (2021)

its recursive analog as

Algorithm 4.

```
static long Euclid(long a, long b, ref long x,
ref long y, ref long iter)
{
long r = a % b; long q1 = a / b; iter = -iter;
if (r < 1) { x = 1; y = 0; return b; }
long u = b % r; long q2 = b / r; iter = -iter;
if (u < 1) { x = q1; y = 1; return r; }
long d = Euclid(r, u, ref x, ref y, ref iter);
y += q2 * x; x += q1 * y;
return d;
}
```

and its calling

```
iter = 1;
if (a > b)
{
gcd = Euclid(a, b, ref y, ref x, ref iter); if (iter < 0) x = b - x;
}
else
{
gcd = Euclid(b, a, ref x, ref y, ref iter); if (iter > 0) x = b - x;
}
y = (gcd - x * a) / b;
```

Using our approach to this task [10]–[44] and the Strassen manner [57] we provide the following:

Algorithm 5.

```
b0 = b; a0 = a; x1 = 1; x2 = 0; iter = 1;
do
if (a > b)
{
q = a / b; a %= b; t = x1 + q * x2; x1 = x2; x2 = t; iter = -iter;
if (a < 1)
{
if (iter < 0) x = b0 - x1; else x = x1;
```

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 15, No. 1 (2021)

```
gcd = b; y = (b - x * a0) / b0; break; }
b -= a; t = x1 + x2; x1 = x2; x2 = t; iter = -iter;
if (a == b)
{
if (iter < 0) x = b0 - x1; else x = x1;
gcd = a; y = (a - x * a0) / b0; break; }
}
else
{
q = b / a; b %= a; t = x2 + q * x1; x2 = x1; x1 = t; iter = -iter;
if (b < 1)
{
if (iter < 0) x = x2; else x = b0 - x2;
gcd = a; y = (a - x * a0) / b0; break; }
a -= b; t = x2 + x1; x2 = x1; x1 = t; iter = -iter;
if (a == b)
{
if (iter < 0) x = x2; else x = b0 - x2;
gcd = b; y = (b - x * a0) / b0; break; }
}
while (true);
```

its recursive analog as

Algorithm 6.

```
static long Euclid(long a, long b, ref long x,
ref long y, ref long iter)
{
long r = a % b; long q1 = a / b; iter = -iter;
if (r < 1) { x = 1; y = 0; return b; }
long u = b - r; iter = -iter;
if (u == r) { x = q1; y = 1; return r; }
long d;
if (u > r) { iter = -iter; d = Euclid(u, r, ref y, ref x, ref iter); }
else d = Euclid(r, u, ref x, ref y, ref iter);
y += x; x += q1 * y;
return d;
}
```

and its calling

```
iter = 1;
```

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 15, No. 1 (2021)

```
if (a > b)
{
gcd = Euclid(a, b, ref y, ref x, ref iter); if (iter < 0) x = b - x;
}
else
{
gcd = Euclid(b, a, ref x, ref y, ref iter); if (iter > 0) x = b - x;
}
y = (gcd - x * a) / b;
```

Numerical Example.

We will test the new algorithms 1.–6. for the following example:

```
long a, b, t, t1, q, iter, d = 0;
long gcd, a0, b0, x1, x2, x = 0, y = 0;
for (int i = 1; i < 100000001; i++) { a = i; b = 200000002 - i;
//here is the source code of every one of algorithms 1, 3 and 5
//and calling of recursive algorithms 2, 4 and 6
d += gcd;
}
Console.WriteLine(d);
```

CPU time of Algorithm 1 is: **35.398 seconds.**

CPU time of Algorithm 2 is: **64.371 seconds.**

CPU time of Algorithm 3 is: **31.926 seconds.**

CPU time of Algorithm 4 is: **47.726 seconds.**

CPU time of Algorithm 5 is: **32.966 seconds.**

CPU time of Algorithm 6 is: **54.932 seconds.**

2. Conclusion

The idea of Strassen [57] provokes us to construct such iteration processes and especially Algorithms 5 and 6.

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 15, No. 1 (2021)

Acknowledgements

This paper is supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICTinSES)", financed by the Ministry of Education and Science.

References

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

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

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

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

[5] 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)

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

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

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

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

[13] 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.

[14] 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** No. 3 (2018), 31–34.

[15] 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.

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

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

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

[19] A. Iliev, N. Kyurkchiev, New Organizing of the Euclid's Algorithm and one of its Applications to the Continued Fractions, *Collection of scientific works from conference "Mathematics. Informatics. Information Technologies. Application in Education"*, Pamporovo, Bulgaria, 10–12 October 2018, (2019), 199–207.

[20] A. Iliev, N. Kyurkchiev, The faster Euclidean algorithm, *Collection of scientific works from conference*, Pamporovo, Bulgaria, 28–30 November 2018, (2019), 15–20.

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

[22] P. Kyurkchiev, V. Matanski, The faster Euclidean algorithm for computing polynomial multiplicative inverse, *Collection of scientific works from conference*, Pamporovo, Bulgaria, 28–30 November 2018, (2019), 43–48.

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 15, No. 1 (2021)

[23] V. Matanski, P. Kyurkchiev, The faster Lehmer's greatest common divisor algorithm, *Collection of scientific works from conference*, Pamporovo, Bulgaria, 28–30 November 2018, (2019), 37–42.

[24] A. Iliev, N. Kyurkchiev, A. Rahnev, A New Improvement Euclidean Algorithm for Greatest Common Divisor. I, *Neural, Parallel, and Scientific Computations*, **26** No. 3 (2018), 355–362.

[25] A. Iliev, N. Kyurkchiev, A. Rahnev, A New Improvement of Harris–Stein Modification of Euclidean Algorithm for Greatest Common Divisor. II, *International Journal of Pure and Applied Mathematics*, **120** No. 3 (2018), 379–388.

[26] 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.

[27] A. Iliev, N. Kyurkchiev, A. Rahnev, A New Improvement of Tembhurne–Sathe Modification of Euclidean Algorithm for Greatest Common Divisor. IV, *Dynamic Systems and Applications*, **28** No. 1 (2019), 143–152.

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

[29] 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)

[30] H. Gyulyustan, A Note on Euclidean Sequencing Algorithm, *Proceedings of the Scientific Conference "Innovative ICT for Digital Research Area in Mathematics, Informatics and Pedagogy of Education"*, Pamporovo, 7–8 November 2019, Plovdiv University Press, (2020), 57–64.

[31] A. Iliev, N. Kyurkchiev, A. Rahnev, New Algorithm for Finding Greatest Common Divisor, *Neural, Parallel, and Scientific Computations*, 28 No. 1 (2020), 69–74.

[32] A. Iliev, N. Kyurkchiev, A. Rahnev, A New Improvement of Stein's Binary Algorithm for Finding Greatest Common Divisor, *Neural, Parallel, and Scientific Computations*, 28 No. 1 (2020), 75–80.

[33] A. Iliev, N. Kyurkchiev, A. Rahnev, New Algorithms for Finding Modular Multiplicative Inverse, *Neural, Parallel, and Scientific Computations*, 28 No. 1 (2020), 81–88.

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 15, No. 1 (2021)

[34] 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.

[35] A. Iliev, N. Kyurkchiev, A. Rahnev, A New Improvement of Modular Multiplicative Inverse Binary Algorithm, *International Electronic Journal of Pure and Applied Mathematics*, 14 No. 1 (2020), 37–44.

[36] A. Iliev, N. Kyurkchiev, A. Rahnev, Recursive Extended Stein's Binary Algorithm, *International Electronic Journal of Pure and Applied Mathematics*, 14 No. 1 (2020), 31–36.

[37] A. Iliev, N. Kyurkchiev, A. Rahnev, A new improvement of Jacobi symbol algorithm, *International Electronic Journal of Pure and Applied Mathematics*, 15 No. 1 (2020), 13–22.

[38] A. Iliev, N. Kyurkchiev, A. Rahnev, A new improvement of Jacobi symbol binary algorithm, *International Electronic Journal of Pure and Applied Mathematics*, 15 No. 1 (2020), 1–11.

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

[40] A. Iliev, N. Kyurkchiev, A. Rahnev, Efficient Algorithm for Kronecker Symbol, *International Electronic Journal of Pure and Applied Mathematics*, 15 No. 1 (2020), 23–30.

[41] A. Iliev, N. Kyurkchiev, A. Rahnev, A Refinement of the Knuth's Extended Euclidean Algorithm for Computing Modular Multiplicative Inverse, (2021), preprint.

[42] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, A Refinement of the Extended Euclidean Algorithm using SGN Function, (2021), preprint.

[43] A. Iliev, N. Kyurkchiev, A. Rahnev, T. Terzieva, A Refinement of the Böhs Algorithm for Computing Modular Multiplicative Inverse, (2021), preprint.

[44] A. Iliev, N. Kyurkchiev, A. Rahnev, A New Improvement of Extended Stein's Binary Algorithm, *Proceedings of the Anniversary International Scientific Conference Synergetics and Reflection in Mathematics Education*, Pamporovo, 16–18 October 2020, Plovdiv University Press, (2020), 259–264.

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

[46] Hr. Krushkov, A. Iliev, *Practical programming guide in Pascal, Parts I and II*, Koala press, Plovdiv (2002). (in Bulgarian)

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

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

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

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

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

[52] 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.

[53] 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.

[54] 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.

[55] 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.

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

[57] V. Strassen, Gaussian Elimination is not Optimal, *Numer. Math.* **13**, (1969), 354–356.