New Organizing of the Euclid's Algorithm and one of its Applications to the Continued Fractions

Conference Paper · May 2019

2 authors:

Anton Iliev
Plovdiv University "Paisii Hilendarski"

Nikolay Kyurkchiev
Plovdiv University "Paisii Hilendarski"

NEW ORGANIZING OF THE EUCLID'S ALGORITHM AND ONE OF ITS APPLICATIONS TO THE CONTINUED FRACTIONS

Anton Iliev, Nikolay Kyurkchiev

Faculty of Mathematics and Informatics,
University of Plovdiv „Paisii Hilendarski“
E-mail: aii@uni-plovdiv.bg, nkyurk@uni-plovdiv.bg

Abstract: *In this paper we will present new in theoretical aspect organizing of Euclidean algorithm for finding greatest common divisor using pseudocode. This solution aims to minimize number of comparisons and number of assignments in Euclid's algorithm. Our organizing is independent of computer processor on which it will performs and as well it is independent of programming language on which the algorithm is written. Once again we are convinced how important it is not only to point out the concrete solution but also to organize it in an optimal way. The pseudocode gives universal solutions, which in every moment can be transformed into solutions to specific programming language. We show how looks like the application of new approach to continued fractions. This paper will be useful for specialists and teachers in informatics and mathematics [1] – [51] as well as for professionals in parallel computations.*

Keywords: *greatest common divisor, optimal organizing of Euclid's algorithm, organizing from Knuth, continued fractions, shorter CPU Time*

НОВО ОРГАНИЗИРАНЕ НА АЛГОРИТЪМА НА ЕВКЛИД И ЕДНО ОТ НЕГОВИТЕ ПРИЛОЖЕНИЯ ПРИ ВЕРИЖНИТЕ ДРОБИ

Антон Илиев, Николай Кюркчиев

Факултет по математика и информатика, ПУ „Паисий Хилендарски“
Имейл: aii@uni-plovdiv.bg, nkyurk@uni-plovdiv.bg

Резюме: *В тази статия ще представим посредством псевдокод ново в теоретичен аспект организиране на алгоритъма на Евклид за намиране на най-голям общ делител. Това организиране цели да минимизира броя сравне-*

ния и броя присвоявания в Евклидовия алгоритъм. Нашата оптимизация е независима от процесора на компютъра, на който се изпълнява, както е и независима от програмния език, на който е написан алгоритъмът. За пореден път се убеждаваме колко съществено е не само посочване на конкретното решение, но и да се организира то по оптимален начин. Псевдокодът дава универсално решение, което във всеки момент би могло да се приведе към решение на конкретен език за програмиране. Посочваме как изглежда приложението на новия подход за верижни дроби. Настоящата статия би била полезна за специалисти и преподаватели по информатика и математика [1] – [51], както и за професионалисти по паралелни изчисления.

Ключови думи: *най-голям общ делител, оптимално организиране на алгоритъма на Евклид, организиране от Кнут, верижни дроби, по-кратко процесорно време*

1. Introduction

The task for searching greatest common divisor excites the mathematical thought from ancient times. As Dirichlet said [7] "The whole structure of number theory rests on a single foundation, namely the algorithm for finding the greatest common divisor". More recently, the problem of searching for the greatest common divisor is also applicable to a number of informatics tasks. One recent example of this is Gennady Korotkevich, who in June 2018 is the highest ranked programmer according to *CodeChef's* ranking. At the World Finals of *Google Code Jam* in 2014, when he solve the task *E. Allergy Testing*, he use the Schmidt organizing [25] for which today it is known that there are at least two more optimal – that of Stepanov [27] and more optimized than Stepanov's – that by Iliev–Kyurkchiev [9].

Organizing of Euclidean algorithm with pseudocode from Knuth [20]:

– *iterative form*

```
function gcd(a, b)
  while b > 0
    t := b;
    b := a mod b;
    a := t;
  return a;
```

– *recursive form*

```
function gcd(a, b)
  if b < 1
     return a;
  else
     return gcd(b, a mod b);
```

2. New organizing pseudocode of Euclidean algorithm

Using the idea given in [9] we receive new theoretical way to solve the classical problem for finding greatest common divisor. This approach [9] is successfully applied to optimization [15] of the organizing of: extended Euclidean algorithm [10], adaption of Knuth's extended algorithm for computing multiplicative inverse [11], Euclidean and extended Euclidean algorithms for greatest common divisor for polynomials [12], Knuth's algorithm for computing extended greatest common divisor using Sgn function [14].

Organizing of Euclidean algorithm with pseudocode from Iliev–Kyurkchiev [9]:

– *iterative form*

```
function gcd(a, b)
  if a > b
      do
         a := a mod b;
         if a < 1
            return b;
            break;
         b := b mod a;
         if b < 1
            return a;
            break;
      while true;
    else
      do
         b := b mod a;
         if b < 1
            return a;
            break;
         a := a mod b;
         if a < 1
            return b;
            break;
      while true;
```

– *recursive form*

```
function gcd(a, b)
  r := a mod b;
```

```
if r < 1
  return b;
u := b mod r;
if u < 1
  return r;
return gcd(r, u);
```

Recursive organizing from Knuth and from Iliev–Kyurkchiev can be calling by:

```
if a > b
    gcd(a, b);
  else
    gcd(b, a);
```

3. Application to continued fractions

There is natural relation of Euclidean algorithm and continued fractions [2]. It is known that:

$$\frac{a}{b} = q_0 + \cfrac{1}{q_1 + \cfrac{1}{q_2 + \cfrac{1}{\ddots + \cfrac{1}{q_P}}}} = \left[q_0; q_1, q_2, \ldots, q_P\right].$$

We will present iterative and recursive programming code in Visual C# 2017 that for given integers a>0, b>0 gives the presentation $\left[q_0\ q_1\ q_2 \ldots q_P\right]$.

By Knuth [20] iterative approach we receive:

```
Console.Write("{0}/{1}= [", a, b);
while (b > 0) { ob = b; q = a / b; b = a % b;
      if (b < 1) Console.WriteLine("{0}].", q); else Console.Write("{0} ", q);
      a = ob; }
```

Application of Iliev–Kyurkchiev [9] iterative approach gives:

```
if (a >= b)
  { Console.Write("{0}/{1}= [", a, b);
    do { q = a / b; a %= b;
```

```
            if (a < 1) { Console.WriteLine("{0}].", q); break; } else Con-
sole.Write("{0} ", q);
            q = b / a; b %= a;
            if (b < 1) { Console.WriteLine("{0}].", q); break; } else Con-
sole.Write("{0} ", q);
          } while (true); }
    else
     { Console.Write("{0}/{1}= [0 ", a, b);
       do { q = b / a; b %= a;
            if (b < 1) { Console.WriteLine("{0}].", q); break; } else Con-
sole.Write("{0} ", q);
            q = a / b; a %= b;
            if (a < 1) { Console.WriteLine("{0}].", q); break; } else Con-
sole.Write("{0} ", q);
          } while (true); }
```

Knuth [20] recursive implementation leads to:

```
    static long Euclid(long a, long b)
          { long q = a / b; long r = a % b;
            if (r < 1) {Console.WriteLine("{0}].", q); return b;} else Con-
sole.Write("{0} ", q);
            return Euclid(b, r); }
```

Iliev–Kyurkchiev [9] recursive implementation is:

```
    static long Euclid(long a, long b)
          { long q = a / b; long r = a % b;
           if (r < 1) { Console.WriteLine("{0}].", q); return b; } else Con-
sole.Write("{0} ", q);
           q = b / r; long u = b % r;
           if (u < 1) { Console.WriteLine("{0}].", q); return r; } else Con-
sole.Write("{0} ", q);
           return Euclid(r, u); }
```

Both recursive implementations can be called by:

```
    if (a >= b) { Console.Write("{0}/{1}= [", a, b); gcd = Euclid(a, b); }
           else { Console.Write("{0}/{1}= [0 ", a, b); gcd = Euclid(b, a); }
```

We will note the advantage of shorter CPU time when compare performing of realization [9] with performing of realization [20] for continued fractions in the iterative and recursive cases. The results in this direction are the same as these in [9] – [15] and [17].

4. Conclusions

The article presents a pseudocode of a new organizing of the Euclidean algorithm which aims to minimize some operations compared to other well–known organizing. The organizing optimization that we offer is independent of the processor on which the algorithm is executed and it is independent of the programming language which is used. Specific results from numerical experiments conducted on a contemporary platform and in a modern programming environment can be seen in [9] – [15] and [17].

Acknowledgment

This work has been supported by the project FP17–FMI–008 of Department for Scientific Research, Paisii Hilendarski University of Plovdiv.

REFERENCES

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

[2] **Vinogradov, I. (1954).** Elements of Number Theory, Dover, New York.

[3] **E. Bach, J. Shallit,** Algorithmic Number Theory, Vol. I: Efficient Algorithms, M. Garey and A. Meyer eds., second printing, The MIT press, Cambridge, 1997.

[4] **J. Silverman, J. Tate,** Rational Points on Elliptic Curves, 2nd ed., Springer International Publishing AG Switzerland, New York, 2015.

[5] **Fine, G. Rosenberger,** Number Theory: An Introduction via the Distribution of Primes, 2nd ed., Springer International Publishing AG, Cham, 2016.

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

[7] **P. Dirichlet,** Lectures on Number Theory. Translated by J. Stillwell. Providence, R.I.: American Mathematical Society, 1999.

[8] **Euclid,** Elements, Greece, 2300 BC.

[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] **B. 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] **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.

[12] **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] **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.

[14] **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.

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

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

[17] **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)

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

[19] **G. Hardy, E. Wright,** An Introduction to Theory of Numbers, 4^{th} ed., Oxford University Press, Glasgow, 1975.

[20] **Knuth,** The Art of Computer Programming, Vol. 2, Seminumerical Algorithms, 3^{rd} ed., Addison–Wesley, Boston, 1998.

[21] **L. Kronecker,** Vorlesungen uber Mathematik, Druck und Verlag von V. B. Teubner, Leipzig 1901.

[22] **R. Manfrino, J. Ortega, R. Delgado,** Topics in Algebra and Analysis: Preparing for the Mathematical Olympiad, International Publishing AG Switzerland, New York, 2015.

[23] **A. Rahnev, K. Garov, O. Gavrailov,** BASIC in examples and tasks, Government Press “Narodna prosveta”, Sofia, 1990. (in Bulgarian)
[24] **K. Rosen, D.** Shier, W. Goddard, Discrete Mathematics and Its Applications, 2nd ed., Chapman and Hall/CRC, New York, 2018.
[25] **Schmidt,** Euclid's GCD Algorithm, 2014.
[26] **S. Miller,** Mathematics of Optimization: How to do Things Faster, Pure and Applied Undergraduate Texts, Vol. 30, American Mathematical Society, USA, 2017.
[27] **A. Stepanov,** Notes on Programming, 2007.
[28] **T. Andreescu, D. Andrica, I. Cucurezeanu,** An Introduction to Diophantine Equations: A Problem–Based Approach, Birkhauser, New York, 2010.
[29] **Bender, S. Gill Williamson,** Lectures in Discrete Mathematics, University of California Press, San Diego, 2017.
[30] **D. Liben–Nowell,** Discrete Mathematics for Computer Science, John Wiley & Sons, New Jersey, 2018.
[31] **A. Golev,** Textbook on algorithms and programs in C#, University Press "Paisii Hilendarski", Plovdiv, 2012. (in Bulgarian)
[32] **Pevac,** Practicing Recursion in Java, CreateSpace Independent Publishing Platform, 2016.
[33] **A. Bantchev,** Algorithmic face of fractions, Collection of scientific works of 5th National Conference "Education in Information Society", Plovdiv, ADIS, May 31 – June 1, 2012, 216–224. (in Bulgarian)
[34] **B. Bantchev,** Fraction space revisited, Mathematics and Education in Mathematics, Proceedings of 41st Spring Conference UBM, Borovetz, April 9–12, 2012, 209–218.
[35] **P. Burgisser,** M. Clausen, M. Amin Shokrollahi, Algebraic Complexity Theory, Springer-Verlag, Berlin, 1997.
[36] **Gathen, J. Gerhard,** Modern Computer Algebra, 3rd ed., Cambridge University Press, New York, 2013.
[37] **Arndt, C. Haenel,** Pi – Unleashed, Springer-Verlag, Berlin, 2001.
[38] **Sanchez,** Introduction to Recursive Programming, CRC Press Taylor & Francis Group, Boca Raton, 2018.
[39] **Pevac,** Practicing Running Time Analysis of Recursive Algorithms, CreateSpace Independent Publishing Platform, 2016.
[40] **Weisstein,** CRC Concise Encyclopedia of Mathematics, Chapman & Hall/CRC, New York, 2003.
[41] **C. A. van Tilborg, S. Jajodia (Eds.),** Encyclopedia of Cryptography and Security, Springer Science+Business Media, New York, 2011.

[42] **W.-H. Steeb, Y. Hardy, A. Hardy, R. Stoop,** Problems & Solutions in Scientific Computing with C++ and Java Simulations, World Scientific, New Jersey, 2004.
[43] **S. Krantz,** Essentials of Mathematical Thinking, CRC Press Taylor & Francis Group, New York, 2018.
[44] **Muller-Hannemann, S. Schirra (Eds.),** Algorithm Engineering: Bridging the Gap between Algorithm Theory and Practice, Springer-Verlag, Berlin, 2010.
[45] **Hromkovic,** Algorithmics for Hard Problems Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics, 2nd ed., Springer-Verlag, Berlin, 2004.
[46] **A. Shaffer,** Data Structures & Algorithm Analysis in C++, 3rd ed., Dover Publications, Inc., Mineola, New York, 2011.
[47] **P. Zeitz,** The Art and Craft of Problem Solving, 2nd ed., John Wiley & Sons, Inc., Danvers, 2007.
[48] **Weiss,** Data structures and algorithm analysis in Java, 3rd ed., Pearson Education, Inc., publishing as Addison-Wesley, Boston, 2012.
[49] **Shaffer,** Data Structures & Algorithm Analysis in Java, 3rd ed., Dover Publications, Inc., Mineola, New York, 2012.
[50] **T. K. Carne,** Codes and Cryptography, Cambridge University, 2015.
[51] **T. W. Korner,** Coding and Cryptography, 2018.