Study of the $k$-ary GCD Algorithm

Ismail F Amer$^{1,2}$ and Muhammad Saad$^3$

$^1$Palestinian Ministry of Education and Higher Education (Ramallah, Palestine)
$^2$Institute of Computational Mathematics and Informational Technologies, Kazan Federal University (Kazan, Russia)
$^3$Institute of Physics, Kazan Federal University (Kazan, Russia)

Email: safadi121979@yahoo.com

Abstract. In our paper, we consider some approaches to accelerating the calculation of the Greatest Common Divisor (GCD) of natural numbers for given $A$ and $B$ based on the $k$-ary GCD algorithm. Such a task arises in the implementation of many mathematical and number-theoretic algorithms, for example, in cryptography and number theory (see [8]).

1. Introduction
The calculation of greatest common divisor (GCD) using the Classical Euclidean Algorithm (CEA) is fast and effective even for pairs of long numbers used in cryptography and factorization algorithms [3]. However, some scientific tasks, such as, for example, calculating the distribution function of smooth numbers [13] or searching for strictly pseudoprime numbers to improve the Miller-Rabin primality test [11] require computing of GCD for a huge number of integer pairs, where $k$-ary GCD algorithm (briefly, KARY) allows to achieve significant time savings [4–7]. One of the fastest known algorithm for calculating GCD is associated with Schönhage [1].

2. $k$-ary GCD Algorithm
The $k$-ary algorithm for calculating GCD was developed in the early 1990s by the American mathematician J. Sorenson [14-15] and the Austrian T. Jebelean [12].

Let $k$ be an even power of two, for example, $k = 16$. Sorenson himself suggested to choose $k$ equal to a power of a (large) prime but later Weber [16] showed that the choice $k = 2^{2s}$, where $s$ is a natural number, is more effective.

Theorem (J. Sorenson [14]). For any natural numbers $A, B$ and $k > 1$, relatively prime with $A$ and $B$, there are integers $x$ and $y$. $1 \leq x < \sqrt{k}$, $-\sqrt{k} \leq y \leq \sqrt{k}$, and

$$Ax + By \equiv 0 \, (mod \, k)$$

It follows from (1) that $y \equiv -AB^{-1}x \, (mod \, k)$. Denote by $q$ the value $q = -AB^{-1} \, mod \, k$, the values of $q$ from the interval $(0,k)$, then $y \equiv -qx \, (mod \, k)$.

2.1. A search of suitable $x$ and $y$ values by brute force and using the Farey series
The desired pair of $x$ and $y$ can be searched by searching, sequentially considering the values of $x$ from 1 to $\sqrt{k}$ and calculating the negative and positive values of $y = -qx \, mod \, k$ and $y = k - (qx \, mod \, k)$
until y is found from the interval $[-\sqrt{k}; \sqrt{k}]$. Such an algorithm has the estimate $O(k^{1/2})$. Consider as a faster algorithm using the Farey series.

We search for possible q values for given x. We divide the algorithm into several cases, depending on the values of the variable x:

1. $x = 1$. Then, $y = -q$, and from the condition $-\sqrt{k} \leq y \leq \sqrt{k}$ we get that $1 \leq q \leq \sqrt{k}$ or $k - \sqrt{k} \leq q \leq k$. The pair $(x, y)$ takes the value $(1, -q)$ in the first case or $(1, k - q)$ in the second.

2. $x = 2$. In this case, the inequality $\frac{k - \sqrt{k}}{2} < y < \frac{k + \sqrt{k}}{2}$ holds, and $(x, y) = (2, k - 2q)$ or $(x, y) = (2, 2q - k)$ depending on which half of the interval $[\frac{k - \sqrt{k}}{2}; \frac{k + \sqrt{k}}{2}]$ q is located.

3. Suppose now that x takes an arbitrary value $x = n$ from the interval $[1; \sqrt{k}]$. Solving the inequality $-\sqrt{k} \leq -qn \mod k \leq \sqrt{k}$, we obtain the range of values of the variable q consisting of the $n - 1$ subinterval of length $2\sqrt{k}/n$ with centers at the points $mk/n$, $m = 1, 2, ..., n - 1$. For example, for $x = 3$ we get two intervals of q values of length $2\sqrt{k}/3$ with centers at the points $k/3$ and $2k/3$.

Summing up the above, we get that we can find the value of x by finding the fraction closest to q of the form $mk/n$, where $0 < m < n < k$ and setting the value of x to n. If we denote the fraction $q/k$ by $\alpha$, then our problem is equivalent to finding for a given real number $\alpha$ from the interval $(0; 1)$ the correct fraction $m/n$ with the numerator and denominator smaller than the parameter k, which approximates the value of $\alpha$ most accurately.

We will find a suitable fraction using the Farey Series. This algorithm is described in detail in [9] and its implementation in [2]. We give a brief description here.

1. Divide the interval $(0; 1)$ into subintervals by points

$$\frac{1}{k - 1}, \frac{1}{k - 2}, ..., \frac{1}{2}, \frac{2}{3}, \frac{3}{4}, ..., \frac{k - 2}{k - 1}.$$

2. Find the interval $\left(\frac{m_1}{n_1}; \frac{m_2}{n_2}\right)$ containing the value of $\alpha$.

3. Calculate the median $m/n = \frac{m_1 + m_2}{n_1 + n_2}$ and check whether $\alpha$ is to the left or right of this median. If $\alpha < \frac{m}{n}$ is satisfied, then we define the specified interval containing $\alpha$ equal to $\left(\frac{m_1}{n_1}; \frac{m}{n}\right)$, otherwise equal to $\left(\frac{m}{n}; \frac{m_2}{n_2}\right)$.

4. We repeat steps 2–3 as long as the median value is less than $k$.

We give an example. Let $k = 16$, and $q = 13$. Then $\alpha = q/k \approx 0.813$. We choose the interval containing the $\alpha$: $\alpha \in \left(\frac{4}{5}; \frac{5}{6}\right)$. Building the medians, we obtain a chain of intervals containing $\alpha$:

$$\left(\frac{4}{5}; \frac{5}{6}\right) \rightarrow \left(\frac{4}{5}; \frac{9}{11}\right)$$

The process of constructing the refinement interval is completed, since the denominator of the next median is equal to $5 + 11 = 16$ and greater than $k - 1 = 15$. The most accurate approximation to $\alpha = 0.813$ is the fraction 9/11. As x, we can take either 5 or 11, getting the pairs $(x, y) = (5, -1)$, and $(x, y) = (11, 1)$. 
Lemma. For given natural numbers \( A, B, \text{and} \ k \), the search for the coefficients \( x \) and \( y \) satisfying condition (1) of Sorenson's theorem can be performed in time \( O(\log_2 k) \).

The proof follows from the fact that constructing a new median reduces the estimate of the distance from the nearest boundary of the interval to \( \alpha \) by half.

3. An acceleration of the \( k \)-ary GCD algorithm

To speed up, we can perform a preliminary calculating a table of values \((x, y)\) for each value of \( q < k \). This table contains \( k \) values (two values for each odd number less than \( k \)). For \( k \), comparable in size to a 32-bit machine word, the table should contain \( 2^{32} = 4.3 \times 10^9 \) rows. The calculation of this table does not take much time, and the table itself can be stored in a text file. Since the value of \( y \) can be calculated from the values of \( x, q \) and \( k \), it is sufficient to store only pairs \((q, x)\).

We also note that since the pair \((x, y)\) corresponding to \( k - q \) differs from the corresponding pair for \( q \) only by the sign \( y \), therefore, only half of the table can be stored for \( q < k/2 \), and for values \( q > k/2 \), take the pair \((x, -y)\), taking the values \((x, y)\) from the table for \( q = k - q \).

It is also useful to have a pre-computed table of inverse elements modulo \( k \). It is not necessary to calculate the entire table of inverse elements in advance. Since \( k = 2^s \) is a power of prime number, there is an algorithm that allows us to find the value of the inverse element modulo \( 2^{s+1} \) by the value of the inverse element modulo \( 2^s \). This algorithm is called Hensel’s Lifting in number theory. Its description can be found in the article [10].

As in the case of the binary algorithm, a single iteration of the \( k \)-ary algorithm becomes ineffective if the length \( A \) is significantly greater than the length \( B \). In this case, it is useful to use the \( dmod \) operation proposed by Weber in [16], which is essentially a single iteration of the classical Euclidean algorithm and allows speed up the overall execution of the algorithm.

4. Experimental results

Below are our graphs (figures 1, 2, 3) for estimating the speed of the KARY compared to the classical Euclidean algorithm (CEA) for computing a series of 20 pairs of numbers with a length of 100 decimal digits for different \( k \) (from \( 2^4 \) to \( 2^{16} \)). Time is in milliseconds (ms).

![Figure 1. Evaluation of the GCD calculation speed by KARY and CEA.](image-url)

Figure 1 shows that with increasing \( k \), the speed of the \( k \)-ary GCD algorithm increases, but the CEA wins in time. The efficiency of the \( k \)-ary algorithm increases with \( k \).
Figure 2. Evaluation of the GCD calculation speed using the $dmod$ operation.

Figure 3. Evaluation of the GCD calculation speed using the $dmod$ operation and the table of inverse elements modulo $k$.

From figure 3 we see an improvement in the runtime of the $k$-ary algorithm for computing the GCD compared with the classical Euclidean scheme.
5. Conclusion
We see that with a good choice of the parameter $k$, using the Farey series, preliminary calculating the values of $(x, y)$ and inverse elements modulo $k$ and using the $dmod$ operation, iterations of the computation of the GCD of natural numbers in the $k$-ary GCD algorithm can be significantly reduced. This will give us the opportunity to speed up the computation of GCD, which is an important and actual task that has many numerical applications in cryptography and number theory. However, some tasks, require computing GCD for a huge number of integer pairs, where the use of the $k$-ary GCD algorithm is justified and can significantly save computation time.

Acknowledgments
This work was supported by the Ministry of Education and Higher Education of Palestine and Kazan Federal University (Kazan, Russia).

References
[1] Amer I, Vasilev M P and Ishmukhametov S T 2017 Analysis of Schönhage-Strassen multiplication algorithm in finite Galois fields Astra Salvensis Vol. 4 13 591–602
[2] Amer I F and Najaira M H 2020 On implementation of an approximating $k$-ary GCD algorithm IOP Conference Series: Materials Science and Engineering Vol. 862 p 052025
[3] Amer I, Ishmukhametov S T, Rubtsova R G 2015 Lenstra Factorization Method Convergence Investigation on Elliptic Curves Research Journal of Applied Sciences Vol. 10 8 365–370
[4] Amer I and Ishmukhametov S T 2019 On acceleration of the $k$-ary GCD algorithm Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki 161 1 110–118
[5] Amer I and Ishmukhametov S T 2019 Analysis of the $k$-ary Euclid for tuples of integers Journal of Physics: Conference Series Vol. 1352 p 012001
[6] Amer I 2020 On acceleration of the $k$-ary GCD algorithm IOP Conference Series: Materials Science and Engineering Vol. 734 p 012149
[7] Amer I, Alkhalidi A M and Ishmukhametov S 2020 Comparing Different GCD Algorithms International Journal of Advanced Science and Technology Vol. 29 4s 1278–1283
[8] Ishmukhametov S T 2011 The factorization methods for natural numbers (Russia: Kazan) p 190 [In Russian]
[9] Ishmukhametov S 2016 An approximating $k$-ary GCD algorithm Lobachevskii Journal of Mathematics 37 (6) 723–9
[10] Ishmukhametov S, Mubarakov B and Maad Kamal Al-Anni 2017 Calculation of Bezout’s coefficients for the $k$-ary algorithm of finding GCD Rus.Math. Vol. 61 11 26–33
[11] Ishmukhametov S T and Mubarakov B 2013 On practical aspects of the Miller-Rabin Primality Test Lobachevskii Journal of Mathematics Vol. 34 4 304–312
[12] Jebelean T 1993 A generalization of the binary GCD algorithm Proceedings of the 1993 international symposium on Symbolic and algebraic computation 111–116
[13] Maximov K L and Ishmukhametov S T About algorithm of smooth numbers calculation Research Journal of Applied Sciences 10 (8) 376–380
[14] Sorrenson J 1990 The $k$-ary GCD Algorithm Univ.Wisc-Madison Tecn.Report p 20
[15] Sorrenson J 1994 Two fast GCD Algorithms J.Alg. 16 1 110–144
[16] Weber K The Accelerated Integer GCD Algorithm ACM Trans. Math. Softw. 21 111–122 DOI:10.1145/200979.201042