Worst case analysis of a subtractive like GCD algorithm

Sidi Mohamed SEDJELMACI

LIPN CNRS UMR 7030,
Université Paris-Nord
Av. J.-B. Clément, 93430 Villetaneuse, France.
E-mail: sms@lipn.univ-paris13.fr
Version: February, 19, 2017

1 Introduction

1.1 A little story of GCD algorithms

The Greek mathematician Euclid [13], 300 BC, describes in his *Elements* euc, a simple and elegant algorithm to compute the GCD of two integers [1]. It is perhaps, the first algorithm that computes the GCD of two integers. He used, in his description, written in modern words, several time the transformation $(u, v) \longrightarrow (v, u \bmod v)$, where $u \bmod v$ is the remainder in the division of u by v, until he reaches 0, the previous integer is the GCD.

A first improvement was proposed in 1938 by D.E. Lehmer [17]. He observed that the first list of the quotients can be obtained only by considering the most significant bits of the input integers.

In 1961, J. Stein [25] (see also Knuth [13]) proposed the binary algorithm, better suited to the first computers, since it was based on simple operations such as subtractions and right-shifts of bits, avoiding divisions with large integers. Gosper suggested later a multi-precision version of Stein binary algorithm (see Knuth [13] pp. 339 and 599, exercise 34).

In the last decades, much work have been done to improve the performance of GCD algorithms. In 1970, Knuth [14] proposed the first subquadratic GCD algorithm which can be achieved in $O(n \log^5 n \log\log n)$ time, where n is the number of bits of the larger input integer. Schönhage [22] improved this time complexity in 1971 since his GCD algorithm can be achieved in $O(n \log^2 n \log\log n)$ time, which is, until now, the fastest sequential GCD algorithm. Many fast sequential GCD algorithms reached this performance with similar divide and conquer approach [24, 18, 20, 21].

In this paper, we consider another gcd algorithm based on subtraction, defined by the sequence $u_{n+2} = |u_{n+1} - u_n|$ where $u_0, u_1 \geq 1$ are two integers. The sequence stops at the first zero, the previous non zero integer is the gcd of u and

[1] Some authors (see [3]) think that this algorithm was known two centuries before.

Authors	Year	Worst-case complexity
Euclid	~ 300 BC	$O(n^2)$
Lehmer	1938	$O(n^2)$
Stein	1961	$O(n^2)$
Knuth	1970	$O(\log^4 n\, M(n))$
Schönhage	1971	$O(\log n\, M(n))$
Brent-Kung	1983	$O(n^2)$
Jebelean-Weber	1993	$O(n^2)$
Sorenson (Left & Right shift)	1994	$O(n^2/\log n)$
Stehlé-Zimmermann	2004	$O(\log n\, M(n))$
Möhler	2008	$O(\log n\, M(n))$

Table 1: Some sequential GCD algorithms.

v. For example, if $(u, v) = (18, 42)$, then we obtain, in 7 steps the sequence $(18, 42), (42, 24), (24, 18), (18, 6), (6, 12), (12, 6), (6, 6), (6, 0)$.

This procedure is similar to an early GCD algorithm, described by Euclid, more than two thousand years ago. See for example the papers *Subtractive Algorithm for GCD* of D.E. Knuth and A.C. Yao in [15], pages 195-204 and [16] for the average analysis of this algorithm. In fact the sequence in Euclid alogorithm is defined by $(u, v) \longrightarrow (|u - v|, \min\{u, v\})$, while our algorithm is based on $(u, v) \longrightarrow (v, |u - v|)$. With the previous example $(u, v) = (18, 42)$, Euclid algorithm gives in 5 steps $(18, 42), (18, 24), (18, 6), (12, 6), (6, 6), (6, 0)$. The aim of this paper is to study the worst case analysis of this new algorithm we call the Sub-like gcd algorithm:

Input: $2 \leq a, b \leq 2^n$ with $n \geq 2$.
Output: $\gcd(a, b)$.
    While $b \neq 0$ **Do**
        $(a, b) \longleftarrow (b, |a - b|)$;
    Endwhile
    Return a.

The Sub-like gcd algorithm.

2 Notations and definitions

Let $n \geq 2$ and u_0, u_1 be two integers such that $2 \leq u_0, u_1 < 2^n$. We define the sequence $(u_k)_k$ by $u_{k+2} = |u_{k+1} - u_k|$, for all $k \geq 0$. The sequence $(u_k)_k$ stops at the first $u_t = 0$, i.e.: $t = \min\{k\,,\ \text{s.t.}:\ u_k = 0\}$.

• We define some related sequences $(V_k)_k$ and $(w_k)_k$. Let $V_k = (u_{3k}, u_{3k+1}, u_{3k+2})$ for $0 \leq k \leq p$, where p is the maximum number of triplets V_k, except V_0, so that $(V_k)_k = (V_0, V_1, \cdots, V_p)$.
• Let $w_k = \max\{u_{3k}, u_{3k+1}, u_{3k+2}\} = \max\{u_{3k}, u_{3k+1}\}$, $0 \leq k \leq p$.

Cases	$\max(a,b)$	$x = \lvert b - \lvert a-b\rvert\rvert$	$y = \lvert x - \lvert a-b\rvert\rvert$	$\max(x,y)$	$w_0 - w_1$
$\frac{2a}{3} \le b < a$	a	$2b-a$	$3b-2a$	$2b-a$	$2a-2b$
$\frac{3a}{5} \le b < \frac{2a}{3}$	a	$2b-a$	$2a-3b$	$2b-a$	$2a-2b$
$\frac{a}{2} \le b < \frac{3a}{5}$	a	$2b-a$	$2a-3b$	$2a-3b$	$3b-a$
$\frac{a}{3} \le b < \frac{a}{2}$	a	$a-2b$	b	b	$a-b$
$b < \frac{a}{3}$	a	$a-2b$	b	$a-2b$	$2b$
$a < b \le 2a$	b	a	$2a-b$	a	$b-a$
$2a < b \le 3a$	b	a	$b-2a$	a	$b-a$
$b > 3a$	b	a	$b-2a$	$b-2a$	$2a$

Table 2: The computation of $w_0 - w_1 = \max\{a,b\} - \max\{x,y\}$.

• Let $(u_0,u_1) = (a,b)$ so that $V_0 = (a,b,|a-b|)$, with $a,b \ge 2$, and $w_0 = \max\{a,b\}$.
• Let $d = \gcd(u_0,u_1)$. The sequences $(u'_k)_k$ starting with $(u_0/d, u_1/d)$ and $(u_k)_k$ will stop both at the same index t. So, WLOG, we assume $\gcd(u_0,u_1) = 1$ and $u_{t-1} = u_{t-2} = 1$.
• Moreover, from the definition of $(u_k)_k$, we can easily prove that:

$$(u_k)_k = (u_0,u_1,\cdots,1,2,1,1,0) \quad \text{or} \quad (u_k)_k = (u_0,u_1,\cdots,3,2,1,1,0)\,.$$

• This shows that $V_p = (1,1,0)$ or $V_p = (2,1,1)$ or $V_p = (x,2,1)$ with $x = 1$ or 3 and $w_p \in \{1,2,3\}$.
• C and t_0 are defined by $C = \sum_{j=0}^{p-1}(w_j - w_{j+1}) = w_0 - w_p$ and $t_0 = 3 \times 2^{n-1}$.
• Table 2 describes the different values of (u_2,u_3,u_4), starting from $(u_0,u_1) = (a,b)$. We denote $x = u_3 = |b - |a-b|\,|$ and $y = u_4 = |x - |a-b|\,|$ so that $(u_2,u_3,u_4) = (|a-b|,x,y)$.

3 The longest sequence $(u_k)_k$:

The lenght t of the sequence $(u_k)_k$ depends on the pair of integers u_0,u_1, so that $t = t(u_0,u_1)$. The aim of this Section is to find the longest sequence $(u_k)_k$, w.r.t. the input integers $2 \le u_0,u_1 < 2^n$, for a fixed $n \ge 2$. In other words, for a fixed $n \ge 2$, if we define $\mathcal{U}_n = \{(u_k)_k \ \ s.t. : 2 \le u_0,u_1 < 2^n\}$, then the aim is to find u_0,u_1 and t such that

$$t = \max\{t = t(u_0,u_1)\ ;\ u \in \mathcal{U}_n\}\,.$$

This also corresponds to the worst-case of the Sub-like gcd algorithm. We have the following results:

Lemma 1: Let $(u_0,u_1) = (a,b)$ are such that u_5 exists (so that w_1 exists). Let $w_k = \max\{u_{3k},u_{3k+1},u_{3k+2}\}$, for $k \ge 0$. Then:

$$i) \quad (w_0 - w_1 = 1) \Longleftrightarrow (b = a+1)$$

$$ii) \quad w_k - w_{k+1} \ge 1\,.$$

Proof : First, thanks to Table 2, we can easily check that u_5 exists if $a \notin \{b/2,\ b,\ 3b/2,\ 2b\}$. Note that this condition yields $a, b \geq 2$ and $a \neq b$.
i) ($\Longleftarrow$) If $b = a+1$, then $V_0 = (a, a+1, 1)$, $V_1 = (a, a-1, 1)$, so $w_0 = a+1$, $w_1 = a$ and $w_0 - w_1 = 1$.
($\Longrightarrow$) We assume that $w_0 - w_1 = 1$. Recall that $a, b \geq 2$. Table 2 describes the different values of $w_0 - w_1$ with respect to a and b. Since $w_0 - w_1$ is odd, then the only cases we have to consider are $(3), (4), (6)$ and (7).

• Case (3): We have $a/2 \leq b < 3a/5$ and $w_0 - w_1 = 3b - a = 1$. So $a = 3b - 1$ which is impossible since $a \leq 2b$.

• Case (4): We have $a/3 \leq b < a/2$ and $w_0 - w_1 = a - b = 1$. So $a = b + 1$ which is impossible since $a > 2b$.

• Case (6): We have $a < b \leq 2a$ and $w_0 - w_1 = b - a = 1$. So $b = a + 1$ which is possible since $a < b = a + 1 \leq 2a$.

• Case (7): We have $2a < b \leq 3a$ and $w_0 - w_1 = b - a = 1$. So $b = a+1$ which is impossible since $b > 2a$.

So the only case where $w_0 - w_1 = 1$ is the case (6) when $b = a + 1$ and we have $V_0 = (a, a+1, 1)$, $V_1 = (a, a-1, 1)$, $w_0 = a+1$, $w_1 = a$ and $w_0 - w_1 = 1$.
ii) Just consider $(a, b) = (u_k, u_{k+1})$. The result follows from Table 2 because $a \neq b$ and $a, b \geq 2$ as noticed at the begining of the proof. Perhaps, the only not obvious case, is the Case (3). However, if $a/2 \leq b < 3a/5$ then $w_0 - w_1 = 3b - a \geq a/2 \geq 1$. Note that *ii*) proves that there exists an index t such that $u_t = 0$ and the terminaison of the SUB-LIKE GCD algorithm. □

Lemma 2 : Let $V_k = (u_{3k}, u_{3k+1}, u_{3k+2})$, for $k \geq 0$. Let t be the first index such that $u_t = 0$. If the sequence $(u)_k$ starts with $(u_0, u_1) = (a, a+1)$, then,

$$i)\ \ (1 \leq j \leq a/2\,) \Longrightarrow\ \ (\,V_j = (a - 2j + 2, a - 2j + 1, 1)\ \text{ and }\ w_j - w_{j+1} = 2\,).$$

$$ii) \qquad t = t(a, a+1) = 3\lfloor a/2 \rfloor + 3\,.$$

Proof : *i*) If $w_0 - w_1 = 1$ then, Lemma 1 yields $V_0 = (a, a+1, 1)$ so $V_1 = (a, a-1, 1)$ and $w_1 = a$. If $a \geq 4$, then $V_2 = (a-2, a-3, 1)$ and $w_2 = a - 2$. So $w_1 - w_2 = 2$. Moreover, as long as a is large enough we have in turn $V_3 = (a-4, a-5, 1)$, $V_4 = (a-6, a-7, 1)$ so $w_2 - w_3 = w_3 - w_4 = 2$. By induction on j, we easily prove that

$$1 \leq j \leq a/2\,, \quad V_j = (a - 2j + 2, a - 2j + 1, 1) \quad \text{and} \quad w_j = a - 2j + 2\,.$$

and obviously $w_j - w_{j+1} = 2$.
ii) If a is even then the last triplet V_p is obtained for the largest j, i.e.: $j = p = a/2$ and $V_p = (2, 1, 1)$. So $(u_k)_k = (a, a+1, 1)[V_1][V_2] \cdots [V_p], 0)$. Since $u_t = 0$ then
$V_p = (u_{3p}, u_{3p+1}, u_{3p+2}) = (u_{t-3}, u_{t-2}, u_{t-1}) = (2, 1, 1)$ and $t = 3p + 3 = (3a/2) + 3$.
Similarly, the case a odd yields $p = (a-1)/2$, $V_p = (3, 2, 1)$ and $t = 3p + 4 = (3a+5)/2 = 3\lfloor a/2 \rfloor + 3$. □

Lemma 3 : Let $k \geq 0$. If $a \geq 2k + 2$ and $V_{k+1} = (a, a+1, 1)$ then
(1) $V_k = (5a+2, 3a+1, 2a+1)$ or $V_k = (a, 3a+1, 2a+1)$ or $V_k = (3a+2, a+1, 2a+1)$.

(2) $w_k - w_{k+1} \geq 2a \geq 4k + 4$.

Proof: Let $V_k = (\alpha, \beta, \gamma)$, with $\alpha, \beta, \gamma \geq 1$. Then from the formula $u_{k+2} = |u_{k+1} - u_k|$, we must have $|\gamma - a| = a + 1$, so $\gamma = 2a + 1$. Moreover $|\beta - \gamma| = |\beta - (2a + 1)| = a$, so $\beta = a + 1$ or $\beta = 3a + 1$. $\gamma = |\alpha - \beta|$ is obtained similarly by considering the two values of β. Hence (1). The result (2) is obvious since $w_{k+1} = a + 1$ and $w_k \geq 3a + 1$. □

Consequently from Lemma 1 and 3, we derive an important result:

Corollary 1: Let $k \geq 1$, then

$$\forall a \geq 2k\,, \quad (\, w_k - w_{k+1} = 1\,) \implies (\, w_{k-1} - w_k \geq 4 \text{ and } w_j - w_{j+1} = 2\,,\ \forall j \geq k + 1\,).$$

Example: For $(u_0, u_1) = (2, 7)$, we obtain in turn $V_0 = (2, 7, 5)$, $w_0 = 7$, $V_1 = (2, 3, 1)$, $w_1 = 3$ and $V_2 = (2, 1, 1)$, $w_2 = 2$, with $k = 1$ and $a = 2$. So $w_1 - w_2 = 1$ and $w_0 - w_1 = 4$.

Lemma 4: Let (u_k) be the sequence defined by $(u_0, u_1) = (a, b)$, $2 \leq a, b < 2^n$, $n \geq 3$ and $u_{k+2} = |u_{k+1} - u_k|$ for $0 \leq k \leq t - 2$. Let $C = w_0 - w_p$. Then there are three cases:
Case (I): If there exists $i \geq 1$ such $w_i - w_{i+1} = 1$ then $p \leq (C - 1)/2$.
Case (II): If $\forall\, j$, $0 \leq j \leq p - 1$, $w_j - w_{j+1} \geq 2$ then $p \leq C/2$.
Case (III): If $w_0 - w_1 = 1$ then $p = (C + 1)/2$.

Proof:
Case (II): For all $0 \leq j \leq p - 1$, $w_j - w_{j+1} \geq 2$. So $C = \sum_{j=0}^{p-1} (w_j - w_{j+1}) \geq 2p$.
Case (I): There exists $i \geq 1$ such that $w_i - w_{i+1} = 1$. By Corollary 1 and Lemma 2, $w_{i-1} - w_i \geq 4$ so $(w_{i-1} - w_i) + (w_i - w_{i+1}) \geq 5$ and $C = \sum_{j=0}^{p-1} (w_j - w_{j+1}) \geq 5 + 2(p - 2) = 2p + 1$. So $p \leq (C - 1)/2$.
Case (III): $w_0 - w_1 = 1$, so by Lemma 2, we have $C = \sum_{j=0}^{p-1} (w_j - w_{j+1}) = 1 + 2(p - 1) = 2p - 1$, so $p = (C+1)/2$. Hence the result. □

Theorem 1: Let (u_k) be the sequence defined by $(u_0, u_1) = (a, b)$, $2 \leq a, b < 2^n$, $n \geq 3$ and all $k \leq 0$, $u_{k+2} = |u_{k+1} - u_k|$. If $t = t(n)$ is the first index for which $u_t = 0$, then $t \leq 3 \times 2^{n-1}$. Moreover, this bound is reached when $(u_0, u_1) = (2^{n-1} - 2, 2^{n-1} - 1)$, in this case $t = t_0 = 3 \times 2^{n-1}$.

Proof: Of course, the largest sequence (u_k) is obtained when $\gcd(u_0, u_1) = 1$ so that the three last integers of the sequence (u_k) are $(u_{t-2}, u_{t-1}, u_t) = (1, 1, 0)$. Consider the sequences V_k and w_k, with $0 \leq k \leq p$. Recall the w_p is equal to 1, 2 or 3 (see the notation Section) and $C = \sum_{j=0}^{p-1} (w_j - w_{j+1}) = w_0 - w_p$, where $w_0 = \max\{u_0, u_1\}$ and $w_p = \max(V_p) = \max\{u_{3p}, u_{3p+1}, u_{3p+2}\}$.

First, when $(u_0, u_1) = (a, a + 1)$, we have $w_0 - w_1 = 1$ (case (III) of Lemma 4), Lemma 2 shows that $t = 3\lfloor a/2 \rfloor + 3$. When $a = 2^n - 2$, then $t = t_0 = 3 \times 2^{n-1}$. It remains to prove that in both cases (I) and (II) of Lemma 4, we have $t < t_0$. Lemma 4 shows that, for both cases (I) and (II), we have $p \leq C/2$. However w_p, C and t depends on V_p. The are three cases for V_p:

* If $V_p = (2,1,1)$: In this case $(u_k)_k = (u_0, u_1, \cdots, 2, 1, 1, 0)$, so that $V_p = (u_{3p}, u_{3p+1}, u_{3p+2}) = (u_{t-3}, u_{t-2}, u_{t-1}) = (2,1,1)$, $t = 3p+3$, $w_0 = \max\{u_0, u_1\} \leq 2^n - 1$, $w_p = 2$, $C = w_0 - w_p \leq 2^n - 3$. So $p \leq C/2 \leq 2^{n-1} - 3/2$ and $t = 3p+3 \leq 3(C/2) + 3 \leq 3 \times 2^{n-1} - 3/2 < t_0$.

* If $V_p = (1,1,0)$: In this case $(u_k)_k = (u_0, u_1, \cdots, 2, 1, 1, 0)$, so that $V_p = (u_{3p}, u_{3p+1}, u_{3p+2}) = (u_{t-2}, u_{t-1}, u_t) = (1,1,0)$, $t = 3p+2$, $w_p = 1$, $w_0 = \max\{u_0, u_1\} \leq 2^n - 1$ and $C = w_0 - w_p \leq 2^n - 2$. So $p \leq C/2 \leq 2^{n-1} - 1$ and $t = 3p+2 \leq 3(C/2) + 2 \leq 3 \times 2^{n-1} - 1 < t_0$.

* If $V_p = (x,2,1)$, with $x = 1$ or $x = 3$: In this case $(u_k)_k = (u_0, u_1, \cdots, x, 2, 1, 1, 0)$, so that $V_p = (u_{3p}, u_{3p+1}, u_{3p+2}) = (u_{t-4}, u_{t-3}, u_{t-2}) = (x,2,1)$. Then $t = 3p+4$, $w_p \geq 2$, $w_0 = \max\{u_0, u_1\} \leq 2^n - 1$ and $C = w_0 - w_p \leq 2^n - 3$. So $p \leq C/2 \leq 2^{n-1} - 3/2$ and $t = 3p+4 \leq 3(C/2) + 4 \leq 3 \times 2^{n-1} - 1/2 < t_0$. □

References

[1] E. Bach, J.O. Shallit, *Algorithmic Number Theory*, Vol.1 (MIT Press, 1996).

[2] W.A Blankinship, A new version of Euclidean algorithm, *Amer. Math. Monthly,* 70 (1963) 742-745.

[3] R.P. Brent, Twenty years analysis of the binary Euclidean algorithm. In A. W. Roscoe, J. Davies and J. Woodcock, editors, *Millenial Perspectives in Computer Science: Proceedings of the 1999 Oxford-Microsoft Symposium in honor of Professor Sir Antony Hoare,* Palgrave, New York, (2000) 41–53.

[4] R.P. Brent and H.T. Kung, Systolic VLSI arrays for linear-time GCD computation, in: Anceau and Aas eds., *Proceedings of VLSI'83* (1983) 145–154.

[5] G. Lejeune Dirichlet, *Abhandlungen Königlich Preuβ. Akad. Wiss.* (1849) 69–83.

[6] Euclid, Book VII. For an English version see Knuth [13].

[7] Euclid Elementa, edited by J.L. Heiberg and E.S. Stamatis, vols, Teubner, Leipzig, 1969-77.

[8] Euclid, The Thirteen Books of Euclid's Elements, translated with introduction and commentary by T.L. Heath, Dover 1956.

[9] J. von zur Gathen and J. Gerhard, Modern Computer Algebra, *Cambridge University Press,* 1st ed., (1999).

[10] T. Jebelean, A Generalization of the Binary GCD Algorithm, *in Proceedings of the International Symposium on Symbolic and Algebraic Computation ISSAC'93* (1993), 111–116.

[11] R. Karp, V. Ramachandran, *Parallel Algorithms for Shared-memory Machines* in J.Van Leeuwen Editor, Algorithms and Complexity, (Elsevier and MIT Press, 1990, Handbook of Theoretical Computer Science) **Vol. A**.

[12] A. Klappenecker, *Lecture notes CPSC 629,* Analysis of Algorithms, available in http://faculty.cse.tamu.edu/ klappi.

[13] D.E. Knuth, *The Art of Computer Programming,* Addison Wesley, 3rd ed., Vol. 2 (1998).

[14] D.E. Knuth, The Analysis of Algorithms, *in Actes du Congrés International des Mathématiciens,* Gauthiers-Villars ed., Nice, France, (1970) 269–274.

[15] D.E. Knuth, A.C. Yao, *Selected Papers on Analysis of Algorithms* (CSLI Lecture notes, No. 102, Stanford) (2000).

[16] A.C. Yao, D.E. Knuth, *Ananlysis of the subtractive algorithm for greatest common divisors,* in Proc. Nat. Acad. Sci. USA, Vol. 72, (1975) 4720–4722.

[17] D.H. Lehmer, Euclid's algorithm for large numbers, *American Math. Monthly* **45** (1938) 227-233.

[18] D. Lichtblau, Half-GCD and Fast rational recovery, *in Proc. of the International Symposium on Symbolic and Algebraic Computation, (ISSAC'2005)* (2005) 254–258.

[19] A.J. Manezes, P.C. van Oorschot and S.A. Vanstone, *Handbook of Applied Cryptography*, Vol. 1-2, 2nd ed., CRC Press, (1997).

[20] N. Möller, On Schönhage's Algorithm and Subquadratic Integer GCD Computation, *Mathematic of Computation*, Vol. 77, 261, (2008) 589–607.

[21] V.Y. Pan and X. Wang, Acceleration of Euclidean Algorithm and extensions, *in Proc. of the International Symposium on Symbolic and Algebraic Computation, ISSAC'2002,* (2002) 207–213.

[22] A. Schönhage, Schnelle Berechnung von Kettenbruchentwicklugen, *Acta Informatica* (1971), 1, 139-144.

[23] J.P. Sorenson, Two Fast GCD Algorithms, *J. of Algorithms* **16** (1994) 110–144.

[24] D. Stehlé, P. Zimmermann, A Binary Recursive Gcd Algorithm, *in Proc. of ANTS VI, University of Vermont, USA,* June 13-18, 2004, 411-425.

[25] J. Stein, Computational Problems Associated with Racca Algebra, *J. Comp. Phys.,* Vol. 1, (1967) 397–405.