On HGCD-D bounds

Juraj Sukop, Niels Möller

On HGCD-D bounds

Juraj Sukop [1], Niels Möller [2]

[1] sukop@xxyxyz.org , [2] nisse@lysator.liu.se

Abstract

An improved bound for one of the founding relations of HGCD-D algorithm is presented. This allows to put a lower limit on the iteration count of the first sdiv loop, to impose a particular structure on the accumulated quotients and to bound the size of the largest matrix element. The matrix product $M \cdot M'$ is proved to have its upper and lower size bound differ by at most two bits.

Keywords: Euclid's algorithm, greatest common division, GCD, Half-GCD

1 Introduction

The present note introduces several bounds relevant to HGCD-D algorithm and its implementation. The algorithm was first introduced in [1] by Niels Möller, where it was also analysed in detail and compared to other asymptotically fast half-GCD functions. Half-GCD can be seen as GCD algorithm interrupted in the middle of the work: it returns a matrix that can be used to compute two consecutive terms of the remainder sequence, each being of roughly half the size of the given input. In order not to repeat the many facts proved therein, this note is structured as a commentary to the original paper and as such only the most relevant results for this work will be recapitulated.

The motivation for this note is the matrix multiplication of HGCD-D algorithm depicted in Listing 1, line 23. In particular, the lower bound of the size that the product could attain as this has implications for a practical implementation with regards to memory allocation. Note that the upper bound is trivial: in general, the size of the product of two matrices is bounded by the sum of their respective sizes plus one. The worry then is just how much smaller the final size could be.

The main argument of this note can be summarized as follows: By improving one of the stated relations it is shown that for the first loop at least one sdiv step must be made. Such a fact has consequences for the structure of the quotients, the relative order in which the factor matrices representing the quotients accumulate and the size of matrix elements. Finally, the lower bound on the the maximal element of the matrix product is proved.

2 Preliminaries

Through the text we will keep using the same notation as in the original: $\#x$ denotes the bit size of positive x, $\#(x, y) = \max(\#x, \#y)$, $\#(x, x'; y, y') =$

$\max(\#x, \#x', \#y, \#y')$, $\underline{\#}(x, y) = \min(\#x, \#y)$. A, B are positive integers (the input), M is the transformation matrix of the accumulated quotient sequence.

HGCD-D accepts integers $a, b > 0$, $n = \#(a, b)$, $s = \lfloor n/2 \rfloor + 1$, $\underline{\#}(a, b) > s$ and returns integers $\alpha, \beta > 0$, $\underline{\#}(\alpha, \beta) > s$, $\#(\alpha - \beta) \leq s$ and matrix $M \geq 0$, $\det(M) = 1$, $(a; b) = M(\alpha; \beta)$.

Nevertheless, here we will deviate slightly from the presentation in the original paper in order to amend one inaccuracy. There the first sdiv loop reads as

```
 9    while #(A, B) > ⌊3N/4⌋ + 1 and #(A − B) > S
10      do
11        One sdiv step on (A, B); update M
```

and later the paper states "The bound $N_2 \leq \lfloor 3N/4 \rfloor + 1$ implies ...". However, the condition $\#(A-B) > S$ means the loop can terminate at that point as well and thus A, B may not get the chance to become less or equal to $\lfloor 3N/4 \rfloor + 1$. Moreover, for the sake of simplified analysis, it is also advantageous to terminate the algorithm early whenever $\#(A - B) \leq S$ as no further progress is possible, anyway. Therefore we write the algorithm as follows

```
    HGCD-D(A, B)
 1    N ← #(A, B), S ← ⌊N/2⌋ + 1
 2    if underline#(A, B) > ⌊3N/4⌋ + 2
 3      then
 4        p1 ← ⌊N/2⌋, n1 ← N − p1 = ⌈N/2⌉
 5        Split: A = 2^p1 a + A', B = 2^p1 b + B'
 6        (α, β, M) ← HGCD-D(a, b)
 7        (A; B) ← 2^p1 (α; β) + M^−1 (A'; B')
 8      else M ← I
 9    loop
10      do
11        if #(A − B) ≤ S
12          then return A, B, M
13        if #(A, B) ≤ ⌊3N/4⌋ + 1
14          then break
15        One sdiv step on (A, B); update M
16    if underline#(A, B) > S + 2
17      then
18        N2 ← #(A, B)
19        p2 ← 2S − N2 + 1, n2 ← N2 − p2
20        Split: A = 2^p2 a + A', B = 2^p2 b + B'
21        (α, β, M') ← HGCD-D(a, b)
22        (A; B) ← 2^p2 (α; β) + M'^−1 (A'; B')
23        M ← M · M'
24    while #(A − B) > S
25      do
26        One sdiv step on (A, B); update M
27    return A, B, M
```

Listing 1: The HGCD-D algorithm

3 The bounds

Lemma 1. *As in Lemma 6 of the original paper, let* $(C; D) = M^{-1}(A; B)$, $N = \#(A, B)$, $0 < p < N$, $n = N - p$. *Then* $\#(C, D) > p + \lfloor n/2 \rfloor + 1$

Proof. Recall the original Equation 4

$$\begin{pmatrix} C \\ D \end{pmatrix} = ... = 2^p \begin{pmatrix} c \\ d \end{pmatrix} + M^{-1} \begin{pmatrix} A' \\ B' \end{pmatrix}$$

where $A = 2^p a + A'$, $B = 2^p b + B'$ and $(c, d, M) = \text{HGCD-D}(a, b)$, $s = \lfloor n/2 \rfloor + 1$, $\underline{\#}(c, d) > s$. Next, consider its rightmost part where a', b' denote the last term

$$\begin{pmatrix} a' \\ b' \end{pmatrix} = M^{-1} \begin{pmatrix} A' \\ B' \end{pmatrix} \iff \begin{pmatrix} A' \\ B' \end{pmatrix} = M \begin{pmatrix} a' \\ b' \end{pmatrix}$$

Suppose $M = (u, u'; v, v')$ and $A', B' \geq 0$, $u, v' > 0$, $u', v \geq 0$ then only one of a', b' can be negative. Write

$$A' = ua' + u'b'$$
$$B' = va' + v'b'$$

If both a', b' were negative, A', B' would also need to be negative, a contradiction.

If $a' \geq 0$ (otherwise $b' \geq 0$ and the same reasoning applies for $\#D$)

$$C = 2^p c + a' \geq 2^p c \geq 2^p 2^s = 2^{p + \lfloor n/2 \rfloor + 1}$$

then

$$\#C > p + \lfloor n/2 \rfloor + 1$$

Previously Lemma 6 stated $\#C, \#D \geq p + \lfloor n/2 \rfloor + 1$ whereas now it has been shown that the equality can hold for at most one of $\#C, \#D$.

Lemma 2. *If* $\#(A - B) > S$, *the first loop iterates by at least one sdiv step.*

Proof. If $\#(A - B) \leq S$, HGCD-D can exit as no further progress is possible. Otherwise, by Lemma 1, the new A, B after the first recursive call satisfy

$$\#(A, B) > p_1 + \lfloor n_1/2 \rfloor + 1 = \lfloor N/2 \rfloor + \lfloor (N + 1)/4 \rfloor + 1 = \lfloor 3N/4 \rfloor + 1$$

Thus the condition of the second branch is never satisfied during the first iteration and at least one sdiv step is performed.

Lemma 3. *Provided HGCD-D does not terminate early, the quotients generated by the first sdiv loop that are accumulated into* M *are identical to the classical Euclidian reduction steps of* $\lfloor A/B \rfloor$ *or* $\lfloor B/A \rfloor$. *As a consequence a quotient* q *cannot be split into last factor of* M *and first factor of* M'.

Proof. Recall that sdiv operation "never returns a too small 'remainder'" and that it splits one from the quotient if needed. As $\#(A - B) > S$, all sdiv steps are identical to Euclid steps using standard division, by the definition of sdiv, and thus all of the quotients are not split.

Informally, whenever the difference between div and sdiv is being exercised, HGCD-D can exit early. Otherwise the (s)div step returns the "complete" quotient. So even if the first recursive call returned M with its last factor split into $q-1$ and 1 subtractions, the first (s)div step will "un-split" it, i.e. accumulate the single remaining subtraction into M.

Corollary 4. *If the last factor of M is $(1, q; 0, 1)$, then the first factor of M' is $(1, 0; q', 1)$ or M' is the identity and vice versa.*

Proof. Suppose $A > B$. As the last quotient q cannot be split, the next remainder $A - qB < B$ and then $a \leq b$ for the second recursive call. Same holds for $A < B$.

Theorem 5. $\#(M \cdot M') \geq \#M + \#M' - 1$

Proof. If neither matrix is the identity and the last factor of M is $(1, q; 0, 1)$ then

$$M = (u_1, u_1'; v_1, v_1')(1, 1; 0, 1) = (u_1, u_1 + u_1'; v_1, v_1 + v_1')$$
$$M' = (1, 0; 1, 1)(u_2, u_2'; v_2, v_2') = (u_2, u_2'; u_2 + v_2, u_2' + v_2')$$

Since all elements are non-negative, $\max(M) = \max(u_1 + u_1', v_1 + v_1')$ and $\max(M') = \max(u_2 + v_2, u_2' + v_2')$. Let

$$M \cdot M' = \begin{pmatrix} u_1u_2 + (u_1 + u_1')(u_2 + v_2) & u_1u_2' + (u_1 + u_1')(u_2' + v_2') \\ v_1u_2 + (v_1 + v_1')(u_2 + v_2) & v_1u_2' + (v_1 + v_1')(u_2' + v_2') \end{pmatrix} \tag{1}$$

and notice that the product exhaustively enumerates each of the four combinations formed by the two candidates for the maximal element of M and the two candidates of M'. Write

$$\begin{aligned} \max(M \cdot M') &\geq \max((u_1 + u_1')(u_2 + v_2), (u_1 + u_1')(u_2' + v_2'), \\ &\qquad (v_1 + v_1')(u_2 + v_2), (v_1 + v_1')(u_2' + v_2')) \\ &= \max(u_1 + u_1', v_1 + v_1') \max(u_2 + v_2, u_2' + v_2') \\ &= \max(M) \max(M') \end{aligned}$$

Seen in the terms of bit sizes

$$\max(M \cdot M') \geq 2^{\#M-1} 2^{\#M'-1} = 2^{\#M+\#M'-2}$$

and then

$$\#(M \cdot M') \geq \#M + \#M' - 1$$

Same holds if the last factor of M is $(1, 0; q, 1)$ and in the case of either matrix being the identity the bound applies trivially.

4 Conclusion

We have shown that tightening one of the bounds by a single bit has the consequence of having the matrix product size estimate off by at most two bits, i.e.

$$\#M + \#M' - 1 \leq \#(M \cdot M') \leq \#M + \#M' + 1$$

References

[1] Niels Möller. On Schönhage's algorithm and subquadratic integer GCD computation. *Mathematics of Computation*, 77:589–607, 2008.