Journal of Integer Sequences, Vol. 27 (2024),
Article 24.4.3

Empirical Verification of a Generalization of Goldbach's Conjecture

Zsófia Juhász
Dept. of Computer Algebra
Faculty of Informatics
Eötvös Loránd University
1117 Budapest
Hungary
jzsofia@inf.elte.hu

Máté Bartalos
bartmate@gmail.com

Péter Magyar and Gábor Farkas
Faculty of Informatics
Eötvös Loránd University Center Savaria
9700 Szombathely
Hungary
map115599@gmail.com
farkasg@inf.elte.hu

Abstract

We test Hardy and Littlewood's generalization (GGC) of Goldbach's and Lemoine's conjectures. According to GGC, for relatively prime positive integers m_1 and m_2, every sufficiently large integer n satisfying certain simple congruence criteria can be expressed as $n = m_1 p + m_2 q$ for some primes p and q. We check GGC up to 10^{12} for all (up to 10^{13} for some) relatively prime coefficients $m_1, m_2 \leq 40$, and present the largest counterexamples that cannot be obtained in this form. We verify Lemoine's conjecture up to a new record of 10^{13}. We compare the running times of four natural verification algorithms for all relatively prime $m_1 \le m_2 \le 40$. The algorithms seek to find either the p- or the q-minimal (m_1, m_2)-partitions of all numbers tested, by either a descending or an ascending search for the prime to be maximized or minimized, respectively, in the partitions. For all m_1, m_2 descending searches were faster than ascending ones. We provide a heuristic explanation. The relative speed of ascending [descending] searches for the p- and the q-minimal partitions, respectively, varied by m_1, m_2. Using the average of $p^*_{m_1,m_2}(n)$—the minimal p in all (m_1, m_2)-partitions of n—up to a sufficiently large threshold, we introduce two functions of m_1, m_2, which may help predict these rankings. Our predictions correspond well with actual rankings, and could inform new verification efforts. Numerical data are presented, including average and maximum values of $p^*_{m_1,m_2}(n)$ up to 10^9.

1 Introduction

One of the best known and longest standing open problems in number theory is due to Goldbach [8], who formulated his famous conjecture in 1742, in a letter to Euler. In modern form, the even (or strong) Goldbach conjecture states that every even number greater than 2 can be expressed as the sum of two primes. Search for a proof or disproof of this claim has fascinated generations of scholars and curious minds since.

Progress includes Lu's proof [26] that the number of even integers up to x which do not have Goldbach partitions is $O(x^{0.879})$. Chen showed that every sufficiently large even number is the sum of a prime and a semiprime (the product of at most two primes) [4]. In 2013 Helfgott provided a proof for the odd (weak or ternary) Goldbach conjecture—a weaker statement than the even Goldbach conjecture—asserting that every odd number greater than 5 is the sum of three primes [12, 13].

With a general proof out of reach, several efforts have targeted the verification of the even Goldbach conjecture (GC) empirically up to increasing limits [21, 23, 24, 9]. Oliveira e Silva et al. [20] achieved the current record of $4 \cdot 10^{18}$ in a large scale computational project in 2014. A Goldbach partition of an even number n is an expression $n = p+q$ where p and q are prime. The Goldbach partition of n containing the smallest value of p is called the minimal Goldbach partition of n, and $p(n)$ and $q(n)$ denote the corresponding values of p and q, respectively [9, 20]. Oliveira e Silva et al. [20] carried out the verification by segments of size 10^{12}, in each interval searching for the minimal Goldbach partitions of even numbers using an efficient sieve method. Subsequently, they handled outstanding values n individually by 'ascending search' for $p(n)$. For each interval to be tested they first generated primes—potential candidates for q—in a somewhat larger interval, using a cache-efficient modified segmented sieve of Eratosthenes.

The rate of growth of $p(n)$ is of some theoretical interest. Granville et al. [9] conjectured $p(n) = O(\log^2 n \log\log n)$. Granville also suggested two more precise, incompatible conjectures of the form $p(n) \le (C + o(1)) \log^2 n \log\log n$, where C is 'sharp' in the sense that C is the smallest constant with this property: one with $C = C_2^{-1} \approx 1.51478$ and the other one with $C = 2e^{-\gamma}C_2^{-1} \approx 1.70098$, where $C_2 \approx 0.66016$ is the twin prime constant and $\gamma \approx 0.57722$ is the Euler-constant [20]. Empirical comparison of the plausibility of these conjectures was inconclusive due to the requirement of data up to even higher limits [20].

In 1894 Lemoine [15] proposed a stronger version of the weak Goldbach conjecture, stating that every odd number $n > 5$ can be expressed as $n = p + 2q$ for some primes p and q [6, p. 424]. The highest threshold of verification of Lemoine's conjecture (LC) the authors have found claims of is 10^{10} [17].

In 1923 Hardy and Littlewood [11] introduced the following generalization (GGC) of the even Goldbach conjecture, also generalizing LC: for all relatively prime positive integers m_1 and m_2, every sufficiently large integer n satisfying certain simple congruence conditions can be expressed as $n = m_1p + m_2q$ for some primes p and q. It appears that this generalization is lesser known, and has not been studied again until 2017 [7]. Hence, current paper is the second one concerned with the verification of GGC in its general form. The authors [7] tested GGC up to 10^9 for each $m_1, m_2 \leq 25$ relatively prime, and provided the smallest values of n satisfying the conditions of GGC starting from which all integers $\leq 10^9$ also satisfying these can be (m_1, m_2)-partitioned.

We extend the scope and limit of verification of GGC to all coefficients $m_1, m_2 \leq 40$ relatively prime up to 10^{12} (up to 10^{13} for some m_1, m_2), and present the greatest values $n \leq 10^{12}$ satisfying the conditions of GGC which cannot be (m_1, m_2)-partitioned. The relatively small sizes of the largest counterexamples support GGC. We confirm LC up to a new record of 10^{13}. We applied four different natural verification algorithms in case of every pair $m_1 < m_2$. (For $m_1 = m_2 = 1$ we only have two different approaches.) We compare their speed for each m_1, m_2, provide heuristic explanations for their speed rankings, and seek predictions for the fastest one when testing up to large thresholds. In this paper we are not aiming to fully optimize the algorithms, but interested in comparing four natural approaches to testing. For each pair m_1, m_2, the fastest one can be further improved, and potentially combined with other—perhaps more efficient, e.g., sieving—methods for testing up to higher limits in the future.

After preliminaries, Section 3 describes the four algorithms. An (m_1, m_2)-partition of n is an expression $n = m_1p + m_2q$ where p and q are prime. We call the (m_1, m_2)-partition of n containing the smallest value of p [q] the p-minimal [q-minimal] (m_1, m_2)-partition of n. Searching for the minimal Goldbach partition at the verification of GC [20] has two analogues when checking GGC with $m_1 \neq m_2$: finding either the p- or the q-minimal (m_1, m_2)-partitions of numbers. In either case one can search in descending order for the prime to be maximized or in ascending order for the prime to be minimized in the partitions. These considerations yield four approaches to testing. We also present some findings about the functions $p^*_{m_1,m_2}(n)$—where $p^*_{m_1,m_2}(n)$ is the smallest value of p in all (m_1, m_2)-partitions of n— and about the largest numbers $\hat{k}_{m_1,m_2}$ found satisfying the conditions of GGC that cannot be (m_1, m_2)-partitioned, which are relevant to the designs of the algorithms.

Section 4 provides information about the implementation of the algorithms and our measures to check the correctness of our computations.

Section 5 discusses the results regarding the speed ranking of the four algorithms for each pair $m_1 \le m_2 \le 40$ relatively prime—presented in Section 7— with some heuristic explanations by the first author. Since primes among larger numbers are scarcer on average, one may hypothesize that descending search for the prime to be maximized in the partition is faster than ascending search for the prime to be minimized. This is fully supported by our data. According to the results, whether descending [ascending] search for the p- or for the q-minimal partitions is faster depends on the pair m_1, m_2. We propose two hypotheses to predict these rankings, using two functions of m_1, m_2 and of the average of $p^*_{m_1,m_2}(n)$ taken up to a sufficiently large threshold. Predicted and actual rankings show reasonably good match. Approximations for the functions $p^*_{m_1,m_2}(n)$ would help estimate the time complexities of the algorithms, and ascertain the plausibility of the hypotheses.

Section 6 outlines our conclusions and some questions for future work.

Section 7 contains a subset of the data generated, including the largest value $n \le 10^{12}$ satisfying the conditions of GGC that cannot be (m_1, m_2)-partitioned for all $m_1, m_2 \le 40$, and the maximum and average values of $p^*_{m_1,m_2}(n)$ when $n \le 10^9$ for all relatively prime $m_1, m_2 \le 20$. We show the actual speed rankings of the four algorithms and those predicted by our hypotheses, for all relatively prime $m_1 < m_2 \le 40$.

Section 8 includes the pseudocode of the main program implementing one of the algorithms.

2 Preliminaries

For every integer a and b, let $\gcd(a, b)$ denote the greatest common divisor of a and b. Hardy and Littlewood [11] introduced the following conjecture:

Conjecture 1. Let m_1 and m_2 be positive integers such that $\gcd(m_1, m_2) = 1$. Then for every sufficiently large integer n satisfying the conditions

1. $\gcd(n, m_1) = \gcd(n, m_2) = 1$ and
2. $n \equiv m_1 + m_2 \pmod 2$,

there exist primes p and q such that

$$n = m_1 p + m_2 q. \tag{1}$$

Furthermore, they also conjectured the following estimate for the number of ways $N(n)$ in which an integer n satisfying the conditions of GGC can be expressed in the form 1:

$$N(n) \sim \frac{2C_2}{m_1 m_2} \frac{n}{(\log n)^2} \prod \left(\frac{p-1}{p-2} \right),$$

where C_2 is the twin prime constant, and the product is taken over all odd primes p which divide m_1, m_2 or n.

We let GGC_{m_1,m_2} denote the claim of GGC for given coefficients m_1, m_2. Then $\mathrm{GGC}_{1,1}$ and $\mathrm{GGC}_{1,2}$ are Goldbach's and Lemoine's conjectures, respectively. It is easy to see [7] that GGC is equivalent to the following:

Conjecture 2. Let m_1 and m_2 be positive integers. Then for every sufficiently large integer n satisfying the conditions

1. $\gcd(n, m_1) = \gcd(n, m_2) = \gcd(m_1, m_2)$ and
2. $n \equiv m_1 + m_2 \pmod{2^{s+1}}$, where 2^s is the largest power of 2 that is a common divisor of m_1 and m_2,

there exist primes p and q such that

$$n = m_1 p + m_2 q.$$

In the sequel we consider GGC. The letters n, m_1, and m_2 denote positive integers such that m_1 and m_2 are relatively prime.

Definition 3. An expression of the form 1 where p and q are primes is called an (m_1, m_2)-*Goldbach partition* (or (m_1, m_2)-*partition*) *of* n. We say that n *can be* (m_1, m_2)*-partitioned* if it possesses at least one (m_1, m_2)-partition.

For every m_1, m_2, the number $n = m_1 + m_2$ satisfies the conditions of GGC_{m_1,m_2} and cannot be (m_1, m_2)-partitioned. Hence, if GGC_{m_1,m_2} is true then there exists a largest positive integer satisfying the conditions of GGC_{m_1,m_2} that cannot be (m_1, m_2)-partitioned, which we denote by k_{m_1,m_2}. Let $\hat{k}_{m_1,m_2}$ stand for the largest integer $\leq 10^{12}$ satisfying the conditions of GGC_{m_1,m_2} that cannot be (m_1, m_2)-partitioned. We conjecture that $\hat{k}_{m_1,m_2} = k_{m_1,m_2}$ for every pair m_1, m_2 tested.

Definition 4. If n can be (m_1, m_2)-partitioned then the smallest and the largest values of p $[q]$ in all (m_1, m_2)-partitions of n are denoted by $p^*_{m_1,m_2}(n)$ $[q^*_{m_1,m_2}(n)]$ and $p^{**}_{m_1,m_2}(n)$ $[q^{**}_{m_1,m_2}(n)]$, respectively. We call $n = m_1 p^*_{m_1,m_2}(n) + m_2 q^{**}_{m_1,m_2}(n)$ the *p-minimal* (or *q-maximal*) and $n = m_1 p^{**}_{m_1,m_2}(n) + m_2 q^*_{m_1,m_2}(n)$ the *p-maximal* (or *q-minimal*) (m_1, m_2)-*partition of* n.

Clearly, for all m_1, m_2 the conditions of GGC_{m_1,m_2} and GGC_{m_2,m_1} on n are equivalent, and every (m_1, m_2)-partition of n is also an (m_2, m_1)-partition if the order of terms is disregarded. Hence, a number n can be (m_1, m_2)-partitioned if and only if it can be (m_2, m_1)-partitioned, and in this case $p^*_{m_1,m_2}(n) = q^*_{m_2,m_1}(n)$ and $p^{**}_{m_1,m_2}(n) = q^{**}_{m_2,m_1}(n)$. Also, we have $\hat{k}_{m_1,m_2} = \hat{k}_{m_2,m_1}$. Conjectures GGC_{m_1,m_2} and GGC_{m_2,m_1} are equivalent, and if they hold, then $k_{m_1,m_2} = k_{m_2,m_1}$.

2.1 Notation

In the sequel p_i denotes the i^{th} prime number ($i \in \mathbb{N}^+$), e.g., we have $p_1 = 2, p_2 = 3$, etc. For every n, the value $\varphi(n)$ of Euler's totient function at n is the number of positive integers less than or equal to n that are relatively prime to n. For given m_1 and m_2, we let $\mathrm{lcm}_{\mathrm{m_1,m_2}}$ denote the least common multiple of m_1, m_2, and 2. For every $L > \hat{k}_{m_1,m_2}$ for which there is at least one n satisfying the conditions of GGC_{m_1,m_2} such that $\hat{k}_{m_1,m_2} < n \leq L$, we refer to the average and the maximum values of $p^*_{m_1,m_2}(n)$ over all $\hat{k}_{m_1,m_2} < n \leq L$ satisfying the conditions of GGC_{m_1,m_2} more succinctly as the *average* and *maximum*, respectively, *of* $p^*_{m_1,m_2}$ *up to* L. For every integer a and $m \neq 0$, the modulo m residue of a is denoted by $a \bmod m$.

3 Verifying algorithms

In this section we describe the four algorithms applied for checking GGC_{m_1,m_2} up to $N_{m_1,m_2} \approx 10^{12}$ for every pair $m_1 \leq m_2 \leq 40$ relatively prime. (This means 490 different pairs $m_1 \leq m_2$.) We also present some results about the functions $p^*_{m_1,m_2}(n)$ and the values $\hat{k}_{m_1,m_2}$.

3.1 Input, output, and some notes on $p^*_{m_1,m_2}(n)$ and $\hat{k}_{m_1,m_2}$

3.1.1 Input and output

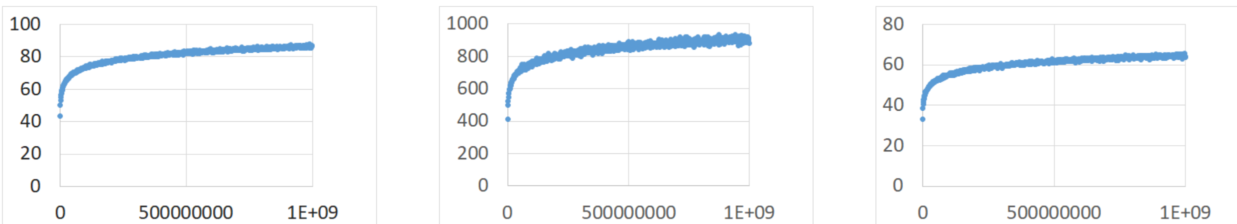
All algorithms verify GGC_{m_1,m_2} in a segmented fashion. The input are m_1 and m_2 relatively prime, the threshold of verification N, the length $\triangle$ of the segments to be checked at a time, and a further, implementation dependent parameter α. These can be set as required—subject to the constraints on the input provided in the outline of the algorithms—giving flexibility to our codes. We chose N to be the smallest multiple of $2m_1m_2$ greater than or equal to 10^{12}—denoted by N_{m_1,m_2}—and $\triangle$ to be the smallest multiple of $2m_1m_2$ greater than or equal to $5 \cdot 10^7$. (Assuming N and $\triangle$ are divisible by $2m_1m_2$ slightly simplified our code at parts.)

For every n satisfying the conditions of GGC_{m_1,m_2} the algorithms only check if n has an (m_1, m_2)-partition $n = m_1p + m_2q$ such that $m_1p \leq \alpha$ (or $m_2q \leq \alpha$). The output is the array residual containing those $n \leq N_{m_1,m_2}$ satisfying the conditions of GGC_{m_1,m_2} which do not possess such a partition. After an algorithm has finished, it remains to check by another method if numbers in residual can be (m_1, m_2)-partitioned.

3.1.2 Functions $p^*_{m_1,m_2}(n)$ and the choice of α

We aimed to set the value of α so that residual only contains numbers that cannot be (m_1, m_2)-partitioned at all, by ensuring that $m_1p^*_{m_1,m_2}(n) \leq \alpha$ holds for all relatively prime $m_1, m_2 \leq 40$ and $n \leq N_{m_1,m_2}$ satisfying the conditions of GGC_{m_1,m_2} that can be (m_1, m_2)-partitioned. We observed that $p^*_{m_1,m_2}(n)$ remains relatively small even for large values of n.

For example, Figure 1 demonstrates the slow growth of $p^*_{m_1,m_2}(n)$ by showing the average of $p^*_{m_1,m_2}(n)$ in each interval of length 10^6 centred at $x = 10^6k + 5 \cdot 10^5$ $(0 \leq k \leq 10^3 - 1)$ in the cases $m_1 = 1, m_2 = 2$ (Subfigure 1a), $m_1 = 4, m_2 = 17$ (Subfigure 1b), and $m_1 = 7, m_2 = 3$ (Subfigure 1c). Table 5 contains the maximum and average values of $p^*_{m_1,m_2}(n)$ up to $n \leq 10^9$ for each $m_1, m_2 \leq 20$ relatively prime. For $n \leq 10^9$, over all $m_1, m_2 \leq 40$ relatively prime the maximum of $p^*_{m_1,m_2}(n)$ is 78697 (at $m_1 = 32$, $m_2 = 37$), and the maximum of $m_1 p^*_{m_1,m_2}(n)$ is 2858879 (at $m_1 = 37$, $m_2 = 38$). Experimentally we also found that $m_1 p^*_{m_1,m_2}(n) \leq 5 \cdot 10^7$ for all $n \leq N_{m_1,m_2}$ satisfying the conditions of GGC_{m_1,m_2} that can be (m_1, m_2)-partitioned, for all $m_1, m_2 \leq 40$ relatively prime. Hence, in our implementation $\alpha = 5 \cdot 10^7$, and so for all m_1, m_2, the largest number in residual equals $\hat{k}_{m_1,m_2}$. Choosing smaller suitable α could have been possible, but the resulting improvements in running times would have been insignificant.



(a) $m_1 = 1, m_2 = 2$ (b) $m_1 = 4, m_2 = 17$ (c) $m_1 = 7, m_2 = 3$

Figure 1: The average value of $p^*_{m_1,m_2}(n)$ in the interval of length 10^6 centred at $x = 10^6k + 5 \cdot 10^5$ for $0 \leq k \leq 10^3 - 1$, in cases of m_1, m_2 indicated under each subfigure.

3.1.3 Values of $\hat{k}_{m_1,m_2}$

Table 4 shows $\hat{k}_{m_1,m_2}$ for all relatively prime $m_1, m_2 \leq 40$. The maximum (at $m_1 = 32$, $m_2 = 37$) and average of $\hat{k}_{m_1,m_2}$ are 412987 and 52004.84, respectively. The relatively small sizes of $\hat{k}_{m_1,m_2}$ support GGC, and also meant that the extra time required for checking numbers in residual was negligible.

3.2 The algorithms

3.2.1 Different approaches to testing

The main difference between Algorithms 1a, 1b, 2a, and 2b lies in their methods for checking if a number can be (m_1, m_2)-partitioned. These—for given ordered pair (m_1, m_2)—are summarized below:

Algorithm 1a [1b]: 'Descending search for the prime to be maximized' in the partitions. Algorithm 1a [1b] searches for the p-minimal [q-minimal] (m_1, m_2)-partition $n = m_1 p^*_{m_1,m_2}(n) + m_2 q^{**}_{m_1,m_2}(n)$ [$n = m_1 p^{**}_{m_1,m_2}(n) + m_2 q^*_{m_1,m_2}(n)$] by trying all possible candidates q [p] for $q^{**}_{m_1,m_2}(n)$ [for $p^{**}_{m_1,m_2}(n)$] in decreasing order until it finds that $n - m_2 q = m_1 p$

$[n - m_1 p = m_2 q]$ for some prime p $[q]$.

Algorithm 2a [2b]: 'Ascending search for the prime to be minimized' in the partitions. Algorithm 2a [2b] searches for the p-minimal [q-minimal] (m_1, m_2)-partition $n = m_1 p^{*}_{m_1,m_2}(n) + m_2 q^{**}_{m_1,m_2}(n)$ $[n = m_1 p^{**}_{m_1,m_2}(n) + m_2 q^{*}_{m_1,m_2}(n)]$ by trying all possible candidates p $[q]$ for $p^{*}_{m_1,m_2}(n)$ [for $q^{*}_{m_1,m_2}(n)$] in increasing order until it finds that $n - m_1 p = m_2 q$ $[n - m_2 q = m_1 p]$ for some prime q $[p]$.

Algorithms 1a and 1b [2a and 2b] can be implemented by the same program by interchanging the values of m_1 and m_2. Hence, only Algorithms 1a and 2a are described in this section, referred to as Algorithms 1 and 2, respectively.

3.2.2 Simplified outlines of Algorithms 1 and 2

Input: $m_1, m_2, N, \triangle, \alpha \in \mathbb{N}^+$ such that $\gcd(m_1, m_2) = 1$, $N > 9$, $2m_1m_2|N$, $2m_1m_2|\triangle$, and $\alpha \leq \triangle$.
Output: array `residual` containing all numbers $n \leq N$ satisfying the conditions of GGC_{m_1,m_2} for which there are no primes p and q such that $n = m_1 p + m_2 q$ and $m_1 p \leq \alpha$.

1. Phase I: Unsegmented phase

   (a) Generate 'small' primes up to $K = \max\left(\lfloor\sqrt{N/m_2}\rfloor, \lfloor\alpha/m_1\rfloor\right)$.
   (proc. `SmallPrimes`(K))

   (b) Generate all numbers $m_1 p \leq \alpha$ where p is prime. In Algorithm 2 these are sorted and stored separately according to their residues modulo m_2.
   (proc. `GenerateIsm1p`(α) [`Generatem1pr`(α)] in Algorithm 1 [2])

   (c) Generate the modulo $\mathrm{lcm}_{\mathrm{m_1,m_2}}$ 'residue wheel', i.e., the array of all $\mathrm{lcm}_{\mathrm{m_1,m_2}}$ residues relatively prime to $m_1 m_2$ and congruent to $m_1 + m_2$ modulo 2. (proc. `GenerateResiduePattern`(m_1, m_2))

2. Phase II: Check GGC_{m_1,m_2} segment by segment. For each interval $[A, B)$:

   (a) Generate m_2-times multiples of 'large' primes in an interval.
   (proc. `Generatem2qr`(C, D) [`Generateism2q`(C, D)] in Algorithm 1 [2])

      i. Generate all primes in interval $[C/m_2, D/m_2)$. (The values C and D depend on A and B.)

      ii. Generate all numbers of the form $m_2 q$ in interval $[C, D)$, where q is prime. Algorithm 1 sorts and stores these numbers separately according to their residues modulo m_1.

   (b) Check GGC_{m_1,m_2} in interval $[A, B)$.
   (proc. `Check1`(A, B) [`Check2`(A, B)] in Algorithm 1 [2])

3.2.3 Some ideas applied in both algorithms

In order to check if every number in an interval $[A, B)$ satisfying the conditions of GGC_{m_1,m_2} has a partition m_1p+m_2q such that $m_1p \le \alpha$, it is sufficient to possess the lists of all numbers $m_1p \le \alpha$ where p is prime, and of all numbers m_2q in interval $[\max(0, A-\alpha), B)$ where q is prime. These lists are generated in Phases I and II, respectively. Although methods with lower asymptotic time complexities exist [3, 1, 18, 2, 22], in Phases I and II the sieve of Eratosthenes and a segmented version of this, respectively, is used to generate primes.

If $n = m_1p + m_2q$ is an (m_1, m_2)-partition then

$$m_2q \equiv n \pmod{m_1} \text{ and} \tag{2}$$

$$m_1p \equiv n \pmod{m_2}. \tag{3}$$

Therefore, for each n, Algorithm 1 [2] in Phase II tries as candidates for $q^{**}_{m_1,m_2}(n)$ $[p^{*}_{m_1,m_2}(n)]$ only primes q $[p]$ satisfying congruence (2) [(3)], which reduces the number of candidates tested by approximately a factor of $1/\varphi(m_1)$ $[1/\varphi(m_2)]$. In order to facilitate this, when generating numbers of the form m_2q $[m_1p]$ in an interval [up to α] Algorithm 1 [2] also sorts them by their residues modulo m_1 $[m_2]$.

3.2.4 Detailed description of the steps in Algorithm 1

Phase I: Procedure `SmallPrimes(`K`)` generates a list of all 'small' primes up to $K = \max(\lfloor\sqrt{N_{m_1,m_2}/m_2}\rfloor, \lfloor\alpha/m_1\rfloor)$, using the sieve of Eratosthenes. Procedure `GenerateIsm1p`(α) outputs the boolean array $\mathsf{ism_1p}$ of length $\alpha+1$ such that for all $0 \le i \le \alpha$: $\mathsf{ism_1p}[i] = 1$ if and only if $i = m_1p$ for some prime p. When checking GGC_{m_1,m_2} only those numbers n need to be tested which satisfy the conditions of GGC_{m_1,m_2}, which holds if and only if the residue $n \bmod \mathrm{lcm_{m_1,m_2}}$ satisfies these. Procedure `GenerateResiduePattern`(m_1, m_2) generates boolean array res of length $\mathrm{lcm_{m_1,m_2}}$ such that for all $0 \le i \le \mathrm{lcm_{m_1,m_2}} - 1$: $\mathsf{res}[i] = 1$ if and only if $\gcd(i, m_1) = \gcd(i, m_2) = 1$ and $i \equiv m_1 + m_2 \pmod 2$.

Phase II: For given integers $0 \le C < D$ such that $2m_1m_2|C$ and $2m_1m_2|D$, procedure `Generatem2qr`(C, D) generates all numbers of the form m_2q in interval $[C, D)$ where q is prime, and stores each m_2q in array $\mathsf{m_2q[r]}$ where $r = m_2q \bmod m_1$ $(0 \le r < m_1)$. For given integers $0 \le A < B$ where $2m_1m_2|A$ and $2m_1m_2|B$, procedure `Check1`(A, B) checks for every n in $[A, B)$ satisfying the conditions of GGC_{m_1,m_2} if there exist primes p and q such that $n = m_1p + m_2q$ and $m_1p \le \alpha$. The procedure looks for the p-minimal (m_1, m_2)-partition of n, applying a 'descending' search for $q^{**}_{m_1,m_2}(n)$: trying in decreasing order the values m_2q where q is prime such that $m_2q \equiv n \pmod{m_1}$ —taking these from array $\mathsf{m_2q[r]}$ where $r = n \bmod m_1$—and checking if $n - m_2q$ is of the form m_1p for some prime p. If such m_2q is found then $q^{**}(n) = q$ and $p^*(n) = (n - m_2q)/m_1$. Otherwise n is added to array $\mathsf{residual}$. The output is array $\mathsf{residual}$ of those n in $[A, B)$ satisfying the conditions of GGC_{m_1,m_2} for which there exist no primes p and q such that $n = m_1p + m_2q$ and $m_1p \le \alpha$.

Section 8 contains the pseudocode of procedure `GGC1`$(N, m_1, m_2, \triangle,\ \alpha)$ implementing Algorithm 1.

4 Implementation and checking for correctness

We implemented Algorithms 1 and 2 in `C++`. For each $m_1 \leq m_2 \leq 40$ relatively prime, we checked GGC_{m_1,m_2} up to N_{m_1,m_2} by Algorithms 1a, 1b, 2a, and 2b. (For $m_1 = m_2 = 1$, Algorithms 1a and 1b [2a and 2b] are identical.) The program for Algorithm 1 [2] performed both Algorithms 1a and 1b [2a and 2b], with the values of m_1 and m_2 interchanged (with $m_1 < m_2$ in Algorithms 1a and 2a). Each algorithm ran on one core of a 32-core 64-bit Intel Xeon Scalable processor.

For each pair $m_1 \leq m_2$ the output arrays `residual` of the four algorithms (only two different algorithms in case $m_1 = m_2 = 1$) were identical. We generated the values $p^*_{m_1,m_2}(n)$ $[q^*_{m_1,m_2}(n)]$ and $q^{**}_{m_1,m_2}(n)$ $[p^{**}_{m_1,m_2}(n)]$ for all $\hat{k}_{m_1,m_2} < n \leq 10^6$ satisfying the conditions of GGC_{m_1,m_2}, which were also identical.

5 Comparing the running times of the algorithms

5.1 Experimental data on running times

For each pair $m_1 \leq m_2 \leq 40$ relatively prime, Algorithms 1a and 1b were both faster than Algorithms 2a and 2b, the former two significantly outperforming on average the latter. The speed rankings of Algorithms 1a and 1b [2a and 2b] varied depending on the pair $m_1 < m_2$. On average over all pairs $m_1 \leq m_2$, Algorithms 1a and 1b [2a and 2b] showed very similar speed performances. Table 1 presents the average, lowest, and highest running times of each algorithm, and the pair m_1, m_2 where the latter occurred.

Algorithm	Lowest			Highest			Average time (sec)
	m_1	m_2	time (sec)	m_1	m_2	time (sec)	
Alg. 1a	7	30	22473	16	29	114177	56045
Alg. 1b	6	35	23345	1	16	108614	54461
Alg. 2a	33	35	55742	31	32	293279	132154
Alg. 2b	35	39	57391	32	37	293734	134559

Table 1: Lowest, highest, and average running times (sec) of the algorithms up to $N_{m_1,m_2} \approx 10^{12}$ over all pairs $m_1 \leq m_2 \leq 40$ relatively prime.

For each pair $m_1 < m_2$ tested the running times of the four algorithms ranked in one of the following four orders from fastest to slowest:

- Group A: Algorithms 1a, 1b, 2a, 2b
- Group B: Algorithms 1a, 1b, 2b, 2a
- Group C: Algorithms 1b, 1a, 2a, 2b
- Group D: Algorithms 1b, 1a, 2b, 2a

Groups A, B, C, and D contain 21, 218, 242, and 8 pairs, respectively, as shown by Table 6 in Section 7. The dominance of groups B and C raises the question whether the pairs in groups A and D would also move to one of these groups when testing up to sufficiently large thresholds. In all 8 pairs in group D the running times of Algorithms 1a and 1b or those of 2a and 2b were 'very close'. We ran all four algorithms for the pairs $(9, 32), (11, 29), (17, 19)$, and $(23, 29)$ in group D—and for six other pairs including $(1, 2)$—up to $\approx 10^{13}$. The running times are shown in Table 2. The speed rankings changed for all four pairs in group D. The pairs $(9, 32)$, $(11, 29)$, $(17, 19)$, and $(23, 29)$ moved to groups B, C, A, and A, respectively. In the latter two cases the running times of Algorithms 1a and 1b were 'very close' to each other, which makes it plausible that the pairs might move again to another group if testing until even higher thresholds. These results suggest that the remaining other four pairs in group D may also leave this group in case of larger thresholds.

m_1	m_2	Running times (sec) up to $\approx 10^{12}$				Running times (sec) up to $\approx 10^{13}$			
		Alg. 1a	Alg. 1b	Alg. 2a	Alg. 2b	Alg. 1a	Alg. 1b	Alg. 2a	Alg. 2b
1	2	77192	104914	290478	182075	754817	1052855	2290673	1873002
1	3	42991	67452	106733	110383	423373	671771	1125928	1154959
1	5	56912	77743	129483	142355	555759	786325	1350077	1515172
1	7	63372	82353	137414	158389	627182	823687	1479262	1704191
1	9	44371	69002	101032	111152	431663	687392	1065525	1172360
1	11	69622	86173	143534	169134	745476	860608	1470691	1753254
9	32	43546	43514	132022	111549	549958	582308	1331716	1146279
11	29	74485	52092	148247	145263	744006	706943	1404459	1563704
17	19	55562	55052	143130	143104	736166	738674	1458811	1545744
23	29	80070	59988	155817	155277	800289	807146	1567925	1676299

Table 2: Running times (sec) of the algorithms up to $\approx 10^{12}$ and $\approx 10^{13}$ for some m_1, m_2.

5.2 Estimations for the running times

The significant parts of the computation in Algorithm 1 [2] are `Generatem2qr` and `Check1` [`Generateism2q` and `Check2`]. During all iterations procedure `Generatem2qr` [`Generateism2q`] generates all primes up to N/m_2, and their m_2-times multiples, using $O(N \log\log N)$ [25] and $\pi(N/m_2) \sim N/(m_2(\ln N - \ln m_2)) = o(N \log\log N)$ operations, respectively. Hence, `Generatem2qr` [`Generateism2q`] takes $O(N \log\log N)$ time.

In absence of approximations for the functions $p^*_{m_1,m_2}(n)$ it is difficult to estimate the number of operations performed by `Check1` [`Check2`]. However, we can establish the following. For given m_1, m_2 relatively prime, the number of values $n \leq N_{m_1,m_2}$ tested—i.e., of those satisfying the conditions of GGC_{m_1,m_2}—is approximately $\varphi(m_1 m_2) N_{m_1,m_2} / \operatorname{lcm}_{\mathrm{m_1,m_2}} \approx 10^{12}\varphi(m_1 m_2)/\operatorname{lcm}_{\mathrm{m_1,m_2}}$.

In Algorithm 1, for each n tested, the number of candidates `Check1` tries for $q^*_{m_1,m_2}(n)$ is approximately the number of primes q in the interval between $n/m_2 - m_1 p^*_{m_1,m_2}(n)/m_2$ and n/m_2 of length $m_1 p^*_{m_1,m_2}(n)/m_2$ satisfying $m_2 q \equiv n \pmod{m_1}$. Using $\pi(x) - \pi(x-y) \approx y/\ln(x)$ [14], this can be estimated as follows:

$$\frac{m_1 p^*_{m_1,m_2}(n)}{\varphi(m_1) m_2 \ln(n/m_2)} \approx \frac{m_1 p^*_{m_1,m_2}(n)}{\varphi(m_1) m_2 \ln(n)}. \tag{4}$$

In Algorithm 2 for each value n tested, the number of candidates `Check2` checks for $p^*_{m_1,m_2}(n)$ is equal to the number of primes p up to $p^*_{m_1,m_2}(n)$ satisfying $m_1 p \equiv n \pmod{m_2}$, which is approximately the following:

$$\frac{\pi(p^*_{m_1,m_2}(n))}{\varphi(m_2)} \sim \frac{p^*_{m_1,m_2}(n)}{\varphi(m_2)\ln p^*_{m_1,m_2}(n)}. \tag{5}$$

5.3 Some heuristics

Currently possessing no approximations for $p^*_{m_1,m_2}(n)$, and thus for the number of operations performed by `Check1` and `Check2`, it is unclear how the time complexities of `Generatem2qr` and `Check1` [`Generateism2q` and `Check2`] compare. In order to obtain empirical data, we ran Algorithm 1a for four pairs $m_1 \leq m_2$ up to the thresholds of approximately $10^6, 10^7, 10^8$, and 10^9, and measured the times taken by `Check1` and `Generatem2qr`. In one case `Check1` took around 66%, and in all other cases above 80% (usually above 90%), whereas `Generatem2qr` took in one case 16%, but in all other cases below 10%, and usually below 5% of the total time. As the threshold increased, Algorithm 1a spent an increasing and a decreasing fraction of the total time on `Check1` and on `Generatem2qr`, respectively.

In the arguments below we assume that in Algorithm 1 [2] `Check1` [`Check2`] is the most time consuming part of the computation, with higher time complexity than `Generatem2qr` [`Generateism2q`]; hence, the relative speed performances of Algorithms 1a, 1b, 2a, and 2b are determined by `Check1` and `Check2`.

5.3.1 Comparing the running times of Algorithms 1a and 2a [1b and 2b]

Granville et al. [9] conjectured that $p(n) = p^*_{1,1}(n) = O(\log^2 n \log\log n)$, implying $p^*_{1,1}(n) = o(n^\varepsilon)$ for every $\varepsilon \in \mathbb{R}^+$. Based on our data we also conjecture that for all m_1 and m_2 and $\varepsilon \in \mathbb{R}^+$ we have $p^*_{m_1,m_2}(n) = o(n^\epsilon)$. This assumption yields $\ln p^*_{m_1,m_2}(n) = o(\ln(n))$. Hence

$$\frac{m_1 p^*_{m_1,m_2}(n)}{\varphi(m_1)\ln(n)} = o\left(\frac{p^*_{m_1,m_2}(n)}{\varphi(m_2)\ln p^*_{m_1,m_2}(n)}\right),$$

which heuristically suggests that Algorithm 1a [1b] is faster than Algorithm 2a [2b] for all m_1, m_2, when run until sufficiently large threshold. This prediction is in complete accordance with our results: for each pair m_1, m_2 tested Algorithms 1a and 1b were both faster than Algorithms 2a and 2b.

5.3.2 Comparing the running times of Algorithms 1a and 1b [2a and 2b]

Since for given m_1, m_2, in Algorithms 1a and 1b [2a and 2b] `Check1` [`Check2`] checks the same number of values n, one may attempt to explain their relative speed performances using some estimate of the 'average' time spent by `Check1` [`Check2`] on processing each value

n. Based on estimates 4 and 5, we introduce the following functions for every sufficiently large number L:

$$f_L(m_1, m_2) := \frac{m_1 \overline{p^*}_L(m_1, m_2)}{\varphi(m_1) m_2} \text{ and } g_L(m_1, m_2) := \frac{\overline{p^*}_L(m_1, m_2)}{\varphi(m_2) \ln \overline{p^*}_L(m_1, m_2)},$$

where $\overline{p^*}_L(m_1, m_2)$ is the average of $p^*_{m_1,m_2}(n)$ up to L. Then for all m_1, m_2 and all L and N sufficiently large, the following hypotheses can be considered when testing GGC_{m_1,m_2} up to N:

$\mathrm{H}_1(L, N)$: Algorithm 1a is faster than Algorithm 1b if and only if

$$f_L(m_1, m_2) < f_L(m_2, m_1) \quad \left(\Leftrightarrow \quad \frac{\overline{p^*}_L(m_1, m_2)}{\overline{p^*}_L(m_2, m_1)} < \frac{m_2^2 \varphi(m_1)}{m_1^2 \varphi(m_2)} \right) \tag{6}$$

$\mathrm{H}_2(L, N)$: Algorithm 2a is faster than Algorithm 2b if and only if

$$g_L(m_1, m_2) < g_L(m_2, m_1) \quad \left(\Leftrightarrow \quad \frac{\overline{p^*}_L(m_1, m_2) \ln \overline{p^*}_L(m_2, m_1)}{\overline{p^*}_L(m_2, m_1) \ln \overline{p^*}_L(m_1, m_2)} < \frac{\varphi(m_2)}{\varphi(m_1)} \right). \tag{7}$$

Then H_1 is the hypothesis that $\mathrm{H}_1(L, N)$ is true for all $N \geq L$ where L is sufficiently large. Hypothesis H_2 is the claim that $\mathrm{H}_2(L, N)$ is true for all $N \geq L$ where L is sufficiently large.

We tested $\mathrm{H}_1(10^9, N_{m_1,m_2})$ and $\mathrm{H}_2(10^9, N_{m_1,m_2})$ for all 489 pairs $m_1 < m_2$ relatively prime. The pairs can be categorized as follows:

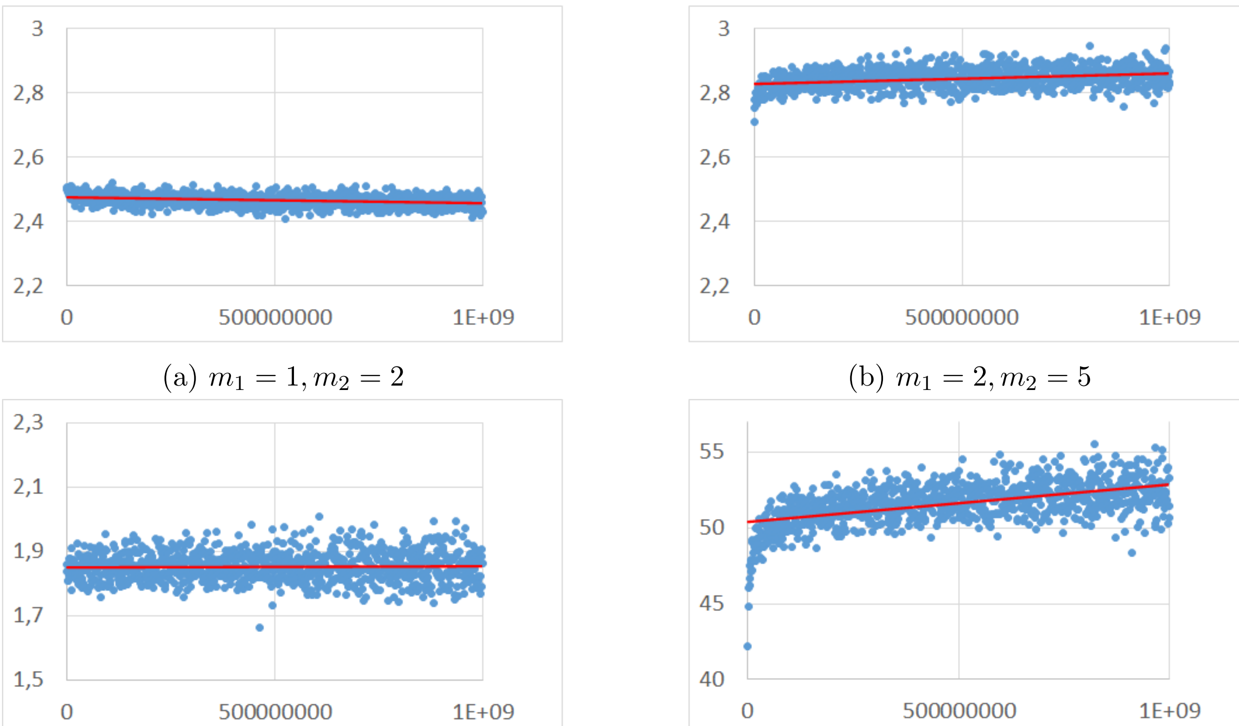
- Group a: $f_{10^9}(m_1, m_2) < f_{10^9}(m_2, m_1)$ and $g_{10^9}(m_1, m_2) < g_{10^9}(m_2, m_1)$.
- Group b: $f_{10^9}(m_1, m_2) < f_{10^9}(m_2, m_1)$ and $g_{10^9}(m_1, m_2) > g_{10^9}(m_2, m_1)$.
- Group c: $f_{10^9}(m_1, m_2) > f_{10^9}(m_2, m_1)$ and $g_{10^9}(m_1, m_2) < g_{10^9}(m_2, m_1)$.
- Group d: $f_{10^9}(m_1, m_2) > f_{10^9}(m_2, m_1)$ and $g_{10^9}(m_1, m_2) > g_{10^9}(m_2, m_1)$.

Group a is empty, while groups b, c, and d contain 227, 258, and 4 pairs, respectively. For all four pairs in group d at least one of the differences $|f_{10^9}(m_1, m_2) - f_{10^9}(m_2, m_1)|$ and $|g_{10^9}(m_1, m_2) - g_{10^9}(m_2, m_1)|$ is 'small' (less than 0.4). Hence, it is plausible that their group allocation may change if L is sufficiently large.

Table 6 shows the classification of the pairs into groups A, B, C, and D and a, b, c, and d, respectively. In our experiment $\mathrm{H}_1(10^9, N_{m_1,m_2})$ is true for 467 pairs (groups Ab, Bb, Cc, and Dc), and $\mathrm{H}_2(10^9, N_{m_1,m_2})$ holds for 476 pairs (groups Ac, Bb, Cc, Bd, and Db). Both claims hold for 458 pairs (groups Bb and Cc) among all 489 pairs. Among those 22 pairs for which $\mathrm{H}_1(10^9, N_{m_1,m_2})$ fails (groups Ac, Ad, Bc, Bd, and Db) in case of 15 pairs either the running times of Algorithms 1a and 1b were 'close' (i.e., differed by less than 10^4 sec) or $|f_{10^9}(m_1, m_2) - f_{10^9}(m_2, m_1)|$ was 'small' (i.e., less than 1). For all those 13 pairs for which $\mathrm{H}_2(10^9, N_{m_1,m_2})$ fails (groups Ab, Ad, Bc, and Dc) either the running times of Algorithms 2a and 2b were 'close' (differed by less than 10^4 sec) or $|g_{10^9}(m_1, m_2) - g_{10^9}(m_2, m_1)|$ was 'small' (less than 1). Hence, it is plausible that for sufficiently large N and L the hypotheses may also hold for most (or for all) of these pairs.

Further computational experiments, understanding the behaviours of, and developing estimations for the functions $p^*_{m_1,m_2}(n)$ could help ascertain the plausibility of the two hypotheses.

5.4 Further observations regarding $p^*_{m_1,m_2}(n)$



(a) $m_1 = 1, m_2 = 2$ (b) $m_1 = 2, m_2 = 5$

(c) $m_1 = 23, m_2 = 40$ (d) $m_1 = 1, m_2 = 33$

Figure 2: The quotient $\frac{\text{average } p^*_{m_1,m_2}(n)}{\text{average } q^*_{m_1,m_2}(n)}$ in each interval of length 10^6 centred at x, for $x = 10^6 k + 5 \cdot 10^5$ $(k = 0, 1, \ldots, 10^3 - 1)$, in cases of some m_1, m_2 indicated under each subfigure.

In Figure 1, one can note the slow growths of the average $p^*_{m_1,m_2}(n)$ in intervals of length 10^6 up to 10^9. The graphs are close to smooth curves and similar in shape.

Figure 2 displays the functions $x \mapsto$ average of $p^*_{m_1,m_2}(n)$/average of $q^*_{m_1,m_2}(n)$ in intervals of length 10^6 centred at $x = 10^6 k + 5 \cdot 10^5$ $(0 \leq k \leq 10^3 - 1)$ for $m_1 = 1, m_2 = 2$ (Subfigure 2a), $m_1 = 2, m_2 = 5$ (Subfigure 2b), $m_1 = 23, m_2 = 40$ (Subfigure 2c), and $m_1 = 1, m_2 = 33$ (Subfigure 2d). The graphs—especially the first three—appear to be remarkably close to straight lines: the trend lines with equations $y = -2 \cdot 10^{-11} x + 2.4745$, $y = 3 \cdot 10^{-11} x + 2.8297$, $y = 5 \cdot 10^{-12} x + 1.851$, and $y = 3 \cdot 10^{-9} x + 50.374$, respectively, indicated in each subfigure. The values of the functions fall within the following narrow intervals between their minima and maxima (correct to 3 decimal places): $[2.408, 2.519]$, $[2.711, 2.945]$, $[1.663, 2.01]$, and $[42.200, 55.582]$ (Subfigures 2a, 2b, 2c, and 2d, respectively). If the smoothly increasing or decreasing trends of these functions continue, it suggests that the functions $L \mapsto \overline{p^*}_L(m_1, m_2)/\overline{p^*}_L(m_2, m_1)$ may also be increasing or decreasing, accordingly. In this case inequality 6 is either simultaneously true or false for all L sufficiently large.

6 Conclusion and future work

The relatively small sizes of $\hat{k}_{m_1,m_2}$ in cases of all pairs m_1, m_2 tested support the plausibility of GGC, suggesting that the conjecture merits further investigation.

For all pairs $m_1, m_2 \leq 40$ relatively prime, algorithms applying descending search were faster than those using ascending search. Heuristic arguments suggest that this is probably the case in general. However, speed rankings of the two algorithms using descending [ascending] search varied by m_1, m_2. The fastest algorithm can be further developed, and potentially combined with sieving methods. Hence, it would be useful to obtain predictions for the fastest one for given m_1, m_2 when testing up to large thresholds. Hypotheses $\mathrm{H}_1(10^9, N_{m_1,m_2})$ and $\mathrm{H}_2(10^9, N_{m_1,m_2})$ were true in our implementation for most m_1, m_2 tested, providing support to H_1 and H_2. Further computational experiments, and developing approximations to $p^*_{m_1,m_2}(n)$ could help assess their plausibility, and possibly propose better predictions. It would be interesting to devise predictions for the speed rankings purely based on m_1, m_2.

Ranking by size of the averages $\overline{p^*}_L(m_1, m_2)$ for different $m_1, m_2 \leq 40$ for L sufficiently large appears to be independent of L. (We could observe this in our data only when $L \leq 10^{12}$, but this is likely to be the case also for all larger L.) Explaining this ranking—and, in particular, the observation that $\overline{p^*}_{10^9}(m_1, m_2) > \overline{p^*}_{10^9}(m_2, m_1)$ for all $m_1 < m_2$ tested (Table 5)—by the properties of m_1 and m_2 is a future goal.

Efficient sieving methods could be developed (and potentially combined with one of the four algorithms described) for testing GGC up to higher thresholds.

7 Tables of data

	5 greatest values		5 smallest values		Average
	value	(m_1, m_2)	value	(m_1, m_2)	value
max $p^*_{m_1,m_2}$ up to 10^9	78697	$(32, 37)$	449	$(30, 1)$	22889.33538
	77723	$(23, 37)$	557	$(17, 1)$	
	77267	$(37, 38)$	571	$(39, 1)$	
	76379	$(29, 38)$	599	$(21, 1)$	
	75989	$(1, 38)$	631	$(24, 1)$	
$\overline{p^*}_{m_1,m_2}$ up to 10^9	2064.47552	$(1, 37)$	12.74269	$(30, 1)$	687.7063317
	2059.89836	$(1, 38)$	15.37864	$(15, 1)$	
	2059.17801	$(16, 37)$	16.68819	$(21, 1)$	
	2059.1531	$(32, 37)$	17.27778	$(36, 1)$	
	2058.97664	$(2, 37)$	17.27898	$(6, 1)$	
$\hat{k}_{m_1,m_2}$	412987	$(32, 37)$, $(37, 32)$	2	$(1, 1)$	52004.838776
	403357	$(34, 37)$, $(37, 34)$	5	$(1, 2)$, $(2, 1)$	
	390367	$(37, 38)$, $(38, 37)$	10	$(1, 3)$, $(3, 1)$	
	377122	$(29, 37)$, $(37, 29)$	13	$(1, 6)$, $(6, 1)$	
	370837	$(29, 32)$, $(32, 29)$	17	$(2, 3)$, $(3, 2)$	

Table 3: The five greatest, smallest, and the average values of $\max p^*_{m_1,m_2}$ and of $\overline{p^*}_{m_1,m_2}$ up to 10^9 and of $\hat{k}_{m_1,m_2}$ over all pairs $m_1, m_2 \leq 40$ relatively prime.

m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$
1	1	2	5	32	12541	12	13	11449	20	31	102659
1	2	5	5	33	3182	12	17	15101	20	33	29797
1	3	10	5	34	13511	12	19	8737	20	37	156137
1	4	77	5	36	4699	12	23	16739	20	39	26251
1	5	24	5	37	12718	12	25	10477	21	22	29191
1	6	13	5	38	14527	12	29	25889	21	23	21962
1	7	36	5	39	4954	12	31	18547	21	25	20554
1	8	49	6	7	421	12	35	14303	21	26	33767
1	9	28	6	11	1361	12	37	67777	21	29	30746
1	10	29	6	13	1723	13	14	17827	21	31	30112
1	11	54	6	17	2447	13	15	3802	21	32	44473
1	12	25	6	19	3133	13	16	32507	21	34	47323
1	13	116	6	23	4901	13	17	28876	21	37	41794
1	14	163	6	25	2489	13	18	11239	21	38	54287
1	15	46	6	29	10987	13	19	30782	21	40	22943
1	16	473	6	31	10369	13	20	25913	22	23	108041
1	17	526	6	35	2059	13	21	6542	22	25	91277
1	18	37	6	37	9427	13	22	49631	22	27	49333
1	19	452	7	8	2711	13	23	44446	22	29	161383
1	20	109	7	9	754	13	24	14221	22	31	133283
1	21	88	7	10	2453	13	25	25658	22	35	91579
1	22	401	7	11	2294	13	27	16078	22	37	229309
1	23	832	7	12	2371	13	28	74849	22	39	56323
1	24	97	7	13	12326	13	29	64634	23	24	39959

Table 4: The value of $\hat{k}_{m_1,m_2}$ for all relatively prime $m_1 \leq m_2 \leq 40$.

m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$
1	25	296	7	15	1192	13	30	12949	23	25	76528
1	26	337	7	16	10463	13	31	82826	23	26	106201
1	27	136	7	17	8104	13	32	80609	23	27	50872
1	28	1157	7	18	6841	13	33	16024	23	28	136651
1	29	1588	7	19	17846	13	34	99131	23	29	172076
1	30	61	7	20	8387	13	35	48364	23	30	26633
1	31	2918	7	22	10729	13	36	31249	23	31	201812
1	32	1951	7	23	13492	13	37	92006	23	32	225457
1	33	214	7	24	6583	13	38	91009	23	33	51094
1	34	1313	7	25	8618	13	40	63913	23	34	163993
1	35	226	7	26	22657	14	15	2921	23	35	81274
1	36	397	7	27	4556	14	17	43423	23	36	68507
1	37	1616	7	29	29516	14	19	56237	23	37	269506
1	38	1117	7	30	3217	14	23	42709	23	38	273151
1	39	272	7	31	25304	14	25	23447	23	39	85906
1	40	1241	7	32	28057	14	27	19787	23	40	181699
2	3	17	7	33	5224	14	29	63871	24	25	44329
2	5	163	7	34	36461	14	31	71413	24	29	83609
2	7	89	7	36	6091	14	33	19571	24	31	83507
2	9	115	7	37	39896	14	37	83717	24	35	50339
2	11	673	7	38	21691	14	39	17189	24	37	100333
2	13	719	7	39	6472	15	16	8221	25	26	110687
2	15	173	7	40	30407	15	17	6668	25	27	39586
2	17	2371	8	9	1633	15	19	9664	25	28	88909
2	19	1757	8	11	6509	15	22	8161	25	29	102808
2	21	275	8	13	18461	15	23	12428	25	31	165446
2	23	2209	8	15	1399	15	26	13421	25	32	215743
2	25	2399	8	17	22273	15	28	16963	25	33	28454
2	27	781	8	19	19427	15	29	29396	25	34	146911
2	29	4339	8	21	3517	15	31	22636	25	36	87859
2	31	3229	8	23	47249	15	32	15227	25	37	251206
2	33	659	8	25	14081	15	34	19219	25	38	197587
2	35	3733	8	27	10427	15	37	21236	25	39	40738
2	37	11251	8	29	43711	15	38	23873	26	27	39293
2	39	1679	8	31	57719	16	17	42103	26	29	174451
3	4	55	8	33	10841	16	19	62507	26	31	233429
3	5	62	8	35	46243	16	21	12349	26	33	65059
3	7	94	8	37	57173	16	23	61861	26	35	142981
3	8	251	8	39	21799	16	25	62849	26	37	262897
3	10	133	9	10	811	16	27	26209	27	28	56647
3	11	140	9	11	2066	16	29	133321	27	29	74744
3	13	322	9	13	3008	16	31	128783	27	31	54784
3	14	461	9	14	2789	16	33	26981	27	32	82343

Table 4: The value of $\hat{k}_{m_1,m_2}$ for all relatively prime $m_1 \leq m_2 \leq 40$.

m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$
3	16	853	9	16	7657	16	35	55963	27	34	86791
3	17	554	9	17	3968	16	37	186427	27	35	41098
3	19	616	9	19	7498	16	39	48067	27	37	94342
3	20	1247	9	20	3803	17	18	16151	27	38	86143
3	22	817	9	22	11119	17	19	48058	27	40	63599
3	23	2204	9	23	7454	17	20	37717	28	29	202273
3	25	838	9	25	6658	17	21	13382	28	31	180791
3	26	1777	9	26	10271	17	22	83597	28	33	78469
3	28	1951	9	28	6469	17	23	89464	28	37	250961
3	29	1178	9	29	12058	17	24	39791	28	39	69259
3	31	3358	9	31	14422	17	25	39332	29	30	60619
3	32	3131	9	32	17021	17	26	89533	29	31	243562
3	34	1423	9	34	14803	17	27	34108	29	32	370837
3	35	608	9	35	5392	17	28	51589	29	33	105254
3	37	3814	9	37	18976	17	29	101834	29	34	244907
3	38	5741	9	38	21271	17	30	13703	29	35	166534
3	40	2347	9	40	20533	17	31	109916	29	36	97793
4	5	361	10	11	7489	17	32	120691	29	37	377122
4	7	1691	10	13	11051	17	33	52004	29	38	289069
4	9	629	10	17	13813	17	35	64166	29	39	117254
4	11	2383	10	19	14621	17	36	45109	29	40	228577
4	13	4073	10	21	3811	17	37	203162	30	31	54337
4	15	1291	10	23	22993	17	38	173681	30	37	56227
4	17	7759	10	27	10537	17	39	45572	31	32	344761
4	19	12167	10	29	28411	17	40	86201	31	33	87794
4	21	1537	10	31	35303	18	19	35353	31	34	317567
4	23	24499	10	33	10567	18	23	28153	31	35	176636
4	25	7181	10	37	45817	18	25	10843	31	36	171971
4	27	6511	10	39	12731	18	29	48683	31	37	363658
4	29	15133	11	12	3623	18	31	37957	31	38	348349
4	31	17723	11	13	13018	18	35	16937	31	39	121438
4	33	2773	11	14	11293	18	37	53407	31	40	313541
4	35	9271	11	15	1646	19	20	76319	32	33	108593
4	37	21881	11	16	25723	19	21	12112	32	35	195197
4	39	5443	11	17	18404	19	22	76493	32	37	412987
5	6	191	11	18	6893	19	23	110416	32	39	113111
5	7	458	11	19	35254	19	24	34129	33	34	136343
5	8	1333	11	20	17911	19	25	91904	33	35	39994
5	9	274	11	21	4022	19	26	120737	33	37	99146
5	11	1516	11	23	44204	19	27	26038	33	38	132331
5	12	953	11	24	9707	19	28	78671	33	40	71023
5	13	4582	11	25	31634	19	29	125218	34	35	166597
5	14	3379	11	26	42073	19	30	27077	34	37	403357

Table 4: The value of $\hat{k}_{m_1,m_2}$ for all relatively prime $m_1 \leq m_2 \leq 40$.

m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$	m_1	m_2	$\hat{k}_{m_1,m_2}$
5	16	4889	11	27	10994	19	31	169292	34	39	139459
5	17	2542	11	28	39167	19	32	171469	35	36	52631
5	18	1187	11	29	70618	19	33	68188	35	37	201062
5	19	3082	11	30	11021	19	34	156803	35	38	206653
5	21	656	11	31	45646	19	35	69442	35	39	53336
5	22	7523	11	32	63601	19	36	44647	36	37	113177
5	23	9218	11	34	64321	19	37	162286	37	38	390367
5	24	4229	11	35	31228	19	39	50608	37	39	140548
5	26	16543	11	36	18121	19	40	103619	37	40	264023
5	27	2858	11	37	68018	20	21	16129	38	39	188473
5	28	8237	11	38	84419	20	23	78457	39	40	145279
5	29	10246	11	39	26018	20	27	20663			
5	31	11668	11	40	59399	20	29	142097			

Table 4: The value of $\hat{k}_{m_1,m_2}$ for all relatively prime $m_1 \leq m_2 \leq 40$.

m_1	m_2	$p^*_{m_1,m_2}(n)$		$q^*_{m_1,m_2}(n)$		m_1	m_2	$p^*_{m_1,m_2}(n)$		$q^*_{m_1,m_2}(n)$		m_1	m_2	$p^*_{m_1,m_2}(n)$		$q^*_{m_1,m_2}(n)$	
		avg	max	avg	max			avg	max	avg	max			avg	max	avg	max
1	1					4	9	241.822	7927	97.774	3001	9	13	333.584	10193	222.26	6761
1	2	80.839	3037	32.8	1609	4	11	494.758	19507	160.372	5939	9	14	331.513	10067	198.584	6337
1	3	72.911	2371	20.072	743	4	13	607.515	24919	163.502	6311	9	16	463.174	13627	241.826	7219
1	4	181.026	6971	32.806	1453	4	15	327.845	9257	73.338	2153	9	17	464.765	13007	227.655	6481
1	5	176.526	6833	26.767	1093	4	17	841.539	29669	167.531	6553	9	19	528.846	15649	229.533	6301
1	6	157.484	4969	17.279	643	4	19	960.026	32801	168.921	6947	9	20	449.818	13921	180.773	5519
1	7	281.84	9431	29.376	1129	5	6	118.745	3457	93.492	2801	10	11	370.28	13093	339.42	12241
1	8	393.604	15497	32.806	1493	5	7	211.95	8969	145.513	6871	10	13	454.483	15731	345.898	11117
1	9	245.866	8431	20.072	647	5	8	295.392	10369	172.142	6229	10	17	632.311	21647	354.305	14369
1	10	382.522	13009	23.958	1153	5	9	184.588	5333	97.315	2731	10	19	721.599	25057	357.248	13033
1	11	500.068	17093	31.678	1499	5	11	375.582	11839	156.587	5881	11	12	304.228	8821	267.184	8293
1	12	342.648	11261	17.279	673	5	12	257.838	7309	93.511	2969	11	13	542.8	17299	452.035	16829
1	13	612.063	23663	32.294	1297	5	13	459.273	16477	159.551	5521	11	14	542.423	20359	406.549	15227
1	14	611.042	20359	26.557	1129	5	14	459.822	15773	141.315	4651	11	15	295.49	8941	203.796	5527
1	15	332.373	9127	15.379	557	5	16	638.409	24677	172.167	6451	11	16	754.067	26839	494.633	17863
1	16	849.623	33997	32.803	1597	5	17	637.215	22751	163.446	5657	11	17	751.415	25621	463.029	17713
1	17	846.422	32779	33.084	1381	5	18	398.036	10499	93.511	2963	11	18	470.391	14251	267.222	7681
1	18	529.975	15313	17.28	701	5	19	726.834	25609	164.784	5711	11	19	856.789	27581	466.872	15467
1	19	964.977	33791	33.364	1321	6	7	149.317	4597	129.94	3923	11	20	734.055	26497	370.239	12853
1	20	825.834	29209	23.957	1069	6	11	267.139	8543	139.856	4813	12	13	328.85	9871	310.206	9479
2	3	69.352	2083	43.626	1399	6	13	328.759	10883	142.486	4957	12	17	459.61	13033	317.598	10657
2	5	172.137	6379	60.482	2459	6	17	459.546	14731	145.948	4201	12	19	523.692	14699	320.198	9437
2	7	277.107	12011	66.282	2663	6	19	523.593	16703	147.126	4423	13	14	552.943	19889	499.815	16843
2	9	241.78	7129	43.628	1549	7	8	323.92	12589	277.119	11197	13	15	301.049	8539	250.574	7151
2	11	494.633	21107	71.487	3061	7	9	202.501	5717	154.108	4271	13	16	768.659	28463	607.268	25127
2	13	607.339	21383	72.924	3049	7	10	315.347	9769	207.433	6841	13	17	765.986	25747	566.894	21851
2	15	327.714	9049	32.917	1031	7	11	411.838	15131	249.973	9439	13	18	479.435	15199	328.849	9277
2	17	841.438	30859	74.71	3121	7	12	282.761	9137	149.358	4663	13	19	873.509	33703	571.528	22079
2	19	959.87	34039	75.341	3001	7	13	504.267	18593	254.754	10099	13	20	748.115	27953	454.483	14851
3	4	97.757	2939	69.363	2411	7	15	274.865	7499	116.406	3583	14	15	270.331	7789	248.994	6689
3	5	97.292	2909	55.338	1709	7	16	700.594	22783	277.12	10357	14	17	693.436	25121	566.271	20717
3	7	154.073	4517	60.42	1789	7	17	698.137	24109	260.916	11069	14	19	791.453	27277	570.866	20873
3	8	211.872	6869	69.37	2383	7	18	436.631	13367	149.359	4481	15	16	347.11	9521	327.84	8893
3	10	208.887	6359	51.951	1471	7	19	796.433	27583	263.145	10289	15	17	349.899	9539	308.256	8179
3	11	271.626	8231	64.839	2113	7	20	682.396	23689	207.435	6841	15	19	398.729	10979	310.687	9109
3	13	333.472	10733	66.064	1999	8	9	241.796	7027	211.92	6961	16	17	841.42	30727	787.381	29531
3	14	331.38	10259	57.045	1867	8	11	494.594	18481	348.832	13499	16	19	959.865	35327	793.796	28631
3	16	463.076	13553	69.361	2239	8	13	607.287	23887	355.623	12107	17	18	491.044	14149	459.684	12953
3	17	464.638	12503	67.628	2269	8	15	327.799	9091	158.333	4817	17	19	894.547	33721	790.894	26927
3	19	528.697	15217	68.167	2063	8	17	841.438	31081	364.403	15749	17	20	765.917	28429	632.339	25237
3	20	449.579	12659	51.956	1579	8	19	959.894	42727	367.416	13999	18	19	523.747	14897	495.035	16943
4	5	172.187	7109	135.388	5521	9	10	208.976	6469	180.762	5501	19	20	772.014	28729	721.715	24071
4	7	277.169	11497	148.746	5939	9	11	271.709	8363	218.093	6827						

Table 5: Average and maximum values of $p^*_{m_1,m_2}(n)$ and $q^*_{m_1,m_2}(n)$ where $n \leq 10^9$, for all relatively prime $m_1 \leq m_2 \leq 40$.

Group	The ordered pairs (m_1, m_2) contained by the group
Ab:	(1, 3), (1, 9), (1, 15), (1, 21), (1, 33), (1, 39)
Ac:	(1, 7), (1, 11), (1, 13), (1, 17), (1, 19), (1, 25), (1, 31), (1, 37), (2, 9), (2, 15), (2, 21), (7, 11)
Ad:	(1, 5), (1, 27), (1, 35)
Bb:	(1, 2), (1, 4), (1, 6), (1, 8), (1, 10), (1, 12), (1, 14), (1, 16), (1, 18), (1, 20), (1, 22), (1, 24), (1, 26), (1, 28), (1, 30), (1, 32), (1, 34), (1, 36), (1, 38), (1, 40), (3, 4), (3, 8), (3, 10), (3, 14), (3, 16), (3, 20), (3, 22), (3, 26), (3, 28), (3, 34), (3, 38), (3, 40), (5, 6), (5, 8), (5, 9), (5, 12), (5, 14), (5, 16), (5, 18), (5, 21), (5, 22), (5, 24), (5, 26), (5, 27), (5, 28), (5, 32), (5, 33), (5, 34), (5, 36), (5, 38), (5, 39), (7, 8), (7, 9), (7, 10), (7, 12), (7, 15), (7, 16), (7, 18), (7, 20), (7, 22), (7, 24), (7, 26), (7, 27), (7, 30), (7, 32), (7, 33), (7, 34), (7, 36), (7, 38), (7, 39), (7, 40), (9, 10), (9, 14), (9, 16), (9, 20), (9, 22), (9, 26), (9, 28), (9, 34), (9, 38), (9, 40), (11, 12), (11, 14), (11, 15), (11, 16), (11, 18), (11, 20), (11, 21), (11, 24), (11, 25), (11, 26), (11, 27), (11, 28), (11, 30), (11, 32), (11, 34), (11, 35), (11, 36), (11, 38), (11, 39), (11, 40), (13, 14), (13, 15), (13, 16), (13, 18), (13, 20), (13, 21), (13, 22), (13, 24), (13, 25), (13, 27), (13, 28), (13, 30), (13, 32), (13, 33), (13, 34), (13, 35), (13, 36), (13, 38), (13, 40), (15, 16), (15, 22), (15, 26), (15, 28), (15, 34), (15, 38), (17, 18), (17, 20), (17, 21), (17, 22), (17, 24), (17, 25), (17, 26), (17, 27), (17, 28), (17, 30), (17, 32), (17, 33), (17, 35), (17, 36), (17, 38), (17, 39), (17, 40), (19, 20), (19, 21), (19, 22), (19, 24), (19, 25), (19, 26), (19, 27), (19, 28), (19, 30), (19, 32), (19, 33), (19, 34), (19, 35), (19, 36), (19, 39), (19, 40), (21, 22), (21, 26), (21, 32), (21, 34), (21, 38), (21, 40), (23, 24), (23, 25), (23, 26), (23, 27), (23, 28), (23, 30), (23, 32), (23, 33), (23, 34), (23, 35), (23, 36), (23, 38), (23, 39), (25, 26), (25, 27), (25, 28), (25, 32), (25, 33), (25, 34), (25, 36), (25, 38), (25, 39), (27, 28), (27, 32), (27, 34), (27, 38), (27, 40), (29, 30), (29, 32), (29, 33), (29, 34), (29, 35), (29, 36), (29, 38), (29, 39), (31, 32), (31, 33), (31, 34), (31, 35), (31, 36), (31, 38), (31, 39), (33, 34), (33, 38), (33, 40), (35, 36), (35, 38), (35, 39), (37, 38), (37, 39), (39, 40)
Bc:	(15, 32)
Bd:	(3, 32)
Cc:	(1, 23), (1, 29), (2, 3), (2, 5), (2, 7), (2, 11), (2, 13), (2, 17), (2, 19), (2, 23), (2, 25), (2, 27), (2, 29), (2, 31), (2, 33), (2, 35), (2, 37), (2, 39), (3, 5), (3, 7), (3, 11), (3, 13), (3, 17), (3, 19), (3, 23), (3, 25), (3, 29), (3, 31), (3, 35), (3, 37), (4, 5), (4, 7), (4, 9), (4, 11), (4, 13), (4, 15), (4, 17), (4, 19), (4, 21), (4, 23), (4, 25), (4, 27), (4, 29), (4, 31), (4, 33), (4, 35), (4, 37), (4, 39), (5, 7), (5, 11), (5, 13), (5, 17), (5, 19), (5, 23), (5, 29), (5, 31), (5, 37), (6, 7), (6, 11), (6, 13), (6, 17), (6, 19), (6, 23), (6, 25), (6, 29), (6, 31), (6, 35), (6, 37), (7, 13), (7, 17), (7, 19), (7, 23), (7, 25), (7, 29), (7, 31), (7, 37), (8, 9), (8, 11), (8, 13), (8, 15), (8, 17), (8, 19), (8, 21), (8, 23), (8, 25), (8, 27), (8, 29), (8, 31), (8, 33), (8, 35), (8, 37), (8, 39), (9, 11), (9, 13), (9, 17), (9, 19), (9, 23), (9, 25), (9, 29), (9, 31), (9, 35), (9, 37), (10, 11), (10, 13), (10, 17), (10, 19), (10, 21), (10, 23), (10, 27), (10, 29), (10, 31), (10, 33), (10, 37), (10, 39), (11, 13), (11, 17), (11, 19), (11, 23), (11, 31), (11, 37), (12, 13), (12, 17), (12, 19), (12, 23), (12, 25), (12, 29), (12, 31), (12, 35), (12, 37), (13, 17), (13, 19), (13, 23), (13, 29), (13, 31), (13, 37), (14, 15), (14, 17), (14, 19), (14, 23), (14, 25), (14, 27), (14, 29), (14, 31), (14, 33), (14, 37), (14, 39), (15, 17), (15, 19), (15, 23), (15, 29), (15, 31), (15, 37), (16, 17), (16, 19), (16, 21), (16, 23), (16, 25), (16, 27), (16, 29), (16, 31), (16, 33), (16, 35), (16, 37), (16, 39), (17, 23), (17, 29), (17, 31), (17, 37), (18, 19), (18, 23), (18, 25), (18, 29), (18, 31), (18, 35), (18, 37), (19, 23), (19, 29), (19, 31), (19, 37), (20, 21), (20, 23), (20, 27), (20, 29), (20, 31), (20, 33), (20, 37), (20, 39), (21, 23), (21, 25), (21, 29), (21, 31), (21, 37), (22, 23), (22, 25), (22, 27), (22, 29), (22, 31), (22, 35), (22, 37), (22, 39), (23, 31), (23, 37), (24, 25), (24, 29), (24, 31), (24, 35), (24, 37), (25, 29), (25, 31), (25, 37), (26, 27), (26, 29), (26, 31), (26, 33), (26, 35), (26, 37), (27, 29), (27, 31), (27, 35), (27, 37), (28, 29), (28, 31), (28, 33), (28, 37), (28, 39), (29, 31), (29, 37), (30, 31), (30, 37), (31, 37), (32, 33), (32, 35), (32, 37), (32, 39), (33, 35), (33, 37), (34, 35), (34, 37), (34, 39), (35, 37), (36, 37), (38, 39)
Db:	(9, 32), (23, 40), (29, 40), (31, 40), (37, 40)
Dc:	(11, 29), (17, 19), (23, 29)

Table 6: Classification of all pairs $m_1 < m_2 \leq 40$ relatively prime into groups A, B, C, and D and a, b, c, and d, indicated in the first column by upper and lower case letters, respectively.

8 Pseudocode for Algorithm 1

```
Function GGC1(N, m_1, m_2, △, α)
    Input  : positive integers N, m_1, m_2, △, and α such that gcd(m_1, m_2) = 1, N > 9, 2m_1m_2|N,
             2m_1m_2|△, and α ≤ △.
    Output : array residual containing all numbers n ≤ N satisfying the conditions of GGC_{m_1,m_2}
             for which there do not exist primes p and q such that n = m_1p + m_2q and m_1p ≤ α.
    /* Start Phase I: Unsegmented phase                                                  */
    /* Generating array primes.                                                          */
    SmallPrimes(max(⌊sqrt(N_{m_1,m_2} / m_2)⌋, ⌊α / m_1⌋));
    /* Assigning values to array ism1p.                                                  */
    GenerateIsm1p(α);
    /* Assigning values to array res.                                                    */
    GenerateResiduePattern(m_1, m_2);
    /* Start Phase II: Segmented phase                                                   */
    /* Initialization                                                                    */
    Set arrays residual and m2q[r] (0 ≤ r < m_1) empty;
    A ← 0;
    /* Start segmented computation                                                       */
    while A < N do
        B ← min(A + △, N);
        /* Keeping only those values in each array m2q[r] generated in previous
           iteration which are greater than A − α and removing all other values.         */
        if A > 0 then
            for r = 0 to m_1 − 1 do
                i ← 0;
                while i < length(m2q[r]) and m2q[r][i] < A − α do
                    i ← i + 1;
                end
                if i ≠ 0 then
                    remove_interval(mq2[r], [0, ..., i − 1])
                end
            end
        end
        /* Assigning new values to arrays m2q[r].                                        */
        Generatem2qr(A, B);
        /* Checking GGC_{m_1,m_2} in new interval.                                       */
        Check1(A, B);
        A ← A + △;
    end
end
```

Algorithm 1: Pseudocode for the main program implementing Algorithm 1.

9 Acknowledgments

We thank the anonymous referees for their helpful comments and suggestions regarding both content and presentation, which have lead to a much improved version of the paper.

References

[1] D. R. Barstow, An experiment in knowledge-based automatic programming, *Artificial Intelligence* **12** (1979), 73–119.

[2] S. Bengelloun, An incremental primal sieve, *Acta Inform.* **23** (1986), 119–125.

[3] C. Bays and R. H. Hudson, The segmented sieve of Eratosthenes and primes in arithmetic progressions to 10^{12}, *BIT* **17** (1977), 121–127.

[4] J. R. Chen, On the representation of a large even number as the sum of a prime and the product of at most two primes, *Sci. Sin.* **16** (1973), 157–176.

[5] L. E. Dickson, A new extension of Dirichlet's theorem on prime numbers, *Messenger of Mathematics* **33** (1904), 155–161.

[6] L. E. Dickson, *History of the Theory of Numbers, Vol. I,* Carnegie Institution of Washington, 1919.

[7] G. Farkas and Zs. Juhász, A generalization of Goldbach's conjecture, *Ann. Univ. Sci. Budapest. Sect. Comput.* **46** (2017), 39–53.

[8] P. N. Fuss, *Correspondance Mathématique et Physique de Quelques Célèbres Géomètres du XVIIIème Siècle*, L'Impr. de l'Académie impériale des sciences, 1843.

[9] A. Granville, J. van de Lune, and H. J. J. te Riele, Checking the Goldbach conjecture on a vector computer, in R. A. Mollin, ed., *Number Theory and Applications,* Kluwer Academic Publishers, 1989, pp. 423–433.

[10] D. Gries and J. Misra, A linear sieve algorithm for finding prime numbers, *Commun. ACM* **21** (1978), 999–1003.

[11] G. H. Hardy and J. E. Littlewood, Some problems of 'Partitio numerorum'; III: On the expression of a number as a sum of primes, *Acta Math.* **44** (1923), 1–70.

[12] H. A. Helfgott, The ternary Goldbach conjecture is true, arxiv preprint arXiv:1312.7748 [math.NT], 2013. Available at `https://arxiv.org/abs/1312.7748`.

[13] H. A. Helfgott, The ternary Goldbach problem, arxiv preprint arXiv:1501.05438 [math.NT], 2015. Available at `https://arxiv.org/abs/1501.05438`.

[14] D. R. Heath-Brown, The number of primes in a short interval, *J. Reine Angew. Math.* **389** (1988), 22–63.

[15] É. Lemoine, Lemoine's conjecture, *L'Intermédiaire des Mathématiciens* **1** (1894), 179.

[16] H. Maier, Primes in short intervals, *Michigan Math. J.* **32** (1985), 221–225.

[17] Make the Brain Happy, Lemoine's conjecture verified to 10^{10}, 2019, https://www.makethebrainhappy.com/2019/06/lemoines-conjecture-verified-to-1010.html.

[18] J. Misra, An exercise in program explanation, *ACM Transactions on Programming Languages and Systems* **3** (1981), 104–109.

[19] T. Oliveira e Silva, Goldbach conjecture verification, 2015, http://sweet.ua.pt/tos/goldbach.html.

[20] T. Oliveira e Silva, S. Herzog, and S. Pardi, Empirical verification of the even Goldbach conjecture and computation of prime gaps up to $4 \cdot 10^{18}$, *Math. Comp.* **83** (2014), 2033–2060.

[21] N. Pipping, Die Goldbachsche Vermutung und der Goldbach-Vinogradovsche Satz, *Acta Acad. Aboensis, Math. Phys.* **11** (1938), 4–25.

[22] P. Pritchard, Linear prime-number sieves: a family tree, *Sci. Comput. Program.* **9** (1987), 17–35.

[23] J. Richstein, Verifying the Goldbach conjecture up to $4 \cdot 10^{14}$, *Math. Comp.* **70** (2000), 1745–1749.

[24] M. K. Sinisalo, Checking the Goldbach conjecture up to $4 \cdot 10^{11}$, *Math. Comp.* **61** (1993), 931–934.

[25] J. Sorenson, An introduction to prime number sieves, Computer Sciences Technical Report #909, University of Wisconsin-Madison, Department of Computer Sciences, 1990. Available at: https://minds.wisconsin.edu/handle/1793/59248.

[26] W. C. Lu, Exceptional set of Goldbach number, *J. Number Theory* **130** (2010), 2359–2392.

2020 *Mathematics Subject Classification*: Primary 11A41; Secondary 11P32.
Keywords: Goldbach conjecture, Lemoine's conjecture, prime number.

(Concerned with sequences A000040, A002091, A002092, A046927, A194828, A195352, A195353, and A195354.)

Received March 31 2023; revised versions received April 1 2023; July 24 2023; February 2 2024. Published in *Journal of Integer Sequences*, March 18 2024.

Return to Journal of Integer Sequences home page.