# Goldbach Strong Conjecture Verification Using Prime Numbers

Marcin Barylski (marcin.a.barylski@gmail.com)

Published: April 8, 2018
The last update: December 9, 2018

## Abstract

Goldbach strong conjecture, still unsolved, states that all even integers $n>2$ can be expressed as the sum of two prime numbers (Goldbach partitions of $n$). We can also formulate it from the opposite perspective: from a set of prime numbers you may pick two primes, sum them, and you will be able to build every even number $n>2$. This work is devoted to studies on sum of two prime numbers.

## 1 Introduction

Goldbach strong (also called binary) conjecture asserts that all positive even integer $n \geq 4$ can be expressed as the sum of two prime numbers. This hypothesis, formulated by Goldbach in 1742 in letter to Euler [1] and then updated by Euler to the form above is one of the oldest and still unsolved problems in number theory. Empirical verification showed that it is true for all $n \leq 4$ x $10^{18}$ [2] [3].

The expression of a given positive even number $n$ as a sum of two primes $p_1$ and $p_2$ is called a Goldbach Partition (GP) of $n$. Let's denote this relation as $GSC(n, p_1, p_2)$. Then Goldbach strong conjecture can be written as (1):

$$\bigvee_{x>1, x\in\mathbb{N}} \exists_{p_1,p_2\in\mathbb{P}} GSC(2x, p_1, p_2) \quad (1)$$

We can also formulate it from the opposite perspective: from a set of all prime numbers you may pick two primes (may be the same), sum them, and you will be able to build every even number $n>2$.

**Lemma 1.** *GSC (x, 2, p)* $\implies$ *$x = 4 \wedge p = 2$.*

*Proof.* If $GSC(x, 2, p)$ then $x = 2+p$. By $GSC()$ definition $x$ is even and $p$ is prime. $p$ must be even otherwise $2 + p$ would not be even. The only even prime is 2, thus $p = 2$. If $p = 2$, then $x = 2 + p = 2 + 2 = 4$. □

**Lemma 2.** *GSC (x, $p_1$, $p_2$)* $\wedge$ *$p_1 > 2 \implies$ $x > 4 \wedge p_2 > 2$.*

*Proof.* If $GSC(x, p_1, p_2)$, then $x = p_1 + p_2$. By $GSC()$ definition $x$ is even. If $p_1$ is prime $>2$ , then $p_1$ is always odd. $p_2 = x - p_1$, thus $p_2$ must be odd too. All odd primes are $>2$, thus $p_2>2$. If both $p_1$ and $p_2$ are $>2$, then $x = p_1 + p_2 > 4$. □

Lemma 1 and Lemma 2 show that prime 2 can be used to build one even number only $(4 = 2 + 2)$ and all even numbers $>4$ do not have 2 in their GPs.

## 2 Algorithm

Main loop of an algorithm (referenced later as $A$) is iterating over possible pairs of primes ($p_1$, $p_2$), starting from prime 3 (prime 2 is excluded from calculations because of Lemma 1 and Lemma 2) and collects information about possible sums (which are always even because both primes are odd). On this stage of processing primality test is not directly required because both numbers, $p_1$ and $p_2$, are already primes, however primality test may be required if $A$ is asking for $i$-th prime which is unknown yet.

Let's define an even number $N_{conf}$ below which all even numbers were already confirmed from $GSC()$ standpoint. It is highly probable that $N_{conf}$ would grow up along with progress of the $A$ because number of GPs for $n$ generally grows with growing $n$. There is no point in doing calculations for any $p_1 + p_2 \leq N_{conf}$ because this range was already confirmed. Assuming that for a given $p_1$ we iterate down over $p_2$ from 3 to $p_1$ (where $2 \times p_1 \geq p_1 + p_2 > N_{conf}$) the most favourable situation would be if after completing all checks for $p_1$ and $p_2$ we have $N_{conf} = 2 \times p_1 = N_{max}$ - next iteration of $A$ would not inherit any backlog.

Presented listings are written in C++ language and are taking advantage of STL::set container.

Listing 1 presents data types and declarations of both methods and global variables used later in the source code. $num\_all\_vf$ is $N_{conf}$, $set\_to\_be\_vf$ is a set containing even numbers to be verified (numbers in this set are $> num\_all\_vf$), $set\_alr\_vf$ is a set of even numbers verified which are greater than $num\_all\_vf$. $num\_all\_vf$ has initial value 4 because we know that $4 = 2 + 2$ (Lemma 1) and continue experiments for primes $>2$ (because of Lemma 2).

Listing 2 presents main program loop with iteration over consecutive pair of primes (their indices are $ip1$ and $ip2$), taking advantage of method $get\_ith\_prime(i)$ which returns $i$-th prime ($get\_ith\_prime(0) = 2$, $get\_ith\_prime(1) = 3, \ldots$).

Listing 1: Declarations and main settings

```
typedef long long unsigned int LLUI;

set<LLUI> set_to_be_vf;
set<LLUI> set_alr_vf;
LLUI num_all_vf = 4

void add_nums_to_be_verified (LLUI);
void rm_nums_to_be_verified (LLUI, LLUI);
```

Listing 3 shows implementation of a method responsible for adding new numbers to the verification set: $add\_nums\_to\_be\_verified()$ - all new even numbers must be greaterr than $num\_all\_vf$.

Listing 4 depicts source code responsible for removing

numbers from further analysis: *rm_nums_to_be_verified*(). This method adds numbers $>N_{conf}$ to *set_alr_vf* and removes numbers $\leq N_{conf}$ from *set_to_be_vf*, updating also *num_all_vf*. Both methods, *add_nums_to_be_verified*() and *rm_nums_to_be_verified*(), are used in Listing 2 and are responsible for the algorithm core logic.

Listing 2: Main program loop

```
int main()
{
  LLUI ip1=1, ip2=1;
  LLUI p1=0, p2=0;
  LLUI num=0;

  while (1)
  {
    p1=get_ith_prime (ip1);
    add_nums_to_be_verified (2*p1);

    for (ip2=1; ip2<=ip1+1; ip2++)
    {
      p2=get_ith_prime(ip2);
      num=p1+p2;
      rm_nums_to_be_verified (num, 2*p1);
    }
    ip1++;
  }
  return 0;
}
```

Listing 3: Adding new number to further analysis

```
// Adds even number to a set of numbers
    to be verified if it is above
    threshold of all even numbers which
    were already verified
void add_nums_to_be_verified (LLUI num)
{
  if (num > num_all_vf)
  {
    for (LLUI k = num_alr_vf + 2; k <=
        num; k += 2)
    {
      auto search = set_alr_vf.find(k);
      if (search == set_alr_vf.end())
        set_to_be_vf.insert(k);
    }
  }
}
```

Listing 4: Cleanup of variables after each iteration

```
// Removes number from a set of already
    verified numbers
void rm_nums_to_be_verified (LLUI num,
    LLUI max_num)
{
  set_alr_vf.insert(num);

  auto search = set_alr_vf.find(num);
  if ( search !=set_alr_vf.end() )
    set_to_be_vf.erase(search);

  // update number below which all
      numbers were already verified
  LLUI min_num = 0;
  if ( !set_to_be_vf.empty() )
  {
    set<LLUI>::iterator it;
    for (it = set_to_be_vf.begin(); it !=
        set_to_be_vf.end(); ++it)
    {
      if (min_num == 0) min_num = *it;
      else if (min_num > *it) min_num = *
          it;
    }

    if (min_num > num_all_vf + 2)
      num_all_vf = min_num;
  }
  else num_all_vf = max_num;

  // remove all numbers below num_all_vf
      from further analysis
  set<LLUI>::iterator itt = set_alr_vf.
      begin();
  for (itt = set_alr_vf.begin(); itt !=
      set_alr_vf.end(); )
  {
    if (*itt < num_all_vf) itt =
        set_alr_vf.erase(itt++);
    else ++itt;
  }
}
```

## 3 Results

All executed experiments are taking advantage of algorithm $A$. One round (iteration) of $A$ equals to all steps done for a single value of $ip1$ in Listing 2, before moving to next value of $ip1$ ($ip1+1$).

Figure 1 depicts cardinality of two sets where $N_{conf} < n \leq N_{max}$: in red - cardinality of a set of even numbers $n$ not yet verified ($S_{nv}$) and, in green - cardinality of a set of verified even numbers $n$ ($S_v$). $S_{nv}$ corresponds to *set_to_be_vf* used in Listings 3 and 4, while $S_n$ to *set_alr_vf*.

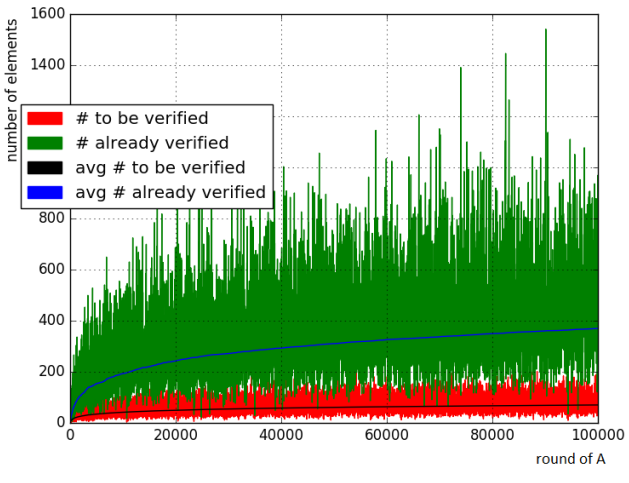


Figure 1: Sum building - first $10^5$ runds of $A$

Figure 2 depicts difference between $N_{max}$ and $N_{conf}$ for first consecutive rounds of $A$. Difference is fluctuating, with periodic ups and downs, but in general is relatively low in comparison to $r$ - for first $10^5$ round of $A$ it is $\leq$ 3500.

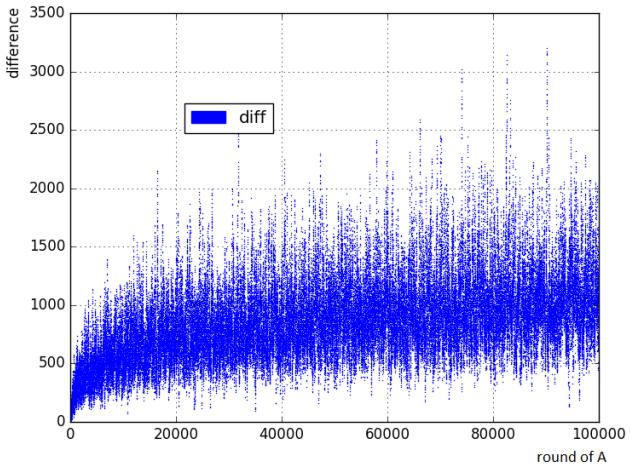


Figure 2: diff($r$) = $N_{max}(r)$ - $N_{conf}(r)$ where $r$ is a round of $A$; results for the first $10^5$ rounds of $A$

As a result of this work interesting integer sequence was submitted to OEIS database: OEIS A301776 [4]. This sequence contains prime numbers $p$ with the property that all even numbers $n$ ($2<n \leq 2p$) are the sum of two primes $\leq p$. Most probably OEIS A301776 is finite and has 7 terms only: $2, 3, 5, 7, 13, 19, 109$ - experiments were run for the first $10^5$ primes and no further terms were found. Figure 3 demonstrates effects of 200 first rounds of $A$ and visualises OEIS A301776. Red line depicts current $N_{conf}$, while green line - theoretical $N_{max}$. If lines are touching each other, we have a new term in OEIS A301776.

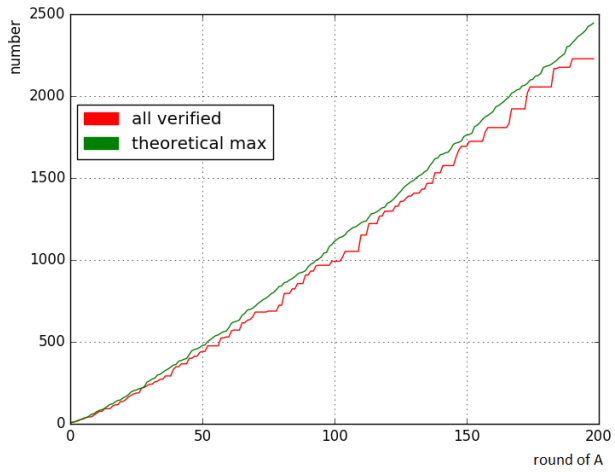


Figure 3: $N_{max}$ (theoretical max) vs. $N_{conf}$ (all verified) - first 200 rounds of $A$

Figure 4 and Figure 5 show that there is correlation between $diff(r)$ and both $S_v(r)$ and $S_{nv}(r)$ (skipping cases from OEIS A301776 for which we have 0 in denominator). After $10^5$ rounds arithmetical average of $\frac{diff(r)}{S_v(r)}$ equals 2.42635613793257 and arithmetical average of $\frac{diff(r)}{S_{nv}(r)}$ is 13.398628763226336. Figure 6 and Figure 7 demonstrate that, in general, the bigger $r$, the smaller delta between two consecutive values of average difference, close to 0.

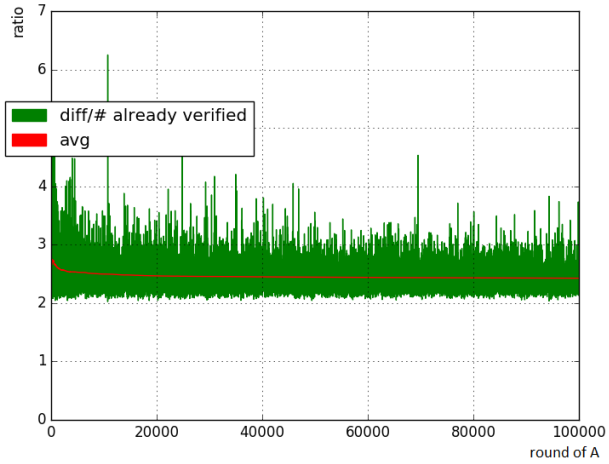


Figure 4: Ratio $\frac{diff(r)}{S_v(r)}$, including average value (avg); results for the first $10^5$ rounds of $A$

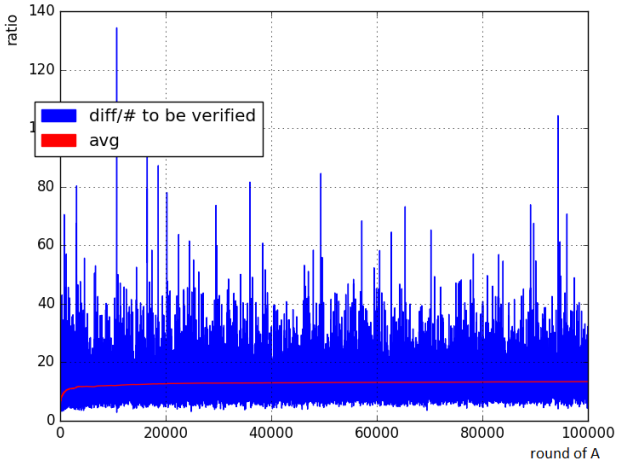


Figure 5: Ratio $\frac{diff(r)}{S_{nv}(r)}$, including average value (avg); results for the first $10^5$ rounds of $A$

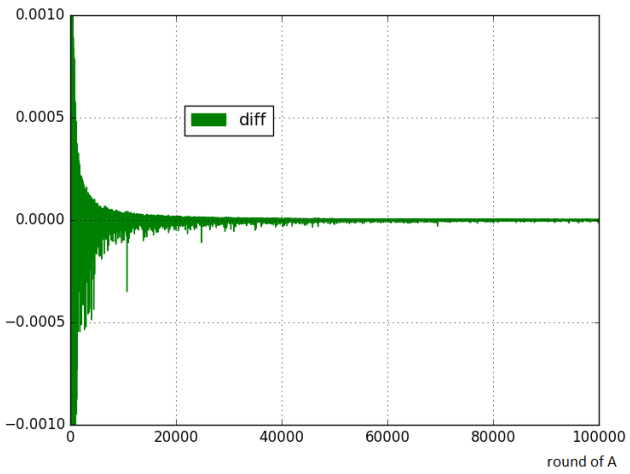


Figure 6: Consecutive difference in average of $\frac{diff(r)}{S_v(r)}$; results for the first $10^5$ rounds of $A$

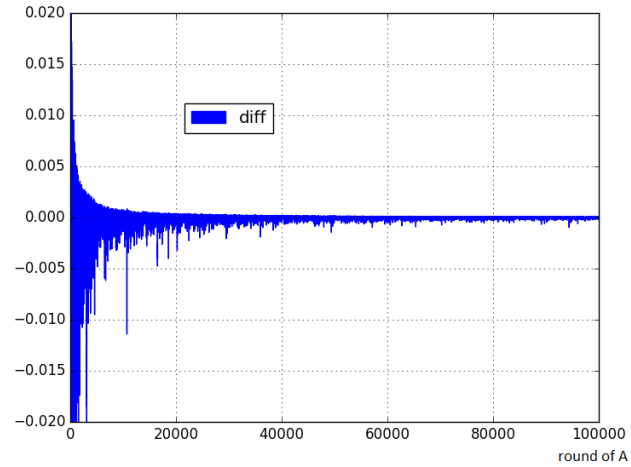


Figure 7: Consecutive difference in average of $\frac{diff(r)}{S_{nv}(r)}$; results for the first $10^5$ rounds of $A$

## 4 Summary

Executed experiments provided data to formulate conjecture that there are only two even numbers >2 (4 and 6) which have GP built from two identical primes: $4 = 2 + 2$, $6 = 3 + 3$ (and both numbers have no other partitions) and all greater even numbers $n$ have at least one partition $GSC(n, p_1, p_2)$, where $p_1 \neq p_2$ (2):

$$\forall_{x>3, x\in\mathbb{N}} \exists_{p_1,p_2\in\mathbb{P}, p_1\neq p_2} GSC(2x, p_1, p_2) \qquad (2)$$

In other words, if even number $n = 2p$>6 (where p is prime), then exists a pair of primes $(p_1, p_2), p_1 \neq p_2$ that $GSC(n, p_1, p_2)$. Conjecture (2) says that all even numbers >6 can be expressed as a sum of two different primes.

It still remains an open question if OEIS A301776 is finite or not. Executed experiments for the first $10^5$ primes confirmed that there are no other terms so far. Figure 2 gives a clue that there are cases for which we might be close to have a new term in the sequence (when difference is falling down, but never to 0 for bigger primes) but then we have ups. Proving that there are other terms in OEIS A301776, especially proving that this sequence is infinite, would be a good base for a proof of *GSC* - if sequence OEIS A301776 is infinite, then there is always a prime $p$ for which we can build all even numbers up to $2p$ using two primes $\leq p$.

## References

[1] Christian Goldbach, *On the margin of a letter to Leonard Euler*, 1742.

[2] Tomás Oliveira e Silva, *Goldbach conjecture verification.* http://sweet.ua.pt/tos/goldbach.html, 2012.

[3] Tomás Oliveira e Silva, Siegfried Herzog, and Silvio Pardi, *Empirical verification of the even Goldbach conjecture and computation of prime gaps up to* $4 \times 10^{18}$., Mathematics of Computation, vol. 83, no. 288, pp. 2033-2060, July 2014 (published electronically on November 18, 2013.

[4] OEIS Foundation Inc. (2018), The On-Line Encyclopedia of Integer Sequences, http://oeis.org/A301776. Prime numbers p with the property that all even numbers n ($2 < n \leq 2p$) are the sum of two primes $\leq$ p.