Bounds on the Range(s) of Prime Divisors of a Class of Baillie-PSW Pseudo-Primes

Dhananjay Phatak
CSEE Dept. UMBC, Baltimore, MD 21250, USA
phatak@umbc.edu

original write-up : October 2019

last revision : July 14, 2020

Symbols used

$|\dot{\div}|$ $\equiv$ "divides" ; for example: "x $|\dot{\div}|$ y" means "x divides y"

$\triangleq$ $\equiv$ equal to by definition ;
for example: ex: "$x \triangleq y$" means "let x denote y" (or vice versa)

$\Rightarrow$ $\equiv$ "implies" which means a one-way "if-then" implication ;
for example: "x $\Rightarrow$ y" means "x implies y"
or in other words: if x is true then y must be true

ABSTRACT

In the literature [1], Carmichael Numbers that satisfy additional constraints $(p+1)$ $|\dot{\div}|$ $(N+1)$ for every prime divisor p $|\dot{\div}|$ N are referred to as "Williams' Numbers"[1]. In the renowned Pomerance-recipe [2] to search for Baillie-PSW pseudoprimes; there are heuristic arguments suggesting that the number of Williams' Numbers could be large (or even unlimited). Moreover, it is shown [2] that if a Williams' number is encountered during a search in accordance with all of the conditions in that recipe [2] ; then it must also be a Baillie-PSW pseudoprime.

We derive new analytic bounds on the prime-divisors of a Williams' Number.
Application of the bounds to Grantham's set of 2030 primes (see [3]) drastically reduces the search space from the impossible size $\approx 2^{(2030)}$ to less than a quarter billion cases (160,681,183 cases to be exact, please see the appendix for details).

We tested every single case in the reduced search space with maple code. The result showed that there is NO Williams' number (and therefore NO Baillie-PSW pseudo-prime which is also a Williams' number) in the entire space of subsets of the Grantham-set. The results thus demonstrate that Williams' numbers either do not exist or are extremely rare. We believe the former; i.e., that No such composite (i.e., a Williams' Number of this type) exists.

[1] more precisely, "1-Williams Numbers" ; however ; the distinctions between different types of Willliams' numbers are not relevant in this document and therefore, we refer to 1-Williams Numbers simply as Williams' numbers.

1 Background

The renowned Pomerance-recipe [2] to search for a Baillie-PSW pseudo-prime (and the related \$620 open research problem [4]) calls for searching a square-free odd composite N that satisfies several conditions, including the following divisibility conditions:

for each distinct prime divisor P_i of N
i.e.

$$\left.\begin{array}{llll} (i) & P_i & |\div| & N \\ (ii) & (P_i - 1) & |\div| & (N-1) \qquad \text{we call this Eqn. the "}\texttt{nsat}\text{" condition} \\ & \text{and} & & \\ (iii) & (P_i + 1) & |\div| & (N+1) \qquad \text{we call this Eqn. the "}\texttt{psat}\text{" condition} \end{array}\right\} \begin{array}{l}\text{basic}\\ \text{divisibility}\\ \text{conditions}\end{array} \tag{1}$$

Following [1], any square-free odd composite number that satisfies all 3 "basic divisibility conditions" stated above is referred to as a "Williams' Number" in this document.

In general let $n \triangleq$ the number of prime divisors of N, so that

$$N = P_1 \times P_2 \times \cdots \times P_i \times \cdots \times P_n \tag{2}$$

Our end goal is to prove or disprove the new primality conjectures unveiled in [5][2]. To that end, it can be assumed that n is an odd number; and

$$P_1 < P_2 < \cdots < P_i < \cdots < P_n \tag{3}$$

Let the ratios be

$$\begin{array}{llll} \frac{P_1}{P_1} \triangleq r_1 & \Rightarrow & r_1 = 1 \\ \\ \frac{P_2}{P_1} \triangleq r_2 & \Rightarrow & r_2 > r_1 \\ \\ \frac{P_3}{P_1} \triangleq r_3 & \Rightarrow & r_3 > r_2 \\ \vdots \\ \frac{P_j}{P_1} \triangleq r_j & \Rightarrow & r_j > r_{j-1} \\ \vdots \\ \frac{P_n}{P_1} \triangleq r_n & \Rightarrow & r_n > r_{n-1} > \cdots > r_2 > 1 \end{array} \tag{4}$$

Finally, for convenience and backward compatibility with the notation in [5] we define

$$\mathcal{M}_i \quad \triangleq \quad \frac{N}{P_i} \tag{5}$$

[2] That is the overall goal; however; that goal is not the focus of this document; even though this document happened to evolve during the pursuit of that overall goal.

Let d be an odd positive integer in the range[3] $1 \leq d << n$

Then, note that

$$N - 1 = N - (P_i)^d + (P_i)^d - 1 \quad \Rightarrow \tag{6}$$

$$(N-1) \mod (P_i - 1) = (N - (P_i)^d) \mod (P_i - 1) \tag{7}$$

since

$$[(P_i)^d - 1] \mod (P_i - 1) = 0 \tag{8}$$

Therefore

$$\begin{aligned} &(P_i - 1) \quad |\div| \quad (N-1) \quad \Rightarrow \quad (P_i - 1) \quad |\div| \quad [N - (P_i)^d] \\ &\quad \Rightarrow \quad (P_i - 1) \quad |\div| \quad P_i \times [\mathcal{M}_i - (P_i)^{(d-1)}] \\ &\quad \Rightarrow \quad (P_i - 1) \quad |\div| \quad [\mathcal{M}_i - (P_i)^{(d-1)}] \end{aligned} \tag{9}$$

Likewise

$$N + 1 = N - (P_i)^d + (P_i)^d + 1 \quad \Rightarrow \tag{10}$$

$$(N+1) \mod (P_i + 1) = (N - (P_i)^d) \mod (P_i + 1) \tag{11}$$

since

$$[(P_i)^d + 1] \mod (P_i + 1) = 0 \tag{12}$$

Therefore

$$\begin{aligned} &(P_i + 1) \quad |\div| \quad (N+1) \quad \Rightarrow \quad (P_i + 1) \quad |\div| \quad [N - (P_i)^d] \\ &\quad \Rightarrow \quad (P_i + 1) \quad |\div| \quad P_i \times [\mathcal{M}_i - (P_i)^{(d-1)}] \\ &\quad \Rightarrow \quad (P_i + 1) \quad |\div| \quad [\mathcal{M}_i - (P_i)^{(d-1)}] \end{aligned} \tag{13}$$

Note that equations (9) and (13) hold for $d = 1$ and imply that

$$(P_i - 1) \quad |\div| \quad (\mathcal{M}_i - 1) \tag{14}$$

and

$$(P_i + 1) \quad |\div| \quad (\mathcal{M}_i - 1) \tag{15}$$

The preceding two equations imply that

$$\operatorname{lcm}(P_i - 1, P_i + 1) \quad |\div| \quad (\mathcal{M}_i - 1) \tag{16}$$

Then invoking the identity

$$\operatorname{lcm}(x, y) \times \gcd(x, y) = x \times y \tag{17}$$

[3] in this document we only use the single value $d = 1$ so the maximum admissible value of d is immaterial

and the fact that $\gcd(P_i - 1, P_i + 1) = 2$, yields

$$\frac{P_i^2 - 1}{2} \quad |\div| \quad (\mathcal{M}_i - 1) \tag{18}$$

$$\Rightarrow \quad (P_i^2 - 1) \quad |\div| \quad 2 \times (\mathcal{M}_i - 1) \quad \forall\, i \in [1, n] \tag{19}$$

Therefore, from the definition of integer division, we obtain

$$P_i^2 - 1 \leq (2\mathcal{M}_i - 2) \tag{20}$$

$$\Rightarrow \quad P_i^2 \ \leq \ (2\mathcal{M}_i - 1) \ < \ 2\mathcal{M}_i \tag{21}$$

2 Analytic Results

2.1 Case 1 : Ruling out $n = 3$; i.e., there does not exist a Williams' number which is a product of 3 distinct primes [4]

Now consider the case when $n = 3$ so that

$$N = P_1 \times P_2 \times P_3 \tag{22}$$

and

$$P_1 < P_2 < P_3 \tag{23}$$

Note that in this case, Relation (21) for $i = 3$, yields

$$P_3^2 < 2P_1P_2 \tag{24}$$

From the definition of the ratios (Eqns. (4)) we have

$$\frac{P_2}{P_1} = r_2 \tag{25}$$

The preceding two relations therefore yield

$$P_3^2 < 2P_1(r_2P_1) = (2r_2)P_1^2 \tag{26}$$

$$\Rightarrow \quad P_3 < \sqrt{2r_2}P_1 \tag{27}$$

But $P_2 < P_3$ which together with the preceding relation yields

$$P_2 = r_2P_1 < P_3 < \sqrt{2r_2}P_1 \tag{28}$$

$$\Rightarrow \quad r_2P_1 < \sqrt{2r_2}P_1 \quad \Rightarrow \quad \sqrt{r_2} < \sqrt{2} \quad \Rightarrow \quad r_2 < 2 \tag{29}$$

And then,

$$P_3 < \sqrt{2r_2}P_1 \quad \Rightarrow \quad P_3 < 2P_1 \tag{30}$$

Further, since

$$\frac{P_3}{P_1} = r_3 \tag{31}$$

then, it is clear that

$$1 < r_2 < r_3 < 2 \tag{32}$$

Finally, note that Relation (19) for $i = 1$, yields

$$(P_1^2 - 1) \quad |\div| \quad 2 \times (\mathcal{M}_1 - 1) = 2(P_2P_3 - 1) = 2(r_2r_3P_1^2 - 1) \tag{33}$$

[4] This was independently proved using different methods in [1] and also in the references therein.

It is easy to verify (by long division or by direct substitution) that

$$2r_2r_3P_1^2 - 2 = 2r_2r_3(P_1^2 - 1) + 2r_2r_3 - 2 \tag{34}$$

Therefore Relation (33) implies that

$$(2r_2r_3 - 2) \mod (P_1^2 - 1) = 0 \tag{35}$$

$$\Rightarrow \quad P_1^2 - 1 \leq 2(r_2r_3 - 1) \quad \Rightarrow \quad P_1^2 < 2r_2r_3 - 1 < (8 - 1) = 7 \tag{36}$$

which is a contradiction since P_1 is an odd prime $\quad \Rightarrow \quad P_1 \geq 3 \quad \Rightarrow \quad P_1^2 \geq 9 \qquad \square$ (37)

2.2 An analogous proof to rule out product of 5 primes

In this case when $n = 5$ so that

$$N = P_1 \times P_2 \times P_3 \times P_4 \times P_5 \tag{38}$$

and

$$P_1 < P_2 < P_3 < P_4 < P_5 \tag{39}$$

Note that in this case, Relation (19) for $i = 5$, yields

$$(P_5^2 - 1) \quad |\div| \quad 2 \times (\mathcal{M}_5 - 1) \tag{40}$$

$$\Rightarrow \quad [r_5^2 (P_1)^2 - 1] \quad |\div| \quad 2 \times \left\{ 1 \cdot r_2 \cdot r_3 \cdot r_4 (P_1)^4 - 1 \right\} \tag{41}$$

Let

$$\text{Dividend = Numerator} \quad = 2 \times \left\{ 1 \cdot r_2 \cdot r_3 \cdot r_4 (P_1)^4 - 1 \right\} \tag{42}$$

and

$$\text{Divisor = Denominator} \quad = [r_5^2 (P_1)^2 - 1] \tag{43}$$

Note that both the Dividend and the Divisor are polynomials of P_1^2.

By carrying out long-division until the remainder is of the same degree as the divisor (or by direct substitution) it can be verified that

$$2 \times \left\{ r_2 r_3 r_4 (P_1)^4 - 1 \right\} = \underbrace{\left(\frac{2 r_2 r_3 r_4}{r_5^2} \right) P_1^2}_{\text{partial quotient}} \quad \times \quad \overbrace{[r_5^2 (P_1)^2 - 1]}^{\text{Divisor}} \quad + \quad \underbrace{\left(\frac{2 r_2 r_3 r_4}{r_5^2} \right) P_1^2 - 2}_{\text{corresponding partial remainder}} \tag{44}$$

Why stop the division when degree of the remainder is the same as the degree of the divisor ?

Because if the last step which renders
the degree of remainder polynomial $<$ degree of divisor polynomial
is carried out then it can be verified that the final-remainder is

$$\text{final-Remainder} \; = \; \left\{ 2 \cdot \left(\frac{r_2 r_3 r_4}{r_5^4} \right) - 2 \right\} < 0 \tag{45}$$

since

$$\frac{r_2 r_3 r_4}{r_5^4} \; = \; \frac{1}{r_5} \cdot \frac{r_2}{r_5} \cdot \frac{r_3}{r_5} \cdot \frac{r_4}{r_5} < 1 \tag{46}$$

$$\tag{47}$$

In other words, carrying out the long division all the way to the end leads to an over-estimate of the INTEGER value of the quotient and therefore cannot yield non-trivial bounds relating the magnitudes of the partial-remainder and the divisor.

We therefore think that correct bounds can be derived if we take the remainder at the penultimate step of the long-division; wherein; the degree of the partial remainder polynomial is the same as the degree of the divisor polynomial.

Accordingly, Relations (41) and (44) together imply that

$$[r_5^2(P_1)^2 - 1] \qquad |\div| \qquad \left(\frac{2r_2r_3r_4}{r_5^2}\right)P_1^2 - 2 \tag{48}$$

$$\Rightarrow \quad r_5^2(P_1)^2 - 1 \quad \leq \quad \left(\frac{2r_2r_3r_4}{r_5^2}\right)P_1^2 - 2 \tag{49}$$

$$\Rightarrow \quad r_5^2(P_1)^2 \quad \leq \quad \left(\frac{2r_2r_3r_4}{r_5^2}\right)P_1^2 - 1 \quad < \quad \left(\frac{2r_2r_3r_4}{r_5^2}\right)P_1^2 \tag{50}$$

$$\Rightarrow \quad r_5^2 \quad < \quad \left(\frac{2r_2r_3r_4}{r_5^2}\right) \tag{51}$$

$$\Rightarrow \quad r_5^4 \quad < \quad (2r_2r_3r_4) < 2(r_4 \cdot r_4 \cdot r_4) = 2(r_4)^3 \tag{52}$$

$$\Rightarrow \quad r_5 < (2)^{\frac{1}{4}}(r_4)^{\frac{3}{4}} \tag{53}$$

But we also have

$$r_4 < r_5 \tag{54}$$

The preceding two relations imply that

$$r_4 < r_5 < (2)^{\frac{1}{4}}(r_4)^{\frac{3}{4}} \tag{55}$$

$$\Rightarrow \quad r_4 < (2)^{\frac{1}{4}}(r_4)^{\frac{3}{4}} \tag{56}$$

$$\Rightarrow \quad (r_4)^{\frac{1}{4}} < (2)^{\frac{1}{4}} \tag{57}$$

$$\Rightarrow \quad (r_4)^{\frac{3}{4}} < (2)^{\frac{3}{4}} \tag{58}$$

$$\Rightarrow \quad r_5 < (2)^{\frac{1}{4}} \times (2)^{\frac{3}{4}} = 2 \tag{59}$$

Therefore it is clear that

$$1 < r_2 < r_3 < \cdots < r_5 < 2 \tag{60}$$

Finally, note that Relation (19) for $i = 1,$ yields

$$(P_1^2 - 1) \quad |\div| \quad 2 \times (\mathcal{M}_1 - 1) = 2(P_2P_3P_4P_5 - 1) = 2(r_2r_3r_4r_5P_1^4 - 1) \tag{61}$$

Let

$$\text{Dividend = Numerator} \quad = 2 \times \left\{r_2 \cdot r_3 \cdot r_4 \cdot r_5(P_1)^4 - 1\right\} \tag{62}$$

and

$$\text{Divisor = Denominator} \quad = [(P_1)^2 - 1] \tag{63}$$

Note that both the Dividend and the Divisor are polynomials of P_1^2.

By carrying out long-division (or by direct substation) it can be verified that

$$2 \times \left\{ r_2 r_3 r_4 r_5 (P_1)^4 - 1 \right\} = (2 r_2 r_3 r_4 r_5) \times \left(P_1^2 + 1 \right) \times (P_1^2 - 1) \quad + \quad (2 r_2 r_3 r_4 r_5) - 2 \tag{64}$$

Therefore, Relation (61) implies that

$$(P_1^2 - 1) \quad |\div| \quad 2 \times [r_2 r_3 r_4 r_5 - 1] \tag{65}$$
$$\Rightarrow \quad (P_1^2 - 1) \leq 2 \times [r_2 r_3 r_4 r_5 - 1] \tag{66}$$
$$\Rightarrow \quad P_1^2 \leq 2 r_2 r_3 r_4 r_5 - 1 \tag{67}$$
$$\Rightarrow \quad P_1^2 < 2 \times (2 \times 2 \times 2 \times 2) - 1 \tag{68}$$
$$\Rightarrow \quad P_1^2 < 31 \quad \Rightarrow \quad P_1 \leq 5 \tag{69}$$
$$\Rightarrow \quad 2P_1 \leq 10 \tag{70}$$
$$\Rightarrow \quad 5 < P_2 < P_3 < P_4 < P_5 < 10 \tag{71}$$

The preceding relation is a contradiction
since the there is only one prime number between 5 and 10. □

2.3 Similar proof to rule out product of 7 primes

In this case, by hypothesis $n = 7$ so that

$$N = P_1 \times \cdots \times P_7 \tag{72}$$

and

$$P_1 < P_2 < P_3 < P_4 < P_5 < P6 < P_7 \tag{73}$$

Note that in this case, Relation (19) for $i = 7$, yields

$$(P_7^2 - 1) \quad |\div| \quad 2 \times (\mathcal{M}_7 - 1) \tag{74}$$

$$\Rightarrow \quad [r_7^2 (P_1)^2 - 1] \quad |\div| \quad 2 \times \left\{ 1 \cdot r_2 \cdots r_6 (P_1)^6 - 1 \right\} \tag{75}$$

For conciseness and clarity, let

$$\alpha_6 \quad \overset{\Delta}{=} \quad r_1 \times r_2 \times r_3 \times r_4 \times r_5 \times r_6 = \prod_{i=1}^{6} r_i \tag{76}$$

Then in Relation (75) note that

$$\text{Dividend = Numerator} \quad = 2 \times \left\{ \alpha_6 (P_1)^6 - 1 \right\} \tag{77}$$

and

$$\text{Divisor = Denominator} \quad = [r_7^2 (P_1)^2 - 1] \tag{78}$$

Note that both the Dividend and the Divisor are polynomials of P_1^2.

As before, if the full polynomial division is carried out until
the degree of remainder polynomial $<$ degree of divisor polynomial
then it can be verified that the final-polynomial-remainder is

$$\text{final-polynomial-Remainder} \; = \; \left\{ 2 \cdot \left(\frac{\alpha_6}{r_7^6} \right) - 2 \right\} < 0 \tag{79}$$

since

$$\frac{\alpha_6}{r_7^6} \; = \; \frac{r_2 \cdots r_6}{r_7^6} \; = \; \left(\frac{1}{r_7} \right) \cdot \left(\frac{r_2}{r_7} \right) \cdots \left(\frac{r_6}{r_7} \right) < 1 \tag{80}$$

$$\tag{81}$$

In other words, carrying out the long division all the way to the end over-estimates the INTEGER value of the quotient and therefore cannot yield correct bounds on magnitudes.

We therefore think that correct and non-trivial bounds can be derived if we use the remainder at the penultimate step of the long-division; wherein; the degree of the partial remainder polynomial is the same as the degree of the divisor polynomial.

Accordingly, by carrying out long-division until the remainder is of the same degree as the divisor (or by direct substitution) it can be verified that

$$2 \times \Big\{\alpha_6 (P_1)^6 - 1\Big\} = \underbrace{\Big\{\left(\frac{2\alpha_6}{r_7^2}\right) P_1^4 + \left(\frac{2\alpha_6}{r_7^4}\right) P_1^2\Big\}}_{\text{partial quotient}} \quad \times \quad \overbrace{[r_7^2 (P_1)^2 - 1]}^{\text{Divisor}} \quad + \quad \underbrace{\left(\frac{2\alpha_6}{r_7^4}\right) P_1^2 - 2}_{\text{corresponding partial remainder}} \tag{82}$$

Therefore, Relation (75) implies that

$$[r_7^2 (P_1)^2 - 1] \qquad |\div| \qquad \left(\frac{2\alpha_6}{r_7^4}\right) P_1^2 - 2 \tag{83}$$

$$\Rightarrow \quad r_7^2 (P_1)^2 - 1 \quad \le \quad \left(\frac{2\alpha_6}{r_7^4}\right) P_1^2 - 2 \tag{84}$$

$$\Rightarrow \quad r_7^2 (P_1)^2 \quad \le \quad \left(\frac{2\alpha_6}{r_7^4}\right) P_1^2 - 1 \quad < \quad \left(\frac{2\alpha_6}{r_7^4}\right) P_1^2 \tag{85}$$

$$\Rightarrow \quad r_7^2 \quad < \quad \left(\frac{2\alpha_6}{r_7^4}\right) \tag{86}$$

$$\Rightarrow \quad r_7^6 \quad < \quad (2\alpha_6) \quad = \quad 2\prod_{i=2}^{6} r_i \quad < \quad 2\prod_{i=2}^{6} r_6 \quad = \quad 2(r_6)^5 \tag{87}$$

$$\Rightarrow \quad r_7 < (2)^{\frac{1}{6}} (r_6)^{\frac{5}{6}} \tag{88}$$

But we also have

$$r_6 < r_7 \tag{89}$$

The preceding two relations imply that

$$r_6 < r_7 < (2)^{\frac{1}{6}} (r_6)^{\frac{5}{6}} \tag{90}$$

$$\Rightarrow \quad r_6 < (2)^{\frac{1}{6}} (r_6)^{\frac{5}{6}} \tag{91}$$

$$\Rightarrow \quad (r_6)^{\frac{1}{6}} < (2)^{\frac{1}{6}} \tag{92}$$

$$\Rightarrow \quad (r_6)^{\frac{5}{6}} < (2)^{\frac{5}{6}} \tag{93}$$

$$\Rightarrow \quad r_7 < (2)^{\frac{1}{6}} \times (2)^{\frac{5}{6}} = 2 \tag{94}$$

Therefore it is clear that

$$1 < r_2 < r_3 < \cdots < r_7 < 2 \tag{95}$$

Finally, note that Relation (19) for $i = 1$, yields

$$(P_1^2 - 1) \quad |\div| \quad 2 \times (\mathcal{M}_1 - 1) = 2(P_2 P_3 P_4 P_5 P_6 P_7 - 1) = 2(r_2 r_3 r_4 r_5 r_6 r_7 P_1^6 - 1) \tag{96}$$

Let

$$\text{Dividend = Numerator} \quad = 2 \times \left\{ r_2 \cdot r_3 \cdot \cdots \cdot r_7 (P_1)^6 - 1 \right\} \tag{97}$$

and

$$\text{Divisor = Denominator} \quad = [(P_1)^2 - 1] \tag{98}$$

Note that both the Dividend and the Divisor are polynomials of P_1^2.

By carrying out long-division (or by direct substitution) it can be verified that

$$2 \times \left\{ r_2 r_3 r_4 r_5 r_6 r_7 (P_1)^6 - 1 \right\} = (2 \cdot r_2 \cdots r_7) \times \left[(P_1^2 - 1)(P_1^4 + P_1^2 + 1) \right] \quad + \quad (2 \cdot r_2 \cdots r_7) - 2 \tag{99}$$

Therefore, Relation (96) implies that

$$(P_1^2 - 1) \quad |\div| \quad 2 \times (r_2 \cdots r_7 - 1) \tag{100}$$

$$\Rightarrow \quad (P_1^2 - 1) \leq 2 \times (r_2 \cdots r_7 - 1) \tag{101}$$

$$\Rightarrow \quad P_1^2 \leq 2 \times r_2 \cdots r_7 - 1 \quad = \quad 2 \times \left(\prod_{i=2}^{7} r_i \right) - 1 \tag{102}$$

$$\Rightarrow \quad P_1^2 < 2 \times \left(\prod_{i=2}^{7} 2 \right) - 1 \tag{103}$$

$$\Rightarrow \quad P_1^2 < 2^7 - 1 \quad \Rightarrow \quad P_1 \leq \lfloor \sqrt{127} \rfloor \quad \Rightarrow \quad P_1 \leq 11 \tag{104}$$

$$\Rightarrow \quad 2P_1 \leq 22 \tag{105}$$

$$\Rightarrow \quad 11 < P_2 < \cdots < P_7 < 22 \tag{106}$$

The preceding relation is a contradiction
since the there are only 3 primes between 11 and 22. □

2.4 Tighter lower bound on the smallest number of factors a Williams' Number must have can be derived analogously

Note that the same argument will hold for at least few more odd values of
n = the number of distinct prime factors of the composite Williams's' number N.

The exact threshold where the same argument, i.e.,
"there are not enough primes of the type we want in the interval $[P_1, 2P_1)$ " holds; thereby leading to a contradiction ; (in turn completing a proof by contradiction)
can be evaluated in a manner identical to the previous derivations (and the evaluation of the exact value of n up to which the sequence of exactly analogous analytic proofs by contradiction holds; will be completed in the next major version/revision of this document).

While it is useful to establish this absolute lower bound on the number of factors a Williams' number must have; we think that it is more useful to derive bounds on magnitudes of prime divisors that hold for any odd value of the number of factors n; which is done next.

3 Analogous bounds that hold for a Williams' Number with any odd number $n = (2k+1)$ of prime factors

In this case, by hypothesis $n = (2k+1)$ so that

$$N = P_1 \times \cdots \times P_n \tag{107}$$

and

$$P_1 < P_2 < \cdots < P_i < \cdots < P_n \tag{108}$$

3.1 The main new analytic bounds

Next, we derive the following bounds:

Lemma 1 :

$$\left.\begin{array}{c} 1 = r_1 < r_2 < \quad \cdots \quad < r_n < 2 \\ \text{which implies that} \\ P_1 < P_2 < \quad \cdots \quad < P_n < (2 * P_1) \end{array}\right\} \tag{109}$$

and

Lemma 2 :

$$P_1^2 < 2^n - 1 \quad \Rightarrow \quad n \geq \left\lceil \log_2 (P_1^2 + 1) \right\rceil \tag{110}$$

3.2 Proofs/analytic derivations

Note that in this case, Relation (19) for $i = n$, yields

$$(P_n^2 - 1) \quad |\div| \quad 2 \times (\mathcal{M}_n - 1) \tag{111}$$

$$\Rightarrow \quad [r_n^2 (P_1)^2 - 1] \quad |\div| \quad 2 \times \left\{ 1 \cdot r_2 \cdots r_{n-1} (P_1)^{n-1} - 1 \right\} \tag{112}$$

For conciseness and clarity, let

$$\alpha_{n-1} \quad \triangleq \quad r_1 \times r_2 \times \cdots \times r_{n-1} = \prod_{i=1}^{n-1} r_i \tag{113}$$

Then in Relation (112) note that

$$\text{Dividend = Numerator} \quad = 2 \times \left\{ \alpha_{n-1} (P_1)^{n-1} - 1 \right\} \tag{114}$$

and

$$\text{Divisor = Denominator} \quad = [r_n^2 (P_1)^2 - 1] \tag{115}$$

Note that both the Dividend and the Divisor are polynomials of P_1^2.

As before, if the full polynomial division is carried out until
the degree of remainder polynomial < degree of divisor polynomial
then it can be verified that the final-polynomial-remainder is

$$\text{final-polynomial-Remainder} = \left\{2 \cdot \left(\frac{\alpha_{n-1}}{r_n^{(n-1)}}\right) - 2\right\} < 0 \tag{116}$$

since

$$\frac{\alpha_{n-1}}{r_n^{(n-1)}} = \frac{r_2 \cdots r_{n-1}}{r_n^{(n-1)}} = \left(\frac{1}{r_n}\right) \cdot \left(\frac{r_2}{r_n}\right) \cdots \left(\frac{r_{n-1}}{r_n}\right) < 1 \tag{117}$$

$$\tag{118}$$

In other words, carrying out the long division all the way to the end leads to an over-estimate of the INTEGER value of the quotient and therefore cannot yield correct bounds relating the magnitudes of the partial-remainder and the divisor.

We therefore think that correct and non-trivial bounds can be derived if we use the remainder at the penultimate step of the long-division; wherein; the degree of the partial remainder polynomial is the same as the degree of the divisor polynomial.

Accordingly, by carrying out long-division until the remainder is of the same degree as the divisor (or by direct substitution) it can be verified that

$$2 \times \left\{\alpha_{n-1}(P_1)^{n-1} - 1\right\} =$$

$$\underbrace{\left\{\left(\frac{2\alpha_{n-1}}{r_n^2}\right) P_1^{n-3} + \left(\frac{2\alpha_{n-1}}{r_n^4}\right) P_1^{n-5} + \cdots + \left(\frac{2\alpha_{n-1}}{r_n^{(n-3)}}\right) P_1^2\right\}}_{\text{partial quotient}} \times \overbrace{[r_n^2(P_1)^2 - 1]}^{\text{Divisor}}$$

$$+ \underbrace{\left(\frac{2\alpha_{n-1}}{r_n^{(n-3)}}\right) P_1^2 - 2}_{\text{corresponding partial remainder}} \tag{119}$$

Therefore, Relation (112) implies that

$$[r_n^2(P_1)^2 - 1] \qquad |\div| \qquad \left(\frac{2\alpha_{n-1}}{r_n^{(n-3)}}\right) P_1^2 - 2 \tag{120}$$

$$\Rightarrow \quad r_n^2(P_1)^2 - 1 \quad \leq \quad \left(\frac{2\alpha_{n-1}}{r_n^{(n-3)}}\right) P_1^2 - 2 \tag{121}$$

$$\Rightarrow \quad r_n^2(P_1)^2 \quad \leq \quad \left(\frac{2\alpha_{n-1}}{r_n^{(n-3)}}\right) P_1^2 - 1 \quad < \quad \left(\frac{2\alpha_{n-1}}{r_n^{(n-3)}}\right) P_1^2 \tag{122}$$

$$\Rightarrow \quad r_n^2 \quad < \quad \left(\frac{2\alpha_{n-1}}{r_n^{(n-3)}}\right) \tag{123}$$

$$\Rightarrow \quad r_n^{(n-1)} \quad < \quad (2\alpha_{n-1}) \quad = \quad 2\prod_{i=2}^{n-1} r_i \qquad < \qquad 2\prod_{i=2}^{n-1} r_{n-1} \quad = \quad 2(r_{n-1})^{(n-2)} \tag{124}$$

$$\Rightarrow \quad r_n < (2)^{\frac{1}{n-1}}(r_{n-1})^{\frac{n-2}{n-1}} \tag{125}$$

But we also have

$$r_{n-1} < r_n \tag{126}$$

The preceding two relations imply that

$$r_{n-1} < r_n < (2)^{\frac{1}{n-1}}(r_{n-1})^{\frac{n-2}{n-1}} \tag{127}$$

$$\Rightarrow \quad r_{n-1} < (2)^{\frac{1}{n-1}}(r_{n-1})^{\frac{n-2}{n-1}} \tag{128}$$

$$\Rightarrow \quad (r_{n-1})^{(1-\frac{n-2}{n-1})} = (r_{n-1})^{\frac{1}{n-1}} < (2)^{\frac{1}{n-1}} \tag{129}$$

$$\Rightarrow \quad (r_{n-1})^{\frac{n-2}{n-1}} < (2)^{\frac{n-2}{n-1}} \tag{130}$$

$$\Rightarrow \quad r_n < (2)^{\frac{1}{n-1}} \times (2)^{\frac{n-2}{n-1}} = 2 \tag{131}$$

Therefore it is clear that

$$1 < r_2 < r_3 < \cdots < r_n < 2 \tag{132}$$

which completes the proof of Lemma 1 $\square$

A more direct way to arrive at the same bound in Lemma 1 is as follows:
Relation (124) is

$$r_n^{(n-1)} \quad < \quad (2\alpha_{n-1}) \quad = \quad 2\prod_{i=2}^{n-1} r_i$$

Then, since

$$r_i \leq r_n \quad \text{it follows that} \quad \prod_{i=2}^{n-1} r_i < \prod_{i=2}^{n-1} r_n = r_n^{(n-2)} \tag{133}$$

therefore

$$r_n^{(n-1)} \quad < \quad 2r_n^{(n-2)} \qquad \Rightarrow \qquad r_n < 2 \qquad \square \tag{134}$$

Finally, note that Relation (19) for $i = 1,$ yields

$$(P_1^2 - 1) \quad |\div| \quad 2 \times (\mathcal{M}_1 - 1) = 2(P_2 \cdots P_n - 1) = 2(r_2 \cdots r_n P_1^{(n-1)} - 1) \tag{135}$$

Let

$$\text{Dividend = Numerator} \quad = 2 \times \Big\{ r_2 \cdot \cdots \cdot r_n (P_1)^{n-1} - 1 \Big\} \tag{136}$$

and

$$\text{Divisor = Denominator} \quad = [(P_1)^2 - 1] \tag{137}$$

Note that both the Dividend and the Divisor are polynomials of P_1^2.

By carrying out long-division (or by direct substitution) it can be verified that

$$\begin{aligned} 2 \times \Big\{ r_2 \cdots r_n (P_1)^{n-1} - 1 \Big\} = (2 \cdot r_2 \cdots r_n) \times \Big[(P_1^2 - 1)(P_1^{(n-3)} + P_1^{(n-5)} + \cdots + + P_1^2 + 1) \Big] \\ + \quad (2 \cdot r_2 \cdots r_n) - 2 \end{aligned} \tag{138}$$

Therefore, Relation (135) implies that

$$(P_1^2 - 1) \quad |\div| \quad 2 \times (r_2 \cdots r_n - 1) \tag{139}$$

$$\Rightarrow \quad (P_1^2 - 1) \leq 2 \times (r_2 \cdots r_n - 1) \tag{140}$$

$$\Rightarrow \quad P_1^2 \leq 2 \times r_2 \cdots r_n - 1 \quad = \quad 2 \times \left(\prod_{i=2}^{n} r_i \right) - 1 \tag{141}$$

$$\Rightarrow \quad P_1^2 < 2^n - 1 \tag{142}$$

which completes the proof of Lemma 2 $\square$

4 Application of the bounds to Grantham's set of 2030 primes: There is NO Williams' Number in the entire space of approximately $2^{(2030)}$ subsets

Applying the bounds in Lemma1 and Lemma2 to the set of 2030 primes by Grantham and Alford [3] reduces the size of the search space drastically. Let

$$G := [5003, 5987, \cdots, 5360297573804957369123, 10659431812315035515387] \qquad (143)$$

be the array of primes in the Grantham-Alford list. We processed the array as per the pseudo-code below.

```
For i from 1 to 2030 do
    P1 := G[i] ;
    Pot_Viable_Set := {P1} ; // Assume that each prime is the smallest element
                             // of a new/distinct potentially viable set
  // first apply Lemma 1
    for j from i+1 to 2030 do
        if G[j] < 2 * P1 then
           append G[j] to the Pot_Viable_Set ;
        else
           break ; // all primes that satisfy Lemma 1 for current value of P1
                      are listed the set
        end if ;

    end of for on j

  // now apply Lemma 2

    n_max := max_num_of_factors_in_Viable_Set := cardinality_of_Pot_Viable_Set ;
    n_min := min_num_of_facrors_as_per_lemma_2 := $\lceil \log_2 (P1^2 + 1) \rceil$
    if n_max < n_min then
       // Lemma 2 is violted so the Pot_Viable_Set is actully not viable
       no op ; // do nothing so that the flow control will go to next P1 value
    else
       Viable_Set := Pot_Viable_Set ;
       Further_Analyze(Viable_Set) ; // Brute-force test every viable sub-set
                                     // of the Viable_Set
    end if ;
end of for on i
```

The actual Maple code for all relevant functions appears in the appendices.

The overall computation was carried out in 2 phases/rounds:

4.1 First round of computations: only enumerate viable-sets and evaluate the exact total number of values that must be tested numerically

In the first round, we only evaluated the exact total number of cases, i.e., distinct values of N that survive the bounds in Lemmas 1 and 2. As mentioned in the abstract, it turns out to be 160,681,183 which is less than a quarter billion "survivor" values of N.

The first Appendix enumerates some of the viable sets (which are subsets of the Grantham's generator set containing the 2030 prime numbers) that survive after the application of Lemma 1 and Lemma 2. It is a verbatim inclusion of excerpts from the screen-capture of the output of a run of the program to fully enumerate each one of the viable sets along with the essential parameters associated with that viable set.

The Maple source code for the main/caller program for the entire first round of computations appears on pages 29, 30 and 31. (A script prints out the source code followed by a screen capture of execution of that code in Maple. Note that Maple code for built-in Library functions, as well as other custom functions that we developed is excluded for the sake of brevity.[5]).

The actual outputs from the run start on Page 31. The very first entity printed out is an array containing the original list of Grantham and Alford (it starts on page 31 and ends on page number 50).

After a few initializations on page 50; from page 51 onward, there is a set of lines printed for each number in the Grantham-list.

A set of lines is delimited from the next set by the dashed-line between the quotes:

```
"
     --------------------------------
    "
```

For instance, the first 4 lines on page 51 correspond to the first prime number = 5003 in the Grantham-list.
The last (i.e., fourth) line in this first set is

```
i= 1, P1= 5003, min_n_factors= 25 , nprimes in (p1, 2p1)= 3
```

Obviously `i = 1` denotes the index of the current value of $P1$ which is 5003.

The last field in that line indicates that the set of primes between $P1 = 5003$ and $2 * P1 = 10006$

[5] However, as repeatedly mentioned throughout this document, the author would be glad to make the source code and every data item generated available to anyone upon request. All the s/w and the run-logs have been uploaded on a server. The size is way too big and therefore full printouts cannot be included in any document. Accordingly, only the main results are included in the appendices

includes 3 primes from the list = $\{5987, 6563, 9803\}$ and therefore survives the bound in Lemma 1. We call a non-empty set that survives Lemma 1 as a Potentially_Viable_set.

However; Lemma 2 is violated because the minimum number of factors n for P1 = 5003 turns out to be $n = 25$ which is way bigger than the cardinality of the potentially viable set which is 3 in this case.

Thus the fourth line in the first set is a concise summary showing that a non-empty set does survive past Lemma 1 as a Potentially_viable_set but that set does not conform to Lemma 2. The end result is that the first prime number in Grantham's list does not generate any viable sets wherein that prime (i.e. 5003) is the smallest one = P1 in the viable set.

——-

The next set of four lines shows the same end result for the 2-nd number = 5987 (which has the index `i = 2`) in the Grantham-list: it does not generate any viable sets.

——-

It turns out that the first 450 numbers in the Grantham-list do not yield any viable sets that survive both Lemma 1 and Lemma 2.

For the sake of brevity we skip all those printed line-sets and directly jump to the finding of the first viable set which appears on page 52.

——-

As seen on page 52; the printout for a viable set includes the same four lines that a non-viable set summary includes. However, the viable set printout has additional lines (beyond the first 4 lines) which contain all the information about that viable set.

The 5-th line in this set of printed lines lists the current value of $P1$ followed by the number of viable sets found thus far (which therefore serves as the index of the current viable set). In this case, it happens to be the first viable set.

The variable referred to as "`myDelta`" on the next line is extremely important and is defined by the relation

`myDelta = no_of_primes_between_p1_and_2p1 - (min_n_factors - 1)` (144)

It is also clear that

`(Cardinality_of_the_viable_set) = 1 + no_of_primes_between_p1_and_2p1` (145)

because when we consider all possible subsets of a viable set; we must keep the first element of every subset fixed at the current value of value $P1$. Therefore, all of the distinct subsets of the current viable set are generated by selecting desired number of elements from the list of primes between $P1$ and $2 * P1$ as long as the total number of factors (also including the additional first factor $P1$) is Greater than or equal to the minimum number of factors dictated by Lemma 2.

The variable "`myDelta`" therefore denotes the maximum number of elements that can be skipped (i.e., NOT included) in any subset of the current viable set.

For the current (i.e., the first) viable set it turns out that
the `no_of_primes_between_p1_and_2p1 = 64`
However; the `min_n_factors = 65`
Therefore, $P1$ and all the 64 other primes in the open interval $(P1, 2 * P1)$ must be included in all viable sub-sets of the first viable set. Thus no prime from this viable can be skipped when creating subsets.
Accordingly, `myDelta = 0` and therefore there is only subset = the entire viable set itself.

Therefore the total number of cases generated by the first viable set is

binomial_coefficient = 64_choose_64 = $({}^{64}\boldsymbol{C}_{64}) = ({}^{64}\boldsymbol{C}_{0})$ = 64_choose_0 = 1

A key point to note is that the full name of the array representing each viable set has all the relevant information in it.
For example; consider the name "`Viable_set_1_4708691963_64_0`"
The number following the second underscore character "_" in the name is the index of the viable set found ; which happens to be "1" here.
The number following the 3-rd underscore character is the value of the smallest prime number = $P1$ in the viable set (= 4708691963 in this case)
The number next number in the name is the number of primes in the open interval $(P1, 2*P1)$ which is 64 in this case;
and finally
The last number in the name is the value of `myDelta` for the viable set (happens to be 0 for the case at hand)

Note that the lower bound on the number of factors can be evaluated either using the value of $P1$ in Lemma 2 ; or by using the 3-rd and 4-th numbers in the name and using Eqn. (144) .

The list between the square braces is the full enumerated viable-set cast as a 1-D array in Maple.

This completes the description of the set of printed lines for viable set number/index 1.

For clarity we resume the description of the ensuing parts of the run-log on the next/new page.

——-

The next set of 4 printout-lines (near the top of page no. 53) indicates that there is NO viable set for the next prime (index = 452) in the Grantham-list.

—–-

However, the following prime (index = 453) does generate a viable set (the second viable set found and hence the number/index = 2).
It is a mere coincidence that the value of `myDelta` for this set is again 0 (like the first set).

—–-

The next set of printout-lines describes the 3-rd viable set encountered. Again, `myDelta = 0` for this set. As a result, this set also generates a single case.

—–-

The following set of printed-lines corresponds to the 112-th viable set found at prime indexed 844 in the Grantham-list. We directly jumped to this viable set because it generates the maximum number of sub-sets or equivalently, number of individual values of N that must be tested numerically.
Note that for this set,
(i) the number primes in the open interval $(P1, 2*P1) = 82$
and
(ii) `myDelta = 5`
Therefore, the sub-sets can exclude up-to 5 numbers from the 82 in the open interval dictated by Lemma 1

Therefore the number of cases generated by this viable set is

$$\begin{aligned}\texttt{N\_cases\_for\_this\_set} &= \left[({}^{82}\boldsymbol{C}_5) + ({}^{82}\boldsymbol{C}_4) + ({}^{82}\boldsymbol{C}_3) + ({}^{82}\boldsymbol{C}_2) + ({}^{82}\boldsymbol{C}_1) + ({}^{82}\boldsymbol{C}_0)\right] \\ &= 27285336 + 1749060 + 88560 + 3321 + 82 + 1 \\ &= 29,126,360 \qquad (146)\end{aligned}$$

—–-

The next set of printed-lines (which begins at approximately the middle of page no. 56) shows that the next prime (index = 845) also leads to a viable set (the 113-rd viable set found).

The total number of cases generated by this viable set is

$$\begin{aligned}\texttt{N\_cases\_for\_viable\_set\_no\_113} &= \left[({}^{81}\boldsymbol{C}_4) + ({}^{81}\boldsymbol{C}_3) + ({}^{81}\boldsymbol{C}_2) + ({}^{81}\boldsymbol{C}_1) + ({}^{81}\boldsymbol{C}_0)\right] \\ &= 1663740 + 85320 + 3240 + 81 + 1 \\ &= 1,752,382 \qquad (147)\end{aligned}$$

—–-

It turns out that the following prime (index = 846) also yields a viable set (the 114-th viable set found). This set also generates 29,126,360 cases.

——-

Next we directly jump to the print sets for the last 3 viable sets which happen to be 121-st, 122-nd and 123-rd viable sets found, corresponding to primes with indexes 853, 854 and 855 respectively.

——-

Then we jump to the last two iterations of the outermost loop, corresponding to indexes = 2028 and 2029 respectively. These last two primes do not yield viable sets.

——-

The final block of printed-lines shows that the overall total number of values of N that survive both Lemma 1 and Lemma 2 is 160,681,183. Therefore 160681183 distinct values of N must be tested numerically which turns out to be a task well within the computing capabilities of ordinary desktops.

——-

4.2 Round 2 of computations: numerically testing each survivor value of N

Once it was determined that the overall number of cases that survive past Lemmas 1 and 2 is within the computing capacity of even ordinary desktop machines;
then we implemented Maple code to further analyze in depth, every possible survivor value of N that is generated by the (corresponding/mother) viable set.

In the second round of computations, the entire code was executed as one huge job which ran for over a day. Worse yet, it turns out that the output of the screen-capture is unusually large: over 45 Giga Bytes.

Therefore Appendix 2 shows only the relevant portions from the screen-capture of the run.

The initial part includes the Maple code for the function `analyze_viable_set`[6]. The printout of the source code of the function to analyze a viable set starts on page number 64 and ends on page 69.

That is immediately followed by the source code of the main/overall program (the file name is “`Bigrun-Grantham.mpl`”; the listing starts at about mid way on page number 69 and ends on page 71).

[6] The author is aware that the Maple code is not the most efficient implementation. However because the total number of cases that survive past Lemmas 1 and 2 turns out to be a relatively small, manageable number ; a bit of inefficiency in the implementation was deemed acceptable in return for clarity and correctness. I hope that the readers will agree with this choice of clarity over a small amount of incremental efficiency.

The next few lines (after the end of source code) show the execution of initializations in the main program.

Then the next block shows that the first two primes do not generate a viable set and therefore there is nothing to test numerically.

Then we directly jump to the block for the first viable set (it starts on page no. 73 and ends on page 75). Since `myDelta = 0` for this viable set, there is only one survivor value of N ; including the product of ALL primes in viable set 1.

The last line of that block of printed-lines says

“`--»» for this set williams = 0, carms = 0, psats = 0`”

indicating that for this entire set; out of all the survivor values of N generated by this viable set; not a single one was a William's number.
Nor was there a single Carmichael number in the survivors from this viable set.
Likewise not even a single value of N was found to satisfy the third divisibility constraint
$\forall P_i \quad |\dot{\div}| \quad N \quad ; \quad (P_i+1) \quad |\dot{\div}| \quad (N+1)$
that a Williams' number must satisfy.

The next four lines show that the prime (with index = 452) does not produce a viable set.

We then directly jump the printed-lines-block for the 7-th viable set. The reason is because all the prior viable sets found (numbered 1 thru 6) have a `myDelta = 0` and therefore lead to only one survivor N value that includes the product of all the primes in the generator viable set. We therefore think that showing the printed-lines-block for the first viable set is sufficient to illustrate our methods and the results obtained. There is no point in repeating analogous print-blocks for every set.

The 7th viable set is generated by prime indexed 464 in Grantham List. It has a `myDelta = 1` and cardinality of 66 (including $P1$); and the minimum number of factors required is 65. Therefore this set generates
$({}^{65}\boldsymbol{C}_0) + ({}^{65}\boldsymbol{C}_1) = 65 + 1 = 66$ values of N that survive (Lemmas 1 and 2).

The last two lines in this block are

```
 *** proc analyze_viable_set exit checksum = 66

--->>>> for this set williams = 0, carms = 0, psats = 0
```

showing that not a single value (out of the total of 66 survivor values of N generated by this (the 7-th) viable set met even one of the two divisibility conditions that each prime divisor of a Williams' number must satisfy.

The printed-lines-blocks for the rest of the viable sets are identical but are large because the number of survivor N values generated by some of the sets is big (for instance, as illustrated in the previous sub-section, viable set number 112 and 113 each generate 29,126,360 survivor values. It is pointless and infeasible to reproduce 5 lines for each of those values in this article.

However, the run-logs are available via web upon request. In order to prevent an even bigger size explosion, instead of printing every survivor value of N (each of which is longer than about 750 decimal digits like the huge number below

```
Totprod including p1 = 2267252761446411903518519694376559052088007254121080772-
1482025387395089961887033324464114740374346534720292248645224085529368503887 30-
24388673586577751109182676535452153218291110686474695830633679174511252704895 7-
64951695674969974865283645693557820426402823657300941901858890292292010180935 9-
99046357040234249461574873568562948802987869788502943927787649433177510511648 4-
81115220923155058732039476935363189698085389957798151239268791055240778757397 0-
64924383684581994033212148796177648069909413687642946759256234256471268368864 7-
19966342485059705610716212955404295933878609619565031794148356694947212402211 5-
9590232437905770140207966355583883012674572045048 3889
```

) ; we have printed out an MD5 hash of the decimal digit string representing each survivor value of N. The hash-strings are substantially smaller than the decimal strings and are sufficient for cross-checking and validation.

Finally we directly jump to the end block with few lines stating the final values of what the search found.

The most striking observation is that there is not a single Williams' number. In other words, we have demonstrated that the entire space of $2^{(2030)}$ subsets of Grantham's master set does not include even a single Williams' number ;
which ; in turn implies that
Not even a single Baillie-PSW pseudoprime (which is also a Williams' number)
exists in the entire space of $\approx 2^{(2030)}$ distinct N values generated by Grantham's master Set.

Even more striking is the observation that for every survivor value of N:
not a single one of its prime divisors satisfied the "`nsat`" divisibility condition.

AND in addition

not a single one of the prime divisors of N satisfied the "`psat`" divisibility condition.

This extreme observation led me to re-check all the derivations and equations multiple times. Most recently, a Ph.D. candidate in the cyber defense lab (CDL) in our dept. also independently checked every single equation for correctness.[7]

The results suggest that Williams' numbers either do not exist or are extremely more rare than what the currently available heuristics suggest. Maybe further analysis could reveal more constraints (like Lemmas 1 and 2) such that all of those constraints cannot be satisfied together; thereby furnishing an analytic proof that Williams' Numbers do not exist. I hope that peers will jump in and resolve this issue in the near future.

5 Discussion and Concluding Remarks

The obvious to-do-next list of items is the following

1. First and foremost, see if the analysis in this article can be modified/extended to cover the two other similar master sets designed by Chen and Green [6, 7]

The basic divisibility conditions assumed by Chen and Green are less restrictive than those that define a Williams' number. Accordingly, It is not immediately clear whether the methodology that worked to establish the bounds for Williams' Numbers will require substantial modifications or whether it won't get carried over at all, even partially.

2. Derive the tighter lower bound on the minimum number of factors a Williams' number must have.

3. Investigate further if there is any way to prove what I suspect/speculate: There does not exist a Williams' Number.

4. Or try to prove that a Williams' Number exists.

5. Then get back to the investigation of whether a Baillie-PSW pseudoprime (which is not a Williams' number) exists or not.

[7] see the acknowledgment

In the end I would like to mention that I am mainly interesting in proving or disproving the "primality conjectures" that I have proposed recently [5]. All the work presented in this document unfolded during the pursuit of the over-arching goal to prove those conjectures in [5].

Specifically, the baseline test in [5] is similar to the Ballie-PSW test;
with two small (but critical) differences:
(i) the Baillie-PSW method applies the Fermat's little theorem test to one base (for example base = 2); and it applies the quadratic Frobenius test to a different base/discriminant (for example 5 and $\sqrt{5}$).

In contrast, my baseline primality test applies both those checks to the same base (the base = any QNR that can be found quickly).

(ii) Additionally, my baseline test also checks that the Euler Criterion is satisfied for the SAME BASE value.

The generalized primality conjecture constructs polynomials of degree bounded by $\log N$ whose roots do not exist as integers modulo-N and then applies the the Euler Criterion check and the Binomial Congruence test at the irrational (non-integer) values of the argument,
since the Modular Binomial congruence
$(1+x)^N \mod N \quad \equiv \quad 1+\left(x^N \mod N\right)$
holds for any transcendental/real value of x.

The intuition is clear: if the binomial congruence test holds even for a single NON integer value of the argument x then it is extremely highly likely that the identity worked out because N is a prime number.

This intuition lead to the entire focus on "explicit and implicit" non-residues; i.e.; values that do not exist as modulo-N integers; and therefore when used in place of the indeterminate x in the Binomial Congruence Check (BCC) should be able to discriminate a prime from a composite in a single shot, i.e. a single modular exponentiation.

Acknowledgment

Ph.D. Student Mario Costa in the Cyber Defense Lab (headed by my colleague Prof. Alan T. Sherman) in our department has independently verified all the derivations and equations in this document. The author thanks Mario for his help.

$***$ This document is a draft of the work in progress.

Bibliography

[1] R. J. McIntosh and M. Dipra, “Carmichael numbers with $p+1|n+1$,” *Journal of Number Theory*, vol. 147, pp. 81–91, 2015.

[2] C. Pomerance, “Are there counter-examples to the baillie-psw primality test,” *Dopo Le Parole aangeboden aan Dr. AK Lenstra. Privately published Amsterdam*, 1984, also available at the url. [Online]. Available: https://math.dartmouth.edu/~carlp/dopo.pdf

[3] J. Grantham, “Pseudoprimes/probable primes,” Web page last modified in approximately 2005, The url for the web-page is
http://pseudoprime.com/pseudo.html

The actual plain-text list of 2030 primes that generate the humungous/impossible search-space of size $\approx 2^{(2030)}$ that should include some Baillie-PSW pseudo-primes is available via this page at the url
http://pseudoprime.com/primes620.txt.

[4] R. Crandall and C. B. Pomerance, *Prime numbers: a computational perspective*. Springer Science & Business Media, 2006, vol. 182.

[5] D. S. Phatak, et. al., “PPT : New Low Complexity Deterministic Primality Tests Leveraging Explicit and Implicit Non-Residues.”
August 2019,
The overall document is a set of 3 companion articles available via the following arXiv url. [Online]. Available: https://arxiv.org/abs/1908.06964

[6] Z. Chen and J. Greene, “Some comments on baillie-psw pseudoprimes,” *Fibonacci Quarterly*, vol. 41, no. 4, pp. 334–344, 2003.

[7] J. R. Green, “Generator sets that could lead to baillie–psw pseudo primes,” Web page last modified in approximately right after 2003,
The url for the web-page is
https://www.d.umn.edu/ jgreene/baillie/Baillie-PSW.html

The authors have compiled two lists that are refined over and analogous to the Grantham-Alford List we have processed in this article.

6 Appendixes : Excerpts from screen-capture-logs of numerical testing of the entirity of the reduced search space

6.1 Appendix 1 : First Round of Computations : only enumerate viable-sets and evaluate the exact total number of values that must be tested numerically

```
>>>>>>>>>>  source code  <<<<<<<<<<<<
 --- file = printout-viable-grantham-subsets-and-estimate-total-ncases.mpl ---
Grantham_BPSW_prob_PSPs_set := [5003  ,
5987  ,
6563  ,
9803  ,
10427  ,
  .    ,
  .    ,
  .    ,
4665648301718926194203  ,
5360297573804957369123  ,
10659431812315035515387
] ;

Ncases := nops(Grantham_BPSW_prob_PSPs_set) ;

n_viable := 0 ;
max_in_viable_set := 0  ;
my_maxdelta := 0 ;

Global_N_values := 0 ;

for i from 1 to (Ncases - 1) do
    print("\n ------------------------------- \n") :

    P1 := Grantham_BPSW_prob_PSPs_set[i] ;

    Pmax := 2 * P1 ;
    min_n_factors := ceil( evalf[2*length(P1)](log[2](1 + P1^2)) ) ;

    no_of_primes_between_p1_and_2p1 := 0 :

    for j from (i+1) while ( (j <= Ncases) and
                           (Grantham_BPSW_prob_PSPs_set[j] < Pmax) ) do
     no_of_primes_between_p1_and_2p1 := no_of_primes_between_p1_and_2p1 + 1 ;
    od :

    printf("i= %d, P1= %d, min_n_factors= %d , nprimes in (p1, 2p1)= %d\n\n",
           i, P1, min_n_factors, no_of_primes_between_p1_and_2p1) ;

    if ( no_of_primes_between_p1_and_2p1 > 10 and
       no_of_primes_between_p1_and_2p1 >= min_n_factors - 1) then

n_viable := n_viable + 1 ;
      printf("*** P1 = %d, set is viable set number = %d \n\n",
        P1, n_viable) ;
myDelta := no_of_primes_between_p1_and_2p1 - (min_n_factors - 1) ;
N_cases_for_this_set := 0 ;
for binomial_arg2 from 0 to myDelta do
    N_cases_for_this_set := N_cases_for_this_set +
                 binomial(no_of_primes_between_p1_and_2p1, binomial_arg2);
od ;
printf(" myDelta = %d, cases to try-out-in-this-set = %d \n\n",
         myDelta, N_cases_for_this_set);

   if no_of_primes_between_p1_and_2p1 > max_in_viable_set then
         max_in_viable_set := no_of_primes_between_p1_and_2p1 ;
   fi ;

        if myDelta > my_maxdelta then
   my_maxdelta := myDelta ;
fi ;

        unassign(’tset’) ;
tset := [P1] ;
for k from 1 to no_of_primes_between_p1_and_2p1 do
  tset := [op(tset), Grantham_BPSW_prob_PSPs_set[i+k] ];
od :
printf(" \n Viable_set_%d_%d_%d_%d :=  \n",
            n_viable, P1, no_of_primes_between_p1_and_2p1, myDelta) ;
print(tset) ;
printf(" ;\n\n") ;

        Global_N_values := Global_N_values + N_cases_for_this_set ;


  fi ;

od ;

printf(" *** n_viable sets = %d \n\n", n_viable);
printf(" *** max_in_viable_set = %d \n\n", max_in_viable_set) ;
printf(" *** Max myDelta = %d \n\n ", my_maxdelta) ;

printf(" *** Total no of N values that must be tested numerically = %d \n",
             Global_N_values) ;


 >>>>>>>>>> end of source code  <<<<<<<<<<<<
```

```
set MYMAPLE = /home/phatak/maple2017/bin/maple

and now the execution snapshot
start time = Tue Feb  4 15:26:06 EST 2020

the nohup maple run status = 0

Grantham_BPSW_prob_PSPs_set := [5003, 5987, 6563, 9803, 10427, 11027, 11867,

    16763, 19403, 22283, 22907, 24923, 25667, 29867, 35747, 40427, 40763, 41243,

    42083, 49307, 54323, 54347, 57203, 57347, 66587, 67307, 73883, 78203, 84347,

    104003, 112067, 121403, 122267, 122363, 123323, 124067, 129083, 129443,

    134507, 177467, 178187, 192323, 198827, 203627, 227147, 265163, 281243,

    281867, 315083, 321947, 365147, 381443, 411347, 455627, 472643, 479027,

    598187, 609443, 707027, 710987, 809507, 845003, 867947, 893147, 998843,

    1051763, 1072763, 1074707, 1136123, 1140803, 1185827, 1195907, 1295027,

    1420667, 1525067, 1558307, 1587563, 1784603, 1856747, 2042123, 2183507,

    2191067, 2583083, 2689403, 3409283, 3525563, 3994763, 4015883, 4172243,

    4274027, 4462883, 4679387, 4902587, 4999187, 5696123, 5751107, 5929307,

    5933987, 6208067, 6245963, 6252227, 6302603, 6401987, 6671747, 7119227,

    7326827, 7340867, 7390907, 7674467, 8072123, 8276867, 8479547, 8605403,

    8695187, 8854883, 8972627, 9197003, 9462227, 9693947, 9700883, 10288403,

    10352603, 12005363, 12137243, 12353147, 12551003, 12874427, 13700987,
    13737707, 14472803, 15491387, 15619907, 16846763, 17182283, 17437187,
    18123467, 18666563, 18789587, 19821107, 19852067, 20076083, 20657027,
    21484187, 21750467, 21834203, 24449627, 24508643, 25174043, 25882667,
    26344667, 27020867, 27819923, 28378787, 29371187, 29810483, 29978603,
    32416667, 32604563, 33474803, 33622427, 33650963, 34130123, 34442627,
    35306147, 35744243, 36012947, 37113443, 37516307, 38694347, 39020603,
    39053123, 39086483, 39864347, 44591123, 45865763, 46729283, 47315627,
    51003707, 51329627, 51479243, 52580483, 56976443, 57577067, 58395083,
    59269907, 61672427, 62107883, 62888363, 63670067, 63766763, 66922547,
    68926763, 70637027, 71362763, 71622347, 74854163, 77324147, 78350963,
    79401923, 83650643, 93228827, 93275027, 94179707, 94716827, 95576003,
    96704723, 100076723, 101681243, 102115547, 105058403, 105841403, 107300003,
    108724307, 110852723, 112171523, 114333827, 114692507, 115781363, 115903427,
    121262987, 123096203, 123454523, 126843803, 127047227, 130223147, 135536147,
    135795203, 137093483, 138090707, 144305963, 146173067, 153352163, 153626843,
    154026107, 155269547, 156242123, 157738283, 159113987, 165252107, 167805203,
    172536803, 176518187, 179637083, 182093603, 184790483, 185355323, 193785947,
    196234163, 196606667, 197161403, 197808827, 205489787, 206790707, 213347027,
    221843123, 223346987, 231615827, 234059843, 240885587, 241917827, 244824563,
    248527787, 252466307, 253791443, 257066603, 259049867, 264860987, 267636203,
    271117643, 277523507, 279355403, 285624947, 293241563, 296891027, 300507563,
    308603987, 312352787, 321965867, 325874267, 334969667, 336768947, 339011147,
    342295307, 346595003, 354146603, 356656067, 357607403, 360582683, 361196027,
```

363729083, 373520363, 375424067, 379160003, 390670307, 403476947, 414422843,
415807523, 416531723, 423892523, 426490187, 426927563, 443363363, 450855107,
455230907, 457505747, 460015547, 461525147, 463876643, 473222507, 474692123,
477937067, 478958507, 484603283, 488253083, 501421043, 503639267, 514371707,
516460283, 549668387, 567413723, 576783227, 589857923, 610954427, 619177547,
631332707, 647050067, 671380763, 671775443, 689769467, 698198723, 741513923,
759563027, 760527323, 762596147, 765368867, 773410787, 785569427, 835549763,
846182243, 857011187, 879230483, 913547267, 923395787, 931271723, 941759867,
942577547, 948179987, 956779427, 968867027, 979410563, 983373803, 985607867,
1042057043, 1054536107, 1076958563, 1081879787, 1101930443, 1107852227,
1117127267, 1133485667, 1139255963, 1170554747, 1184969267, 1186597043,
1217688947, 1280799803, 1340255363, 1345056467, 1370534027, 1384129163,
1439565587, 1442558963, 1483832387, 1500979787, 1518523883, 1568588867,
1583134307, 1586152907, 1593956963, 1638246707, 1638279563, 1640450363,
1673498867, 1698578243, 1712121707, 1723037483, 1741472963, 1782299507,
1869339827, 1907889227, 1952265563, 1981211363, 1981686227, 2002702547,
2161363283, 2164332587, 2223821387, 2247384803, 2257182923, 2313900683,
2342335643, 2358285563, 2365924307, 2431111283, 2445382763, 2457692603,
2493548747, 2515505747, 2530852547, 2543139587, 2548304387, 2585622827,
2609674307, 2694000707, 2713215203, 2753719883, 2794046243, 2854921187,
2893877243, 2955515027, 2964009803, 2964577643, 2988618803, 3028146203,
3030236843, 3108495443, 3134256443, 3196293467, 3212474747, 3325160243,
3355611587, 3361707467, 3369832883, 3378789203, 3422391443, 3535300523,

3674441363, 3778275203, 3898069163, 3904512083, 3941727947, 3961815323,
3964367123, 4162171067, 4321112267, 4407440363, 4489950947, 4554695627,
4577360603, 4620088427, 4708691963, 4710995963, 4754130083, 4889392283,
4910078843, 4998584363, 5003629523, 5042545667, 5125415987, 5151593267,
5240243243, 5333689787, 5343444707, 5459704403, 5560240403, 5592303947,
5631606683, 5747783627, 5814384083, 5823545723, 5881089107, 5885109323,
6052388267, 6053200667, 6128580827, 6134338403, 6448080803, 6516763307,
6519138947, 6866836283, 6889292027, 6930553163, 7016120627, 7092744083,
7276231883, 7422712883, 7490008067, 7578493547, 7599012683, 7714374923,
7761591683, 7786043747, 8020986203, 8063123843, 8063438627, 8229446123,
8297435963, 8300117867, 8369939507, 8437088027, 8513676107, 8570641067,
8594608427, 8631134027, 8803105163, 8927440883, 8967039563, 9069641027,
9115211723, 9115855187, 9131776883, 9207476123, 9213008243, 9280410467,
9302901947, 9446845643, 9454452587, 9858736187, 9948847787, 9990625523,
9999012323, 10135853243, 10362267467, 10393347563, 10451207267, 10653747803,
10689483347, 10691597003, 10794517667, 11134161827, 11286550043,
11312941547, 11519200307, 11798205803, 11884439363, 12200325947,
12521956787, 12586462643, 12917675843, 13300058243, 13747853627,
13828077563, 13969017947, 14002976147, 14107702187, 14339106107,
14604332003, 14638838723, 14738355323, 14871859523, 15139602323,
15596627867, 16141024547, 16180934963, 16338933827, 16360973243,
16823182403, 16880215043, 16954379747, 17056960667, 17191871003,
17334273347, 17385021443, 18335189003, 18342030227, 18647320163,

18733506347, 19775714747, 19833840683, 20368490603, 20565105467,
20589672707, 21057947987, 21117748787, 21219047387, 21573284963,
22218640883, 22527606587, 22917099323, 23118302027, 23375098763,
23580332387, 23923588643, 23945201363, 24018493547, 24392140163,
24985186883, 25866345467, 26506168523, 27003107123, 27293562323,
27664318523, 27741985163, 28689361403, 28690012283, 28712103083,
28923095963, 29340641483, 29689750403, 29712899507, 29782788587,
29958583643, 30432597947, 31085005643, 31107483083, 31358465123,
31415143787, 31768665443, 31941266363, 32475836843, 32636181107,
32670887003, 34555999547, 35048520323, 35296832003, 35746771763,
35796377003, 36810435923, 37564142747, 37594450427, 37606204187,
37922832107, 38773348883, 39191947283, 41209653203, 41570885123,
41657960987, 41658585443, 41779795787, 42075259763, 42249648923,
42757295387, 42778691603, 43201394387, 44597952083, 44664257003,
44853972347, 45359601587, 45782696627, 46107829907, 46412869043,
46609601387, 46795294643, 46919227427, 48146296043, 49374402587,
49466006963, 49751734787, 49830414947, 49990649147, 50241747323,
50617402763, 50684708483, 50856202907, 51212522267, 51441771083,
53185590083, 53350350947, 54568358963, 54656463707, 55303780787,
55872613187, 56925121787, 56980887347, 57452276363, 57604693403,
58058642123, 59644363403, 59923040867, 60172250003, 61061968883,
61661532323, 62029379267, 62093447723, 62556898907, 63699450803,
64603919747, 64644871763, 64667429243, 67522523147, 68087211947,

68392313387, 68882171243, 69206712803, 69342422147, 69683830523,
69915194747, 71914894523, 74000769083, 74188243667, 74823927947,
75234473747, 76713317243, 77585913107, 78546193883, 79612442027,
81589719203, 82008857867, 82404490283, 83035387427, 84038101163,
84763405187, 85662176003, 87288931043, 87980005643, 88223306867,
89923686323, 90271937627, 90313967603, 93041693027, 93828985403,
95481516227, 95637508643, 95815522547, 96323604683, 97003443203,
97212896483, 97225612427, 97649446667, 98151599963, 99402680243,
99409344467, 99923410187, 100477976843, 101219449547, 102688601003,
102866924267, 103061359547, 104381380067, 105406472747, 105757693523,
106287312683, 107104329947, 107722167083, 108509708243, 111931116827,
112096143563, 112942222403, 116198253107, 117139713683, 118669446587,
122381441027, 123139241867, 125860941443, 129686499347, 130505375027,
136323616547, 137236352987, 144078692123, 144112553027, 145220447003,
151674980867, 157057740083, 157705095203, 164322518387, 167591475923,
170139940907, 170997917507, 174232764683, 174376550483, 178261044323,
178337403467, 181474997843, 181716358427, 183564155603, 186482787083,
188006855987, 188097317747, 190673527883, 196003489043, 198077690627,
200013846563, 200358727187, 202842865763, 203487797267, 205130303363,
207226320347, 207871567763, 213264316523, 214329465563, 215837440307,
216733724507, 218386460627, 219577306643, 223241130083, 225712966883,
226775820347, 228109213307, 231052848443, 231474062483, 238428504467,
239095981547, 239522611787, 240211407827, 242333626523, 244729690523,

251815959107, 253197504587, 254168076683, 257548609163, 260470414907,
266538452003, 269243581667, 269775359963, 270808117667, 275154292643,
281179860947, 287697646763, 292783804067, 295268933987, 296570177147,
297894166883, 305241543947, 307734569603, 308412920843, 320152474787,
324579617723, 325875351563, 330736658867, 341815929563, 341918171027,
342692788283, 343006290083, 344024701283, 345995082947, 351275193323,
357890740547, 358507800947, 360537447563, 366790292147, 367688833643,
381202315283, 381945133187, 391367639627, 393653940347, 394145274563,
397347839723, 403313415467, 410050919123, 414667642787, 415162028987,
415831263923, 423850193843, 427818906107, 428025528347, 429180667163,
430413272363, 432495257963, 433198584563, 434651995307, 435473314187,
436289270867, 439819408907, 442258478507, 443905148147, 445826046203,
451303799363, 454593262283, 462158577323, 463348308443, 467800826387,
473371037387, 476089881203, 483806996483, 488054739563, 491585609147,
492758265107, 494306767667, 498605667347, 499042902203, 500152847507,
501310383107, 501363158603, 502966754867, 504293193683, 511475588747,
518266418723, 521971349987, 522253666547, 527375567843, 529504598747,
534911108483, 540551833667, 541527928163, 543055226243, 559642230083,
571511485187, 600446279267, 603379559243, 607077919403, 617589344147,
618008321867, 626280474587, 626983355987, 628858189403, 633659633963,
641675992283, 677732986523, 678192782987, 691568479907, 705282671627,
706344656723, 706699143827, 710446286723, 712059512243, 719162705363,
726328494107, 727763057867, 734455978787, 736180055267, 746327700323,

756281607227, 758133126683, 779371277387, 792797117867, 805420338083,
814343332547, 815655075227, 824029836947, 825924486323, 840532074227,
840895046507, 842652530747, 843116449163, 849941922227, 853008584147,
857545289243, 858056950427, 863713501067, 864430945163, 866981421347,
875925661907, 906061773587, 938295978443, 959291106707, 980239383323,
986850768203, 999219432803, 1001353386203, 1002556667507, 1042202411963,
1060704003347, 1078234050563, 1080200570627, 1084477543043, 1087092730667,
1087123594787, 1088567910707, 1103543831843, 1111355799347, 1152132122363,
1154116410467, 1161112785587, 1178169476147, 1195925340923, 1217104108163,
1221121768187, 1233464151947, 1237123165067, 1238853505427, 1257897178163,
1269013593203, 1270679649227, 1295696932067, 1296901027547, 1298142023603,
1311542907443, 1316052108083, 1329187409987, 1331485579883, 1338954143603,
1360378850123, 1382417765123, 1389833015003, 1401228277907, 1412527990283,
1413277594067, 1413485664947, 1440603609827, 1469213662307, 1484391565403,
1509250206683, 1528471334147, 1550750402387, 1592957434163, 1609013101427,
1677109112843, 1691954303003, 1694156671283, 1695927494243, 1707964770203,
1709696200547, 1715140069883, 1746802611563, 1762438832747, 1764367920803,
1765018711763, 1765733907563, 1771302610907, 1917119954483, 1927252748123,
1982411103203, 1983446892587, 2047822976603, 2049515406683, 2108768893307,
2118321564587, 2118981915467, 2126231729507, 2195134794467, 2247245178587,
2261960474603, 2293949351483, 2294964431003, 2336075481563, 2385895946003,
2386495572563, 2446571849483, 2537739463907, 2722133536163, 2793076822787,
2795345465867, 2801567860307, 2836431147923, 2852325262043, 2880757904363,

2960783389283, 2976447958307, 3039800203547, 3065513512907, 3077795184803,
3106017135443, 3141844350347, 3146545727723, 3303999496643, 3313829031203,
3335889715883, 3350310368387, 3355617971603, 3379314480947, 3473422815323,
3485282353307, 3505950530387, 3624902059403, 3661271038907, 3670748734883,
3726920463803, 3768534197507, 3791312625083, 3863872180403, 3886559107763,
3909871577483, 3921104116907, 3963192553043, 3996312084827, 4018601356667,
4036824307547, 4118666801003, 4128944343443, 4160373693347, 4175078861483,
4259608479587, 4429747456067, 4478476506083, 4501744655867, 4502311050803,
4514009790443, 4558065649427, 4579875283283, 4601840961563, 4829532901643,
4879793874563, 4911628283363, 4955557838027, 4970468701043, 5123318768027,
5137273601987, 5357577765347, 5409242934587, 5415839352587, 5440983231347,
5482729057787, 5558457793643, 5705723135867, 5781908833403, 5792185812947,
6022337325227, 6025227910547, 6121382021387, 6198112277867, 6333519653843,
6380006277347, 6394402726403, 6556061823803, 6562385103203, 6578916370787,
6593735393363, 6757247086523, 6782132725163, 6825098551787, 6857721811547,
6868371552443, 6935079048587, 7089508736363, 7162721962307, 7306345780667,
7440087686723, 7522490825747, 7531954720307, 7554323185667, 7582770737387,
7651182540707, 7745439047747, 7775377415627, 7842430725923, 7904670066347,
7941923157563, 7978670549387, 8192183978123, 8209830953147, 8259105813083,
8286631580243, 8481086497547, 8613717425147, 8938801960307, 8944420602563,
9076559140403, 9096576003227, 9219793002083, 9254810718467, 9292443749027,
9343441114307, 9361915665563, 9540337531427, 9674513510507, 9685255163147,
9688352206763, 9826720820003, 9959251126523, 10020563196467, 10047976765067,

10048608565883, 10091907996587, 10129909677947, 10142885853803,

10227534612803, 10295586301643, 10360065250883, 10364753829587,

10388318665427, 10400813054243, 10681706954027, 10742815880267,

10869505984763, 10900201957523, 10934495759123, 11211493371707,

11316247908443, 11329659315083, 11382923254643, 11441423775107,

11509475679347, 11570508049187, 11619092274947, 11706343315883,

11726422907363, 11764893056003, 12238205854907, 12249904453307,

12254837741747, 12463082133347, 12725326040507, 12949841371043,

13101796392107, 13103787884387, 13255898179787, 13571566517123,

13652879923547, 13743931311323, 13755001916963, 14144547460427,

14179029470027, 14427616013747, 14552069753603, 14827357763963,

14954888982347, 15161143183763, 15287891623883, 15311568641723,

15467314209587, 15495765291803, 15656573431547, 15760693624667,

15849308230523, 15854754991547, 15887762832563, 16004324745947,

16040331525203, 16084064843387, 16242898903643, 16405715580587,

17028415047563, 17290895168123, 17426441003267, 17588265716867,

18254656170827, 18314753553683, 18884008226123, 18962013223787,

19022254104083, 19082007788507, 19434054532283, 19689978232187,

19774780091483, 19937455896203, 20453821551827, 20671588148987,

21217226920907, 21462127223963, 21744381856427, 22472347939667,

22814410039307, 23077926086723, 23093636218883, 23138093384843,

24217748762507, 24966673160963, 25181979646427, 25269245084747,

25436779902827, 25513833729827, 25939374193907, 26950529333867,

27430893278963, 27433051197947, 27478158745883, 27995244090347,
27998600149547, 28111138628747, 28264851805643, 28558820279723,
28799022467483, 30030651188723, 30117259892123, 30321016458347,
31551033516803, 32369868797507, 32479468525643, 32687225459027,
33067009392227, 33425312567723, 33566165185307, 33851251809443,
34579652792123, 34751356594523, 35630843544323, 35918538773963,
36323149965227, 36600713556923, 36763326214163, 36765194669243,
36859970583587, 37988454215843, 38848563885203, 39328340621723,
39801554148827, 39876319505987, 40280716723547, 40418257493987,
40460293395227, 40463928189227, 40853630623283, 40980393954587,
41469439917707, 41634467037203, 41988809964947, 42710425439987,
43091033488907, 44199623457107, 44208600403043, 44392892480243,
45214211159963, 45605226380003, 45747135119843, 46153132695347,
46514026762667, 46554518348747, 46723263972587, 46739187917267,
47369169281363, 49075380599123, 49277047241267, 49519594020203,
49693346789147, 50657706647987, 51084356425787, 51868980758027,
53524274838347, 53725579892627, 54344643015347, 54585395516987,
55066584524363, 55220904946787, 56159214828947, 56528255403107,
57703608585083, 57986601434243, 58011282472547, 58055027232443,
58085406024323, 58595065221923, 59247642174563, 61003876810907,
62646030831563, 63484756628507, 63940625647667, 64847471815283,
64866633320747, 65372742123443, 67218452863307, 67393499486147,
69657154063307, 69816842799827, 71755383731363, 71903087679947,

76753513980107, 78195909340427, 79072665506603, 80889030232547,
80907674130827, 81961124494787, 82170666090827, 82477579727267,
82512682655003, 83918566007363, 84312963329363, 85017825914867,
85856369511443, 86871666384683, 88748852069723, 90761499519107,
90870201867563, 93033273964067, 93993849496907, 96834533603987,
98446668636707, 99551255004683, 100081876087043, 100309684136507,
100433656257227, 101213471059403, 101244754250243, 102205410482867,
102705330827843, 103732122165803, 104133850700483, 104176031151323,
104287827860243, 105253093034627, 105801589427603, 106035105636827,
106700314541867, 106874063173883, 107911507848227, 107956006972883,
115224896052683, 118948395977387, 120860723927123, 121815826783067,
122252895627923, 122387561455403, 123008411741627, 124392407571563,
126421753375763, 126936189944507, 127393284190787, 129267607064987,
129995179070963, 133188548898107, 134179888732427, 135616487040347,
137469089275643, 141013010179187, 141375024853523, 144230627902547,
144254468471123, 144531812139587, 154769388595667, 157125105378827,
164769782790707, 165754369026923, 166723317758843, 166883131585043,
170415343839443, 171633803051267, 178524279644507, 189395258903267,
191525461927043, 192287225086907, 193332330105227, 193468016586563,
195886884723947, 196575793998923, 201940235869643, 202298675949683,
209015721821507, 214278043870307, 215640850861523, 215883708797987,
219953270226227, 227053742773283, 229215549487427, 229648254616763,
231226614503483, 238427845745627, 243037883621483, 244323366527267,

244473627150827, 248429934021323, 248497900497923, 249208244104067,

249411536855387, 253367577523763, 253902200038787, 253919345073803,

259076542928867, 262858694831963, 264243123132587, 264497094927683,

265333595447267, 267791027437643, 267934269304547, 272829780264227,

278927270689643, 282841671209627, 285109823477987, 285951316285403,

290383597238723, 295221100358267, 300267023207843, 303637983271787,

309988160207483, 311201395777523, 314832107659403, 314991177652667,

318043273007627, 328813534485323, 334350409772987, 338038575831347,

338745870168803, 341683155136187, 346843236425963, 347245567890083,

359979202494347, 360679668865283, 364737641015723, 366357717044867,

369329285765963, 381150307254947, 381403229753867, 381962663529827,

384897844809323, 385559214556883, 386553120173723, 388453166085947,

392881118028923, 398289276206387, 398368406614187, 406477416390443,

408358865366483, 409937120891003, 411246391908323, 416471075861147,

434586909175043, 442615506258467, 444527917396187, 457884313171403,

464370024490283, 472156393842803, 482851512092123, 494039103065603,

494089269135443, 501911255266523, 505002688685267, 509321189046203,

522892208304563, 531145552587803, 533836952814707, 533973275772107,

553466720228267, 555219277716683, 558427819551083, 558711406317707,

561630157088027, 564657608583347, 572502757382027, 583110119102867,

588333795767387, 597184171441307, 597325750939763, 601132611478643,

603257010320747, 603421182689867, 605716136403947, 608721511540547,

617032338804347, 618248060417027, 618322175982587, 623959671285323,

645022608528923, 652112946998267, 660809694162803, 664656142888547,
678681420845963, 685177835011667, 691744278095027, 727028073472403,
735538049713067, 753599254597307, 760531495066643, 773127966513923,
780282675991307, 784454799049283, 796854246941003, 808348146672107,
833416832908427, 842557168108787, 843198010390763, 848609655132347,
871230814463267, 900772097527283, 919708815954827, 927282962390627,
933909122464163, 945112523777267, 954973489971203, 973616959049723,
976786130872043, 999183888454883, 1015170695639603, 1025185545528587,
1049142192182027, 1061514938720627, 1093664393343707, 1097209821479867,
1100247937039187, 1123865741920403, 1135644701881787, 1150713702023003,
1177808574356507, 1209634145420507, 1210070231072387, 1210188010338467,
1214874659305523, 1242657800952803, 1300393000972547, 1310423546669243,
1326166286896667, 1387038618621683, 1396592628314267, 1417068859020443,
1440342337943123, 1445364459801107, 1448391985610267, 1495713663430403,
1521207125511083, 1569693758717963, 1609972827806747, 1624444498937147,
1654042258227347, 1672824889599803, 1700010806378363, 1702668802430987,
1745771701384043, 1769411802386843, 1846664487998027, 1880418022028027,
1926424778152643, 1961097943845347, 1980478101623867, 1996018391680067,
1999911821728427, 2070488321013587, 2085054758121323, 2088262645010387,
2180915411922827, 2189422390531403, 2216307542171243, 2218516275145307,
2219414000274683, 2286764405381483, 2342969961676307, 2379732783470027,
2445934918896083, 2460691881833723, 2470932432917027, 2690997813743003,
2762254961702627, 2814589806558683, 2834363765334947, 2869535501850707,

2929172293317323, 2932577454014963, 2935854023789987, 2969213748336323,

3020818569551243, 3073673110706747, 3101371000200227, 3104849562515003,

3235238796281747, 3285815298164027, 3315161303303747, 3368947841039483,

3374982547293947, 3463360238101427, 3508731371470643, 3559625136261683,

3662373266422163, 3869311853707067, 3894122396803427, 4021855436339963,

4092334246001483, 4127045219115587, 4127816498745803, 4178487155887307,

4195777155959003, 4297612842652667, 4350902250536747, 4354702872331283,

4445385868576043, 4521311482843403, 4578809697819683, 4584399980277347,

4656482488678067, 4740220268428067, 4753230160843427, 4767239770155323,

4768318465377227, 4803981956525987, 4912275946718843, 4950233219804387,

5140463227973867, 5177154186574523, 5277936608002547, 5313508817995307,

5386560892100603, 5401510425664523, 5479062062615987, 5575361763719723,

5648459339875043, 5723737895914667, 6137401596884507, 6159802295032187,

6284420587607963, 6519831855735923, 6600316905303083, 6662683558988603,

6759712564297307, 6764580537823547, 6842873739207923, 6932302379792267,

7090323182194643, 7123896896797427, 7301837439873107, 7423032005898107,

7483685142532283, 7572010928000507, 7585210991707187, 7598318792034563,

7604488697514323, 7651923699830603, 7795299669308843, 7834545175010003,

7934360359558643, 8058082706175347, 8197408649073923, 8285341102435907,

8470328642519963, 8493684414768563, 8513468052684083, 8561200981449587,

8607491839865267, 8634933383942627, 8784707256866243, 8836569697022363,

8904054682585523, 8930920857479123, 9074525601051803, 9129829656292883,

9182724004227827, 9280412587535963, 9552203739121427, 10145286484466867,

```
10448013779960987, 10546884222902243, 10982573825805683, 11470847797382147,

11518275717582563, 11727660219060803, 11728190625795467, 11733906922970363,

12274184870022107, 12900124117888307, 13860804126519587, 14710317777427523,

14876404848807347, 15424852035191843, 15456233614284707, 15516182040130403,

15881251723875107, 15993559951849787, 16278199354365227, 16373247986319083,

16432225455935987, 16456981879192403, 17219972620249643, 17599336000468067,

17712183264506387, 17713242391363403, 17946824623023323, 18554485112684387,

18603905156978363, 18656807807083067, 19539078446270147, 19587941983265963,

20032994903640443, 20062739448254267, 20073762882803387, 20359633782746723,

20659400066441267, 20836233894557267, 21118565841067307, 21199872837863243,

21639863226002867, 21665921732653043, 21806240640021683, 21816982773446363,

21921768913642547, 22306424294890763, 22758386560600907, 23272156387715363,

23362136758549763, 23765686580141963, 23779912818978347, 24388673355443627,

25068477461488163, 25120668132244187, 28010077357379963, 28175136593318963,

28234760408801027, 28253421604483667, 28342894049307683, 28418793067182587,

28959000075093563, 28973635637593547, 28981865717066267, 29408837670188003,

30879024243546203, 31122911995061123, 34003034963387387, 34081016516391443,

34527755084118203, 34812554951408387, 35910014608004507, 36038400357251267,

36734225912784707, 40821325439590043, 43275561188284907, 45773909357813027,

45803392370932523, 46440190707594947, 46924172239098923, 48893432166202787,

49200142688511803, 49512050826222707, 49966879943561027, 50299228064987747,

50751627588104363, 50979565139595707, 51160874892711443, 51342043565115323,

52336515816096587, 53849090199502163, 56084293968300707, 56736105721363427,
```

57370939462458467, 59890538126188643, 62898373801967003, 63224830418726843,

63911087308130963, 64069860644436107, 65102211710974283, 65244956051352923,

68364114302945243, 68993625120758027, 69535625261957027, 70605691106667107,

70698154617852227, 74775688667115827, 77366514012572747, 77810517606645347,

81832868558259587, 84662170560167507, 85424126092904387, 88241901308943323,

88370577190598147, 89487944473229147, 90738049414775363, 92944479636891203,

93968413778246963, 94721650894464443, 98264729422537523, 100574867669404523,

105670092454526627, 106088472540298643, 107346675917023787,

109015944852075587, 110496757301649707, 119202335417867027,

120280161551631707, 123720497254227323, 123842703385128947,

139710885756410723, 142874667019602923, 144258585973273523,

148475293301464667, 149208771106892243, 150002901112328123,

151284810087833267, 155698225865750147, 156513316779821843,

158560157015337347, 159086994355643507, 161270234044769387,

182564686170812843, 185708623556971547, 186193009554484067,

186245783915355107, 187341391659000347, 193717105258811507,

209722418336488067, 214113212378603963, 217073524798507787,

219815153110441643, 221833965898638683, 222602612000153867,

228254762701474403, 234997175544575867, 242201050111964243,

243971856698516603, 245356346004186563, 249159259686463667,

258711794854487963, 267423977122819523, 271207705130884283,

288978492932019827, 291417082342436987, 302104887647400683,

313347078159591347, 315401630443322723, 321868979702422283,

325252632148305707, 332495942603426387, 342620644410625283,

353895843153126707, 374827336534364507, 374861765074159043,

378130477373807003, 392136195581446067, 453946097174059907,

468546146281174187, 475825128666231203, 488190214082668283,

513154194642154643, 523258024248646403, 523349974867018427,

542327418333492923, 547470795822983027, 547919001088955747,

555354140733500723, 572800553736798323, 573477691058513603,

580391299838820707, 580619268411606107, 587338419994431587,

608013369631893707, 643469090691644387, 648045186006733283,

690661450844434187, 694536474567709307, 695438444244168683,

709446148414434107, 715451685459108443, 757579277535813347,

771759984877375427, 818372340673892387, 834181931088858347,

862826575313833547, 881646446747895443, 904015626673291523,

917810275647435587, 926409902091049403, 957983612646444947,

970982876341566563, 988149678333335363, 1025460835415625563,

1135599215835240323, 1202488837681561667, 1202522950114508123,

1226454591140121587, 1283804986363619747, 1371063411574482467,

1374893187269319227, 1377484934835789947, 1395510653067507107,

1420738526009576003, 1440960146056135043, 1573938069761878523,

1658172557110780787, 1684876560586682867, 1725759802001944547,

1793205797680999547, 1798586889203203643, 1903210484910652763,

1930118566796892707, 1931107742142052427, 1943687942530023227,

1982956912291955627, 2045365290778092947, 2070575218629414467,

2128848802727382467, 2454907762770286403, 2881989581418653507,

2998963193285601803, 3057770866359333203, 3154852656683229347,

3243775175812344323, 3271649597763288587, 3293273341130047403,

3332003011782396563, 3379284211219592003, 3396033738339798107,

3668473736112124187, 4003282181723375963, 4104045934768959083,

4659174843785017163, 4756232973285560843, 4999420107398518163,

5384779401589724267, 5610619522633944803, 5630644485296016083,

5671974956911486523, 5732790094374343307, 5806713398544773483,

5968145502581738867, 6427261973867594243, 6622941681544599347,

6666972642646886987, 7006872751891625507, 7459916710490561123,

7712999766918759707, 8148134324527771643, 9518588067396692987,

9773976493068889187, 11377522503928924523, 11721529218605866307,

12418836529297484123, 14054980557432714203, 15257999331498847643,

15287120590231014227, 15808204622333584163, 16262625832371156083,

17176760531183413187, 18990570901948665683, 19424601827727592523,

19446136417030749947, 20021616827803179467, 21508580852131788923,

21712583176312725443, 22319868786904952603, 23811891485065963523,

24335625721109885027, 25667576698820885363, 28119136750979534027,

28778581223621774627, 31179575422425923723, 31805457664382582267,

32048800053660230147, 33301385469202933547, 33609583816541973683,

34958371361476721003, 36797010991101076883, 37837107119115295067,

39960712541695314107, 42594233069313285947, 53124488079102901787,

54436313269475606867, 57458390828623213547, 65318530118241733283,

```
65935732681157937443, 69203604098679896243, 71253805272249312467,

77615310528594222467, 87231729687142377803, 97494609888938325587,

118787204331030171347, 149783654928707474867, 169046290814577523283,

174739451001440889227, 176021198853828547883, 188825582091255593747,

197675610583408730387, 209822375767195448963, 218515727134757195723,

231239574705714414227, 250997709850412900363, 283417845636198816707,

436854357849340155563, 457160175195343015787, 519145721217285286283,

617373942999942504347, 812190152340933363227, 1131374128588340976947,

1204537852186137098027, 1494204424138930374947, 2091686873470882434323,

4098694483916743167203, 4665648301718926194203, 5360297573804957369123,

10659431812315035515387]
```

```
Ncases := 2030

n_viable := 0

max_in_viable_set := 0

my_maxdelta := 0

Global_N_values := 0
```

```
"
     --------------------------------
    "

                                P1 := 5003

                              Pmax := 10006

                           min_n_factors := 25

i= 1, P1= 5003, min_n_factors= 25 , nprimes in (p1, 2p1)= 3

"
     --------------------------------
    "

                                P1 := 5987

                              Pmax := 11974

                           min_n_factors := 26

i= 2, P1= 5987, min_n_factors= 26 , nprimes in (p1, 2p1)= 5
```

●
●
●

To prevent the document size from blowing up; we have skipped a whole bunch of analogous sets of printed lines, wherein each set is delimited from the next set by the dashed line within quotes;
and wherein;
one set of similar lines is printed out for each number in the original Grantham-list. A set of lines of this type indicates that that P1 value does not actually yield a viable set.

```
                   |
                   |
                   |
                   V
```

The first viable set info appears below.

```
"
     --------------------------------
    "

                                P1 := 4708691963

                               Pmax := 9417383926

                              min_n_factors := 65

i= 451, P1= 4708691963, min_n_factors= 65 , nprimes in (p1, 2p1)= 64

*** P1 = 4708691963, set is viable set number = 1

 myDelta = 0, cases to try-out-in-this-set = 1

 Viable_set_1_4708691963_64_0 :=
[4708691963, 4710995963, 4754130083, 4889392283, 4910078843, 4998584363,

    5003629523, 5042545667, 5125415987, 5151593267, 5240243243, 5333689787,

    5343444707, 5459704403, 5560240403, 5592303947, 5631606683, 5747783627,

    5814384083, 5823545723, 5881089107, 5885109323, 6052388267, 6053200667,

    6128580827, 6134338403, 6448080803, 6516763307, 6519138947, 6866836283,

    6889292027, 6930553163, 7016120627, 7092744083, 7276231883, 7422712883,

    7490008067, 7578493547, 7599012683, 7714374923, 7761591683, 7786043747,

    8020986203, 8063123843, 8063438627, 8229446123, 8297435963, 8300117867,

    8369939507, 8437088027, 8513676107, 8570641067, 8594608427, 8631134027,

    8803105163, 8927440883, 8967039563, 9069641027, 9115211723, 9115855187,

    9131776883, 9207476123, 9213008243, 9280410467, 9302901947]
 ;
```

The field “i = 451” in the line

```
i= 451, P1= 4708691963, min_n_factors= 65 , nprimes in (p1, 2p1)= 64
```

indicates that the first 450 values in the array (i.e. the Grantham-list) did not generate viable subsets.

We continue with successive lines from the printout in the verbatim mode below.

```
"
     --------------------------------
    "
                                P1 := 4710995963

                               Pmax := 9421991926

                              min_n_factors := 65

i= 452, P1= 4710995963, min_n_factors= 65 , nprimes in (p1, 2p1)= 63

"
     --------------------------------
    "
                                P1 := 4754130083

                               Pmax := 9508260166

                              min_n_factors := 65

i= 453, P1= 4754130083, min_n_factors= 65 , nprimes in (p1, 2p1)= 64

*** P1 = 4754130083, set is viable set number = 2

 myDelta = 0, cases to try-out-in-this-set = 1

 Viable_set_2_4754130083_64_0 :=
[4754130083, 4889392283, 4910078843, 4998584363, 5003629523, 5042545667,

    5125415987, 5151593267, 5240243243, 5333689787, 5343444707, 5459704403,

    5560240403, 5592303947, 5631606683, 5747783627, 5814384083, 5823545723,

    5881089107, 5885109323, 6052388267, 6053200667, 6128580827, 6134338403,

    6448080803, 6516763307, 6519138947, 6866836283, 6889292027, 6930553163,

    7016120627, 7092744083, 7276231883, 7422712883, 7490008067, 7578493547,
    7599012683, 7714374923, 7761591683, 7786043747, 8020986203, 8063123843,

    8063438627, 8229446123, 8297435963, 8300117867, 8369939507, 8437088027,

    8513676107, 8570641067, 8594608427, 8631134027, 8803105163, 8927440883,

    8967039563, 9069641027, 9115211723, 9115855187, 9131776883, 9207476123,

    9213008243, 9280410467, 9302901947, 9446845643, 9454452587]
 ;

                                    |
skipping a whole of bunch of lines  |  to directly jump to the 3rd viable set
                                    |
                                    V

"
     --------------------------------
    "

                                P1 := 4998584363

                               Pmax := 9997168726

                              min_n_factors := 65

i= 456, P1= 4998584363, min_n_factors= 65 , nprimes in (p1, 2p1)= 64

*** P1 = 4998584363, set is viable set number = 3

 myDelta = 0, cases to try-out-in-this-set = 1


 Viable_set_3_4998584363_64_0 :=
[4998584363, 5003629523, 5042545667, 5125415987, 5151593267, 5240243243,

    5333689787, 5343444707, 5459704403, 5560240403, 5592303947, 5631606683,

    5747783627, 5814384083, 5823545723, 5881089107, 5885109323, 6052388267,

    6053200667, 6128580827, 6134338403, 6448080803, 6516763307, 6519138947,

    6866836283, 6889292027, 6930553163, 7016120627, 7092744083, 7276231883,

    7422712883, 7490008067, 7578493547, 7599012683, 7714374923, 7761591683,
    7786043747, 8020986203, 8063123843, 8063438627, 8229446123, 8297435963,

    8300117867, 8369939507, 8437088027, 8513676107, 8570641067, 8594608427,

    8631134027, 8803105163, 8927440883, 8967039563, 9069641027, 9115211723,

    9115855187, 9131776883, 9207476123, 9213008243, 9280410467, 9302901947,

    9446845643, 9454452587, 9858736187, 9948847787, 9990625523]
 ;

                        |
 next we jump to the set | that yields maximum number of viable N values
                        |
                        V
"
     --------------------------------
    "

                              P1 := 427818906107

                             Pmax := 855637812214

                             min_n_factors := 78

i= 844, P1= 427818906107, min_n_factors= 78 , nprimes in (p1, 2p1)= 82

*** P1 = 427818906107, set is viable set number = 112

 myDelta = 5, cases to try-out-in-this-set = 29126360

 Viable_set_112_427818906107_82_5 :=
[427818906107, 428025528347, 429180667163, 430413272363, 432495257963,

    433198584563, 434651995307, 435473314187, 436289270867, 439819408907,

    442258478507, 443905148147, 445826046203, 451303799363, 454593262283,

    462158577323, 463348308443, 467800826387, 473371037387, 476089881203,

    483806996483, 488054739563, 491585609147, 492758265107, 494306767667,

    498605667347, 499042902203, 500152847507, 501310383107, 501363158603,

    502966754867, 504293193683, 511475588747, 518266418723, 521971349987,
    522253666547, 527375567843, 529504598747, 534911108483, 540551833667,

    541527928163, 543055226243, 559642230083, 571511485187, 600446279267,

    603379559243, 607077919403, 617589344147, 618008321867, 626280474587,

    626983355987, 628858189403, 633659633963, 641675992283, 677732986523,

    678192782987, 691568479907, 705282671627, 706344656723, 706699143827,

    710446286723, 712059512243, 719162705363, 726328494107, 727763057867,

    734455978787, 736180055267, 746327700323, 756281607227, 758133126683,

    779371277387, 792797117867, 805420338083, 814343332547, 815655075227,

    824029836947, 825924486323, 840532074227, 840895046507, 842652530747,

    843116449163, 849941922227, 853008584147]

 ;

"
     ----------------------------------
    "

                                P1 := 428025528347

                              Pmax := 856051056694

                              min_n_factors := 78

i= 845, P1= 428025528347, min_n_factors= 78 , nprimes in (p1, 2p1)= 81

*** P1 = 428025528347, set is viable set number = 113

 myDelta = 4, cases to try-out-in-this-set = 1752382


 Viable_set_113_428025528347_81_4 :=
[428025528347, 429180667163, 430413272363, 432495257963, 433198584563,

    434651995307, 435473314187, 436289270867, 439819408907, 442258478507,

    443905148147, 445826046203, 451303799363, 454593262283, 462158577323,
    463348308443, 467800826387, 473371037387, 476089881203, 483806996483,

    488054739563, 491585609147, 492758265107, 494306767667, 498605667347,

    499042902203, 500152847507, 501310383107, 501363158603, 502966754867,

    504293193683, 511475588747, 518266418723, 521971349987, 522253666547,

    527375567843, 529504598747, 534911108483, 540551833667, 541527928163,

    543055226243, 559642230083, 571511485187, 600446279267, 603379559243,

    607077919403, 617589344147, 618008321867, 626280474587, 626983355987,

    628858189403, 633659633963, 641675992283, 677732986523, 678192782987,

    691568479907, 705282671627, 706344656723, 706699143827, 710446286723,

    712059512243, 719162705363, 726328494107, 727763057867, 734455978787,

    736180055267, 746327700323, 756281607227, 758133126683, 779371277387,

    792797117867, 805420338083, 814343332547, 815655075227, 824029836947,

    825924486323, 840532074227, 840895046507, 842652530747, 843116449163,

    849941922227, 853008584147]

 ;

"
     --------------------------------
    "

                              P1 := 429180667163

                             Pmax := 858361334326

                             min_n_factors := 78

i= 846, P1= 429180667163, min_n_factors= 78 , nprimes in (p1, 2p1)= 82

*** P1 = 429180667163, set is viable set number = 114

 myDelta = 5, cases to try-out-in-this-set = 29126360
 Viable_set_114_429180667163_82_5 :=
[429180667163, 430413272363, 432495257963, 433198584563, 434651995307,

    435473314187, 436289270867, 439819408907, 442258478507, 443905148147,

    445826046203, 451303799363, 454593262283, 462158577323, 463348308443,

    467800826387, 473371037387, 476089881203, 483806996483, 488054739563,

    491585609147, 492758265107, 494306767667, 498605667347, 499042902203,

    500152847507, 501310383107, 501363158603, 502966754867, 504293193683,

    511475588747, 518266418723, 521971349987, 522253666547, 527375567843,

    529504598747, 534911108483, 540551833667, 541527928163, 543055226243,

    559642230083, 571511485187, 600446279267, 603379559243, 607077919403,

    617589344147, 618008321867, 626280474587, 626983355987, 628858189403,

    633659633963, 641675992283, 677732986523, 678192782987, 691568479907,

    705282671627, 706344656723, 706699143827, 710446286723, 712059512243,

    719162705363, 726328494107, 727763057867, 734455978787, 736180055267,

    746327700323, 756281607227, 758133126683, 779371277387, 792797117867,

    805420338083, 814343332547, 815655075227, 824029836947, 825924486323,

    840532074227, 840895046507, 842652530747, 843116449163, 849941922227,

    853008584147, 857545289243, 858056950427]

 ;

                         |
                         |
                         |
                         V
```

Finally we jump directly to the printout lines enumerating last 3 viable sets.

```
"
     -------------------------------
    "
                              P1 := 439819408907

                             Pmax := 879638817814

                             min_n_factors := 78

i= 853, P1= 439819408907, min_n_factors= 78 , nprimes in (p1, 2p1)= 79

*** P1 = 439819408907, set is viable set number = 121

 myDelta = 2, cases to try-out-in-this-set = 3161


 Viable_set_121_439819408907_79_2 :=
[439819408907, 442258478507, 443905148147, 445826046203, 451303799363,

    454593262283, 462158577323, 463348308443, 467800826387, 473371037387,

    476089881203, 483806996483, 488054739563, 491585609147, 492758265107,

    494306767667, 498605667347, 499042902203, 500152847507, 501310383107,

    501363158603, 502966754867, 504293193683, 511475588747, 518266418723,

    521971349987, 522253666547, 527375567843, 529504598747, 534911108483,

    540551833667, 541527928163, 543055226243, 559642230083, 571511485187,

    600446279267, 603379559243, 607077919403, 617589344147, 618008321867,

    626280474587, 626983355987, 628858189403, 633659633963, 641675992283,

    677732986523, 678192782987, 691568479907, 705282671627, 706344656723,

    706699143827, 710446286723, 712059512243, 719162705363, 726328494107,

    727763057867, 734455978787, 736180055267, 746327700323, 756281607227,

    758133126683, 779371277387, 792797117867, 805420338083, 814343332547,

    815655075227, 824029836947, 825924486323, 840532074227, 840895046507,
    842652530747, 843116449163, 849941922227, 853008584147, 857545289243,

    858056950427, 863713501067, 864430945163, 866981421347, 875925661907]

 ;

"
     --------------------------------
    "
                                P1 := 442258478507

                               Pmax := 884516957014

                               min_n_factors := 78

i= 854, P1= 442258478507, min_n_factors= 78 , nprimes in (p1, 2p1)= 78

*** P1 = 442258478507, set is viable set number = 122

 myDelta = 1, cases to try-out-in-this-set = 79


 Viable_set_122_442258478507_78_1 :=
[442258478507, 443905148147, 445826046203, 451303799363, 454593262283,

    462158577323, 463348308443, 467800826387, 473371037387, 476089881203,

    483806996483, 488054739563, 491585609147, 492758265107, 494306767667,

    498605667347, 499042902203, 500152847507, 501310383107, 501363158603,

    502966754867, 504293193683, 511475588747, 518266418723, 521971349987,

    522253666547, 527375567843, 529504598747, 534911108483, 540551833667,

    541527928163, 543055226243, 559642230083, 571511485187, 600446279267,

    603379559243, 607077919403, 617589344147, 618008321867, 626280474587,

    626983355987, 628858189403, 633659633963, 641675992283, 677732986523,

    678192782987, 691568479907, 705282671627, 706344656723, 706699143827,

    710446286723, 712059512243, 719162705363, 726328494107, 727763057867,
    734455978787, 736180055267, 746327700323, 756281607227, 758133126683,

    779371277387, 792797117867, 805420338083, 814343332547, 815655075227,

    824029836947, 825924486323, 840532074227, 840895046507, 842652530747,

    843116449163, 849941922227, 853008584147, 857545289243, 858056950427,

    863713501067, 864430945163, 866981421347, 875925661907]

 ;

"
     --------------------------------
    "

                              P1 := 443905148147

                             Pmax := 887810296294

                             min_n_factors := 78

i= 855, P1= 443905148147, min_n_factors= 78 , nprimes in (p1, 2p1)= 77

*** P1 = 443905148147, set is viable set number = 123

 myDelta = 0, cases to try-out-in-this-set = 1


 Viable_set_123_443905148147_77_0 :=
[443905148147, 445826046203, 451303799363, 454593262283, 462158577323,

    463348308443, 467800826387, 473371037387, 476089881203, 483806996483,

    488054739563, 491585609147, 492758265107, 494306767667, 498605667347,

    499042902203, 500152847507, 501310383107, 501363158603, 502966754867,

    504293193683, 511475588747, 518266418723, 521971349987, 522253666547,

    527375567843, 529504598747, 534911108483, 540551833667, 541527928163,

    543055226243, 559642230083, 571511485187, 600446279267, 603379559243,

    607077919403, 617589344147, 618008321867, 626280474587, 626983355987,
    628858189403, 633659633963, 641675992283, 677732986523, 678192782987,

    691568479907, 705282671627, 706344656723, 706699143827, 710446286723,

    712059512243, 719162705363, 726328494107, 727763057867, 734455978787,

    736180055267, 746327700323, 756281607227, 758133126683, 779371277387,

    792797117867, 805420338083, 814343332547, 815655075227, 824029836947,

    825924486323, 840532074227, 840895046507, 842652530747, 843116449163,

    849941922227, 853008584147, 857545289243, 858056950427, 863713501067,

    864430945163, 866981421347, 875925661907]

 ;

                       |
jump to print-sets for |  last two iterations of the outermost loop
                       |
                       V
"
     --------------------------------
    "

                          P1 := 466564830171892619420 3

                         Pmax := 933129660343785238840 6

                              min_n_factors := 144

i= 2028, P1= 4665648301718926194203, min_n_factors= 144 ,
         nprimes in (p1, 2p1)= 1

"
     --------------------------------
    "

                          P1 := 5360297573804957369123

                        Pmax := 10720595147609914738246

                              min_n_factors := 145

i= 2029, P1= 5360297573804957369123, min_n_factors= 145 ,
             nprimes in (p1, 2p1)= 1

"
     --------------------------------
    "

############ the final counts at the end of the first round ############

 *** n_viable sets = 123

 *** max_in_viable_set = 82

 *** Max myDelta = 5

  *** Total no of N values that must be tested numerically = 160681183


       "maple timestamp at exit || date = 2020-02-04 :: time = 15:26:09"
```

6.2 Appendix 2 : Second Round of Computations : numerically testing each survivor value of N

```
>>>>>>>>>>  source code  <<<<<<<<<<<<
 ------ file = analyze-viable-set.mpl -----------

analyze_viable_set := proc( set_id::posint , smallest_prime::posint ,
                            primes_bet_p_and_2p::posint , myDelta::nonnegint ,
                            vset::{list,Array,Vector} )

local p1 , n_in_range , Totprod , N , p , i, j, k, max_nfactors , i2 ,
      min_nfactors, nfactors, adelta, rest_of_vset, skipset, skipprod ,
      Total_ncases , caseno::int := 0 , nby2, my_index_list, bigset,
      N_COUNTER_EXAMPLES , N_carmichaels , N_psats , ii , Np1, Nm1,
      nsat, psat, j_is_in_skip_set, prevset, chksum, dummy_loop_limit,
      final_comb, niter, binomial_arg2 , tmp_nfacs
      ;

      if (type(vset,list) = true) then
        max_nfactors := nops(vset) ;

      elif (type(vset,Array) = true) then
        max_nfactors := upperbound(vset) ;

      elif (type(vset,Vector) = true) then
        max_nfactors := Vectdim(vset) ;

      else
         print(" XXXXX analyze_viable_set: vset is not of proper type exit");
         ERROR(" XXXXX analyze_viable_set: vset is not of proper type exit");
         quit ;

      fi ;

      if ( max_nfactors <> (primes_bet_p_and_2p + 1 ) ) then
         print(" XXXXX analyze_viable_set: dim(vset) mismatch exit") ;
         ERROR(" XXXXX analyze_viable_set: dim(vset) mismatch exit") ;
         quit ;
      fi ;

      p1 := smallest_prime ;
      min_nfactors := ceil( evalf[2*length(p1)](log[2](1 + p1^2)) ) ;

      adelta := primes_bet_p_and_2p - (min_nfactors - 1) ;
```

```
#      printf("dbg : p1= %d, min_nfactors= %d, n to 2p= %d \n",
#              p1, min_nfactors, primes_bet_p_and_2p) ;

#      printf("dbg : adelta=%d, myDelta= %d \n", adelta, myDelta) ;

      if adelta <> myDelta then
         print(" XXXXX analyze_viable_set: Delta mismatch exit") ;
         ERROR(" XXXXX analyze_viable_set: Delta mismatch exit") ;
         quit ;
      fi ;

      n_in_range := primes_bet_p_and_2p ;

      Totprod := 1 ;
      for i2 from 1 to max_nfactors do
         Totprod :=  Totprod * vset[i2] ;
      od ;

      unassign('rest_of_vset') ;
      rest_of_vset := [seq(vset[1+k], k = 1..n_in_range)] ;

      printf(" >>> analyze_viable_set no %d, P1 = %d \n\n", set_id, p1);
      print("analyze_viable_set: rest of vset = ", rest_of_vset);
      printf("Totprod including p1 = %d \n", Totprod) ;

      if n_in_range mod 2 = 0 then
        nby2 := n_in_range/2 ;
      else
         nby2 := (n_in_range - 1)/2 ;
      fi ;

      if adelta >  nby2 then
         adelta := n_in_range - adelta ;
      fi ;

      chksum := 0 ;

#      unassign('k') ;
#      my_index_list := [seq(k, k = 1..n_in_range)];
#
#      unaqssign('bigset') ;
#      bigset := choose(my_index_list, adelta) ;
#
#      print("dbg2 : bigset = ", bigset) ;
#      print("dbg3 : nops(bigset) = ", nops(bigset)) ;
```

```
      N_COUNTER_EXAMPLES := 0 ;
      N_carmichaels := 0 ;
      N_psats := 0 ;


#######
#      for ii from 1  to nops(bigset) do

     dummy_loop_limit := 2^(n_in_range) ;

######
     for binomial_arg2 from 0 to adelta do

       final_comb := lastcomb(n_in_range, binomial_arg2) ;
       unassign('skipset') ;
       niter := binomial(n_in_range, binomial_arg2) ;

       printf("\n\n  -_-_ dbg: binomial_arg2 = %d, ncases = %d _-_- \n\n",
                    binomial_arg2, niter) ;

       caseno := 0 ;

########
       for ii from 1 to dummy_loop_limit do

          caseno := caseno + 1 ;
          unassign('prevset') ;

#         skipset := bigset[ii] ;

          if ii = 1 then
             prevset := {} ;
             skipset := firstcomb(n_in_range, binomial_arg2) ;
          else
             if ii > niter then
                break ;
             else
                unassign('prevset') ;
                prevset := {seq(skipset[kk], kk = 1..nops(skipset))} ;
                unassign('skipset') ;
                skipset := nextcomb(prevset, n_in_range) ;
             fi ;
          fi ;

#         nfactors := 1 + n_in_range;
```

```
          nfactors := 1 + n_in_range - binomial_arg2;

          skipprod := 1 ;
          for j from 1 to nops(skipset) do
              skipprod := skipprod * rest_of_vset[skipset[j]] ;
          od ;

          # test the n_choose_detlta base case first

          printf(" \n ---- \n") ;
          print("dbg4 : ii, caseno, nfactors, skipset = ",
                 ii, caseno, nfactors,skipset) ;

#          printf("anal vset case no: %d \n",   caseno) ;
#          print("skipset = ", skipset) ;

          N := Totprod/skipprod ;

          printf(" skipprod = %d \n", skipprod) ;
          print(" MD5-hash of decimal string N = ",
                   Hash(sprintf("%d", N), method = md5) ) ;

          Np1 := N+1 ;
          Nm1 := N-1 ;
          nsat := 0 ;
          psat := 0 ;

          if (N mod p1) <> 0 then
             printf("p1 = %d, N = %d \n", p1, N) ;
             print("ERROR, p1 does not divide N, quit \n");
             ERROR("ERROR, p1 does not divide N, quit \n");
             return NULL ;
         fi ;

         if Nm1 mod (p1-1) = 0 then nsat := nsat + 1 ; fi ;
         if Np1 mod (p1+1) = 0 then psat := psat + 1 ; fi ;

         tmp_nfacs := 1 ;      # first factor is p1
         for j from 1 to n_in_range do
              j_is_in_skip_set := false ;
              for k from 1 to nops(skipset) do
                  if j = skipset[k] then
                     j_is_in_skip_set := true ;
                     break ;
                  fi ;
              od ;
              if j_is_in_skip_set = false then
                 p := rest_of_vset[j] ;
                 if (N mod p) <> 0 then
                         printf("p = %d, N = %d \n", p, N) ;
                         print("ERROR, p does not divide N, quit \n");
                         ERROR("ERROR, p does not divide N, quit \n");
                         return NULL ;
                 else
                    tmp_nfacs := tmp_nfacs + 1 ;
                 fi ;

                   if Nm1 mod (p-1) = 0 then nsat := nsat + 1 ;  fi ;
                   if Np1 mod (p+1) = 0 then psat := psat + 1 ;  fi ;

              fi ;

#             printf("deep dbg5 : j=%d , nsat=%d , psat=%d \n", j, nsat, psat);
#             print("deep dbg5 : j, j is in skipset, nsat, psat = \n",
#                    j, j_is_in_skip_set, nsat, psat);

          od ;

          if tmp_nfacs <> nfactors then
             print("n_factors mismatch, exit -1 \n") ;
             ERROR("n_factors mismatch, exit -1 \n") ;
             return NULL ;
          fi ;
          printf(" for this N, nsat = %d , psat = %d \n", nsat, psat) ;
          if nsat = nfactors then
                   N_carmichaels := 1 + N_carmichaels ;
                   printf(" ** found a CARMICHAEL = %d \n\n", N) ;
          fi ;

          if psat = nfactors then
                   N_psats := 1 + N_psats ;
                   printf(" ** found a Psat = %d \n\n", N) ;
          fi ;

          if nsat = nfactors and psat = nfactors then
             N_COUNTER_EXAMPLES := 1 + N_COUNTER_EXAMPLES ;
             printf(" *****_-_-_->>>> FOUND THE COUNTER EXAMPLE N = %d \n\n") ;
          fi ;

#          chksum := chksum + niter ;

          if skipset = final_comb then
```

```
             break ;
          fi ;


#######
      od ;  # end of the big outermost loop on ii stepping thru n_C_k combs

      if ii <> niter then
               print("ii, niter, skipset, final comb mismatch quit",
                ii, niter, skipset, final_comb) ;
               ERROR("ii, niter, skipset, final comb mismatch quit",
                ii, niter, skipset, final_comb) ;
               return NULL ;

      fi ;

      chksum := chksum + niter ;
#
    od ; # end of for on binomial_arg2 from 0 to adelta

    printf(" \n *** proc analyze_viable_set exit checksum = %d \n\n",
             chksum) ;

    return(N_COUNTER_EXAMPLES, N_carmichaels, N_psats, chksum) ;

end proc :



 ------>>>> source code file 2 = Bigrun-Grantham.mpl -----------



Grantham_BPSW_prob_PSPs_set :=
     [5003, 5987,  .... , 536029757380495736912З, 10659431812315035515387] ;

Ncases := nops(Grantham_BPSW_prob_PSPs_set) ;

n_viable := 0 ;
max_in_viable_set := 0  ;
my_maxdelta := 0 ;

Global_N_WILLIAMS := 0 ;
Global_N_CARMS := 0 ;
Global_N_PSATS := 0 ;
Global_N_cases := 0 ;
```

```
for i from 1 to (Ncases - 1) do
    print("\n -------------------------------- \n") :

    P1 := Grantham_BPSW_prob_PSPs_set[i] ;

    Pmax := 2 * P1 ;
    min_n_factors := ceil( evalf[2*length(P1)](log[2](1 + P1^2)) ) ;

    primes_between_p1_and_2p1 := 0 :

    for j from (i+1) while ( (j <= Ncases) and
                           (Grantham_BPSW_prob_PSPs_set[j] < Pmax) ) do
            primes_between_p1_and_2p1 := primes_between_p1_and_2p1 + 1 ;
    od :

    printf("i= %d, P1= %d, min_n_factors= %d , nprimes in (p1, 2p1)= %d\n\n",
           i, P1, min_n_factors, primes_between_p1_and_2p1) ;

    if ( primes_between_p1_and_2p1 > 10 and
       primes_between_p1_and_2p1 >= min_n_factors - 1) then

        n_viable := n_viable + 1 ;
             printf("*** P1 = %d, set is viable set number = %d \n\n",
                P1, n_viable) ;
        myDelta := primes_between_p1_and_2p1 - (min_n_factors - 1) ;
        printf(" Delta = %d, cases to try-out-in-this-set = %d \n\n",
                 myDelta, binomial(primes_between_p1_and_2p1, myDelta) );

          if primes_between_p1_and_2p1 > max_in_viable_set then
                max_in_viable_set := primes_between_p1_and_2p1 ;
          fi ;

        if myDelta > my_maxdelta then
           my_maxdelta := myDelta ;
        fi ;


        unassign(’tset’) ;
        tset := [P1] ;
        for k from 1 to primes_between_p1_and_2p1 do
          tset := [op(tset), Grantham_BPSW_prob_PSPs_set[i+k] ];
        od :
        printf(" \n Viable_set_%d_%d_%d_%d :=  \n",
                    n_viable, P1, primes_between_p1_and_2p1, myDelta) ;
        print(tset) ;
        printf(" ;\n\n") ;
        n_williams, n_carms, n_psats, n_cases :=
          analyze_viable_set(n_viable, P1, primes_between_p1_and_2p1,
                             myDelta, tset);

        printf("--->>>> for this set williams = %d, carms = %d, psats = %d \n",
                n_williams, n_carms, n_psats) ;

        Global_N_WILLIAMS := Global_N_WILLIAMS + n_williams ;
        Global_N_CARMS := Global_N_CARMS + n_carms ;
        Global_N_PSATS := Global_N_PSATS + n_psats ;
        Global_N_cases := Global_N_cases + n_cases ;

  fi ;


od ;

printf(" *** n_viable sets = %d \n\n", n_viable);
printf(" *** max_in_viable_set = %d \n\n", max_in_viable_set) ;
printf(" *** Max Delta = %d \n\n ", my_maxdelta) ;

printf(" *** Total number of survivor N values tested = %d \n\n",
         Global_N_cases) ;

printf(" ***** wILLIAM’S NUMBERS FOUND = %d \n\n", Global_N_WILLIAMS);
printf(" ***** CARMS = %d \n\n", Global_N_CARMS) ;
printf(" ***** PSATS = %d \n\n ", Global_N_PSATS) ;

  >>>>>>>>>> end of source code  <<<<<<<<<<<<
```

```
set MYMAPLE = /home/phatak/maple2017/bin/maple

and now the execution snapshot
start time = Mon Dec  2 17:46:03 EST 2019

the nohup maple run status = 0

Grantham_BPSW_prob_PSPs_set := [5003, 5987, ... , 9280410467, 9302901947] ;


                                 Ncases := 2030

                                 n_viable := 0

                             max_in_viable_set := 0
```

```
                                my_maxdelta := 0

                             Global_N_WILLIAMS := 0

                              Global_N_CARMS := 0

                              Global_N_PSATS := 0

                              Global_N_cases := 0

"
     --------------------------------
    "

                                   P1 := 5003

                                 Pmax := 10006

                              min_n_factors := 25

i= 1, P1= 5003, min_n_factors= 25 , nprimes in (p1, 2p1)= 3

"
     --------------------------------
    "

                                   P1 := 5987

                                 Pmax := 11974

                              min_n_factors := 26

i= 2, P1= 5987, min_n_factors= 26 , nprimes in (p1, 2p1)= 5

"
     --------------------------------
    "
```

```
                                      |
                                      |
skipping a whole of bunch of lines    |  directly jump to the 1st viable set
it has myDelta = 0.                   |
Therefore only 1 value of N including | all elements in the viable set survives
                                      |
                                      V

"
     --------------------------------
    "

                                P1 := 4708691963

                               Pmax := 9417383926

                              min_n_factors := 65

i= 451, P1= 4708691963, min_n_factors= 65 , nprimes in (p1, 2p1)= 64

*** P1 = 4708691963, set is viable set number = 1

 Delta = 0, cases to try-out-in-this-set = 1


 Viable_set_1_4708691963_64_0 :=
[4708691963, 4710995963, 4754130083, 4889392283, 4910078843, 4998584363,

    5003629523, 5042545667, 5125415987, 5151593267, 5240243243, 5333689787,

    5343444707, 5459704403, 5560240403, 5592303947, 5631606683, 5747783627,

    5814384083, 5823545723, 5881089107, 5885109323, 6052388267, 6053200667,

    6128580827, 6134338403, 6448080803, 6516763307, 6519138947, 6866836283,

    6889292027, 6930553163, 7016120627, 7092744083, 7276231883, 7422712883,

    7490008067, 7578493547, 7599012683, 7714374923, 7761591683, 7786043747,

    8020986203, 8063123843, 8063438627, 8229446123, 8297435963, 8300117867,

    8369939507, 8437088027, 8513676107, 8570641067, 8594608427, 8631134027,

    8803105163, 8927440883, 8967039563, 9069641027, 9115211723, 9115855187,
    9131776883, 9207476123, 9213008243, 9280410467, 9302901947]
 ;

 >>> analyze_viable_set no 1, P1 = 4708691963

"analyze_viable_set: rest of vset = ", [4710995963, 4754130083, 4889392283,

    4910078843, 4998584363, 5003629523, 5042545667, 5125415987, 5151593267,

    5240243243, 5333689787, 5343444707, 5459704403, 5560240403, 5592303947,

    5631606683, 5747783627, 5814384083, 5823545723, 5881089107, 5885109323,

    6052388267, 6053200667, 6128580827, 6134338403, 6448080803, 6516763307,

    6519138947, 6866836283, 6889292027, 6930553163, 7016120627, 7092744083,

    7276231883, 7422712883, 7490008067, 7578493547, 7599012683, 7714374923,

    7761591683, 7786043747, 8020986203, 8063123843, 8063438627, 8229446123,

    8297435963, 8300117867, 8369939507, 8437088027, 8513676107, 8570641067,

    8594608427, 8631134027, 8803105163, 8927440883, 8967039563, 9069641027,

    9115211723, 9115855187, 9131776883, 9207476123, 9213008243, 9280410467,

    9302901947]

Totprod including p1 = 21755625975536841632906952602019669266226596078377792487-
73376430737669452398254486491452295411306903906531361614147066665251632360378-
40743027814394395441469043486014265219620201108682926825571670398670698974388-
27524017092130125550536563625568202608325097738127070170225019542719860331230-
69142795117612897360559429874424712852577829356806869624368840054088517232354-
63918872608334052731208906900152718706529670691465755023830390352895624395263-
23298374806649074037614583798935327066209818121400147072596836036804391100426-
21940480969182161575294530738591837369529136936377855993003826100260422592527-
119694299415413186816421095642740112123


  -_-_ dbg: binomial_arg2 = 0, ncases = 1 _-_-


 ----
           "dbg4 : ii, caseno, nfactors, skipset = ", 1, 1, 65, {}
 skipprod = 1
     " MD5-hash of decimal string N = ", "3daa747bce5f99c2a7c7b42f518fb792"

 for this N, nsat = 0 , psat = 0

 *** proc analyze_viable_set exit checksum = 1

--->>>> for this set williams = 0, carms = 0, psats = 0

"
     --------------------------------
    "

                                P1 := 4710995963

                               Pmax := 9421991926

                              min_n_factors := 65

 i= 452, P1= 4710995963, min_n_factors= 65 , nprimes in (p1, 2p1)= 63

"
     --------------------------------
    "



                                       |
                                       |
skipping a whole of bunch of lines     |  directly jump to the 7-th viable set
it has myDelta = 1.                    |
Therefore only 66 value of N  survive  |
one value includes all 66 primes in    |  viable set 7
The others are obtained by skipping    |  one of the 65 in (P1, 2*P1) at a time
                                       |
                                       V


"
     --------------------------------
    "

                                P1 := 5459704403

                              Pmax := 10919408806

                              min_n_factors := 65
 i= 464, P1= 5459704403, min_n_factors= 65 , nprimes in (p1, 2p1)= 65

*** P1 = 5459704403, set is viable set number = 7

 Delta = 1, cases to try-out-in-this-set = 66

 Viable_set_7_5459704403_65_1 :=
[5459704403, 5560240403, 5592303947, 5631606683, 5747783627, 5814384083,

    5823545723, 5881089107, 5885109323, 6052388267, 6053200667, 6128580827,

    6134338403, 6448080803, 6516763307, 6519138947, 6866836283, 6889292027,

    6930553163, 7016120627, 7092744083, 7276231883, 7422712883, 7490008067,

    7578493547, 7599012683, 7714374923, 7761591683, 7786043747, 8020986203,

    8063123843, 8063438627, 8229446123, 8297435963, 8300117867, 8369939507,

    8437088027, 8513676107, 8570641067, 8594608427, 8631134027, 8803105163,

    8927440883, 8967039563, 9069641027, 9115211723, 9115855187, 9131776883,

    9207476123, 9213008243, 9280410467, 9302901947, 9446845643, 9454452587,

    9858736187, 9948847787, 9990625523, 9999012323, 10135853243, 10362267467,

    10393347563, 10451207267, 10653747803, 10689483347, 10691597003,

    10794517667]

 ;

 >>> analyze_viable_set no 7, P1 = 5459704403

"analyze_viable_set: rest of vset = ", [5560240403, 5592303947, 5631606683,

    5747783627, 5814384083, 5823545723, 5881089107, 5885109323, 6052388267,

    6053200667, 6128580827, 6134338403, 6448080803, 6516763307, 6519138947,

    6866836283, 6889292027, 6930553163, 7016120627, 7092744083, 7276231883,

    7422712883, 7490008067, 7578493547, 7599012683, 7714374923, 7761591683,

    7786043747, 8020986203, 8063123843, 8063438627, 8229446123, 8297435963,
    8300117867, 8369939507, 8437088027, 8513676107, 8570641067, 8594608427,

    8631134027, 8803105163, 8927440883, 8967039563, 9069641027, 9115211723,

    9115855187, 9131776883, 9207476123, 9213008243, 9280410467, 9302901947,

    9446845643, 9454452587, 9858736187, 9948847787, 9990625523, 9999012323,

    10135853243, 10362267467, 10393347563, 10451207267, 10653747803,

    10689483347, 10691597003, 10794517667]

Totprod including p1 = 2267252761446411903518519694376559052088007254121080772-
148202538739508996188703332446411474037434653472029224864522408552936850388730-
243886735865777511091826765354521532182911106864746958306336791745112527048957-
649516956749699748652836456935578204264028236573009419018588902922920101809359-
990463570402342494615748735685629488029878697885029439277876494331775105116484-
811152209231550587320394769353631896980853899577981512392687910552407787573970-
649243836845819940332121487961776480699094136876429467592562342564712683688647-
199663424850597056107162129554042959338786096195650317941483566949472124022115-
95902324379057701402079663555838830126745720450483889


  -_-_ dbg: binomial_arg2 = 0, ncases = 1 _-_-


 ----
           "dbg4 : ii, caseno, nfactors, skipset = ", 1, 1, 66, {}

 skipprod = 1
    " MD5-hash of decimal string N = ", "383863221f2d8b7b469213e3acd85f5b"

 for this N, nsat = 0 , psat = 0


  -_-_ dbg: binomial_arg2 = 1, ncases = 65 _-_-




 ----
           "dbg4 : ii, caseno, nfactors, skipset = ", 1, 1, 65, {1}

 skipprod = 5560240403
    " MD5-hash of decimal string N = ", "c0ec64ef02a5b3fb04642c22c47d80cd"
 for this N, nsat = 0 , psat = 0

 ----
            "dbg4 : ii, caseno, nfactors, skipset = ", 2, 2, 65, {2}

 skipprod = 5592303947
     " MD5-hash of decimal string N = ", "ab497f37687a7679dd2887a899744c6d"

 for this N, nsat = 0 , psat = 0

 ----
            "dbg4 : ii, caseno, nfactors, skipset = ", 3, 3, 65, {3}

 skipprod = 5631606683
     " MD5-hash of decimal string N = ", "a4fc0a74e7b3adde206538e1d73b9e3e"

 for this N, nsat = 0 , psat = 0

 ----


                                       |
                                       |
skipping a whole of bunch of lines     |  directly jump to the last two
values of N  that survive from         |  viable set 7
                                       |
                                       |
                                      V

----

          "dbg4 : ii, caseno, nfactors, skipset = ", 64, 64, 65, {64}

 skipprod = 10691597003
     " MD5-hash of decimal string N = ", "e6225f2e64c463ceb54edece2beee1c2"

 for this N, nsat = 0 , psat = 0



 ----
          "dbg4 : ii, caseno, nfactors, skipset = ", 65, 65, 65, {65}

 skipprod = 10794517667
     " MD5-hash of decimal string N = ", "b6b49e04918f4c9980c341a2d336b9af"
```

```
  for this N, nsat = 0 , psat = 0

  *** proc analyze_viable_set exit checksum = 66

--->>>> for this set williams = 0, carms = 0, psats = 0


                                       |
                                       |
 Finally, directly jump to the summary | printout lines at the very end
                                       |
                                       |
                                       V



  *** n_viable sets = 123

  *** max_in_viable_set = 82

  *** Max Delta = 5

   *** Total number of survivor N values tested = 160681183

  ***** wILLIAM'S NUMBERS FOUND = 0

  ***** CARMS = 0

  ***** PSATS = 0

        "maple timestamp at exit || date = 2019-12-03 :: time = 22:01:26"
```