Title: Unknown Pattern of prime numbers.
Author: Zeolla, Gabriel Martin
Comments: 11 pages
gabrielzvirgo@hotmail.com

Keywords: Pattern, prime numbers, composite number.

Abstract:
This text develops and formulates the discovery of an unknown pattern for prime numbers, with amazing and calculable characteristics. Using a mechanism similar to the Collatz conjecture.

Prime numbers Pattern

A clear rule of thumb states exactly what makes a prime number: it is an integer that cannot be divided exactly by any other number except 1 and itself. But there is no discernible pattern in the appearance of the prime numbers. Beyond the obvious - after numbers 2 and 5, prime numbers cannot be even or end in 5. There seems to be little structure that can help predict where the next prime number will appear.

Discovering the pattern of prime numbers

Let the following operation be applicable to any odd natural number greater than 1.
Let (m): the number to be tested.
We apply

Development has two variables

Formula A	**Formula B**
$\mathrm{k} > 1 \in \mathbb{N}$ $m = 2k + 1 \Leftrightarrow k \equiv 1 \vee 2 \ (Mod\ 4)$ $\longrightarrow n = -1$ $n = initial\ succession\ number$	$\mathrm{k} > 1 \in \mathbb{N}$ $m = 2k + 1 \Leftrightarrow k \equiv 0 \vee 3 \ (Mod\ 4)$ $\longrightarrow n = 1$ $n = initial\ succession\ number$

- If (n) is even, divide by 2.
- If (n) is odd, add (m) and divide by 2.

Formally, this corresponds to a function $f\colon \mathbb{N} \mapsto \mathbb{N}$

$$f(m) = \begin{cases} \dfrac{n}{2}, & if\ n\ is\ even \\ \dfrac{n+m}{2}, & if\ n\ is\ odd \end{cases}$$

Given any number, we can consider its cycle, that is, the successive i images when iterating the function.

We can calculate the number of images that the cycle forms f_x as follows:

For example: $f(m) = 13$:

$$\boldsymbol{f_x = \left(\frac{f(m)-1}{2}\right) - 1}$$

$$f_x = \left(\frac{13-1}{2}\right) - 1 = 5$$

$$\therefore 0 \le f_x \le 5$$

<u>Formula A</u>: Calculation of the starting number.

$13 = 2 * 6 + 1 \Leftrightarrow \ \ 6 \equiv 1 \vee 2\ (Mod\ 4) \longrightarrow n = -\mathbf{1}$ (nº initial)

$$\boldsymbol{f_x(m) = i}$$

$$f_5(13) = \frac{-1+13}{2} = \mathbf{6}$$

$$f_4(f_5(13)) = \frac{6}{2} = \mathbf{3}$$

$$f_3\left(f_4(f_5(13))\right) = \frac{3+13}{2} = \mathbf{8}$$

$$f_2\left(f_3\left(f_4(f_5(13))\right)\right) = \frac{8}{2} = \mathbf{4}$$

$$f_1\left(f_2\left(f_3\left(f_4(f_5(13))\right)\right)\right) = \frac{4}{2} = \mathbf{2}$$

$$f_0\left(f_1\left(f_2\left(f_3\left(f_4(f_5(13))\right)\right)\right)\right) = \frac{2}{2} = \mathbf{1}$$

<u>Proposition:</u>

- <u>**New conjeture of prime numbers:**</u> Once the function is executed $f(m)$ all odd prime numbers end the sequence in $f_0 = 1$

$P = \{3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,79,83,89, \ldots\ldots\}$

- Base 2 pseudoprimes also end in 1.
- Therefore, odd composite numbers that are not base 2 pseudoprimes end with $f_0 \neq 1$

Pseudoprime numbers (Euler-Jacobi).

$$Psp = \{561, 1.105, 1.729, 1.905, 2.047, 2.465, 3.277, 4.033, 4.681, 6.601, 8.321, 8.481, 10.585, 12.801, 15.841, 16.705, 18.705, 25.761, 29.341, 30.121, 33.153, 34.945, 41041, 42.799, \dots.\}$$

Reference OEIS A047713

These represent a very small portion of the set of composite numbers.
His images form patterns within the cycle.

The succession of images forms a cycle

- Each odd number (m) has a unique and unrepeatable cycle, with images (i) such that, $0 < (i) < m$.
- The cycle will be formed by the total number of images, cycle $= f_x + 1$

Cycle characteristics

A. A. There are cycles with repeated images, forming patterns.
B. B. There are cycles with images without repeating.

A) Prime numbers with patterns in their cycles

They are those prime numbers whose images are repeated forming patterns.

$$P_A = \{31,43,73,89,109,113,127,151,157,223,229,233,241,251,257,277,281,283,307, \dots\}$$

Reference OEIS A082595

Example:

$f(m) = 31$

f_x	i
f_{14}	16
f_{13}	8
f_{12}	4
f_{11}	2
f_{10}	1
f_9	16
f_8	8
f_7	4
f_6	2
f_5	1
f_4	16
f_3	8
f_2	4
f_1	2
$\boldsymbol{f_0}$	**1**

$$\boldsymbol{f_x = \left(\frac{f(m)-1}{2}\right) - 1}$$

$$f_x = \left(\frac{31-1}{2}\right) - 1 = 14$$

$$\therefore 0 \le f_x \le 14$$

Formula B:

$31 = 2 * 15 + 1 \Leftrightarrow$
$15 \equiv 0 \vee 3 \ (Mod\ 4)$
$\longrightarrow n = 1$ (initial number)

$$f_{14} = \frac{1+31}{2} = \mathbf{16}$$

$$cycle = f_x + 1$$
$$cycle = f_{14} + 1 = \mathbf{15}$$

<u>The patterns of each cycle are linked to the dividers of the cycle.</u>
Example above with loop 15, you have a pattern of 1 * 15, 15 * 1, 5 * 3, or 3 * 5.
In this case you have 3 patterns of 5 images each.

B) Prime numbers with cycles without repetition

They are those prime numbers whose images are not repeated.

$$P_B = \{3,5,7,11,13,17,19,23,29,37,41,47,53,59,61,67,71,79,83,97,101,103,107, \ldots\ldots\}$$

Examples of prime numbers with cycles without repeating numbers.

$f(m) = 17$

f_x	i
f_7	9
f_6	13
f_5	15
f_4	16
f_3	8
f_2	4
f_1	2
f_0	1

Formula B

$f(m) = 19$

f_x	i
f_8	9
f_7	14
f_6	7
f_5	13
f_4	16
f_3	8
f_2	4
f_1	2
f_0	1

Formula A

$f(m) = 23$

f_x	i
f_{10}	12
f_9	6
f_8	3
f_7	13
f_6	18
f_5	9
f_4	16
f_3	8
f_2	4
f_1	2
f_0	1

Formula B

$f(m) = 29$

f_x	i
f_{13}	14
f_{12}	7
f_{11}	18
f_{10}	9
f_9	19
f_8	24
f_7	12
f_6	6
f_5	3
f_4	16
f_3	8
f_2	4
f_1	2
f_0	1

Formula A

$f(m) = 37$

f_x	i
f_{17}	18
f_{16}	9
f_{15}	23
f_{14}	30
f_{13}	15
f_{12}	26
f_{11}	13
f_{10}	25
f_9	31
f_8	34
f_7	17
f_6	27
f_5	32
f_4	16
f_3	8
f_2	4
f_1	2
f_0	1

Formula A

Odd Composite Numbers

The characteristic of odd composite numbers is that $\boldsymbol{f_0} \neq \mathbf{1}$, this happens for all odd composite numbers that are not base 2 pseudoprimes.

$C = \{9,15,21,25,27,33,35,39,45,49,51,55,57,63,65,69,75,77,81,85,91,93,95, \ldots\ldots\}$

Examples of odd composite numbers

$f(m) = 15$

f_x	i
f_6	8
f_5	4
f_4	2
f_3	1
f_2	8
f_1	4
$\boldsymbol{f_0}$	2

Formula B

$f(m) = 25$

f_x	i
f_{11}	13
f_{10}	19
f_9	22
f_8	11
f_7	18
f_6	9
f_5	17
f_4	21
f_3	23
f_2	24
f_1	12
$\boldsymbol{f_0}$	6

Formula B

$f(m) = 27$

f_x	i
f_{12}	13
f_{11}	20
f_{10}	10
f_9	5
f_8	16
f_7	8
f_6	4
f_5	2
f_4	1
f_3	14
f_2	7
f_1	17
$\boldsymbol{f_0}$	22

Formula A

Demonstration

Each image that forms the cycle has an order in the power of 2, so if we apply modular arithmetic to these we can obtain equivalent congruences for all images.

Example of what happens with prime numbers:

$f(m) = 19$; $2^{f_x} \equiv i \left(Mod\ f(m)\right)$

f_x	i	2^{f_x}	$\equiv i \left(Mod\ f(m)\right)$
f_8	9	2^8	≡9 (Mod 19)
f_7	14	2^7	≡14 (Mod 19)
f_6	7	2^6	≡7 (Mod 19)
f_5	13	2^5	≡13 (Mod 19)
f_4	16	2^4	≡16 (Mod 19)
f_3	8	2^3	≡8 (Mod 19)
f_2	4	2^2	≡4 (Mod 19)
f_1	2	2^1	≡2 (Mod 19)
$\boldsymbol{f_0}$	**1**	2^0	≡**1** (Mod 19)

"Ultimately each image is transformed into a congruent residue"

Example of what happens with odd composite numbers:

$f(m) = 25$; $2^{f_x} \not\equiv i\left(Mod\ f(m)\right)$

f_x	i	2^{f_x}	$\not\equiv i\left(Mod\ f(m)\right)$
f_{11}	13	2^{11}	≢13 (Mod 25)
f_{10}	19	2^{10}	≢19 (Mod 25)
f_9	22	2^9	≢22 (Mod 25)
f_8	11	2^8	≢11 (Mod 25)
f_7	18	2^7	≢18 (Mod 25)
f_6	9	2^6	≢9 (Mod 25)
f_5	17	2^5	≢17 (Mod 25)
f_4	21	2^4	≢21 (Mod 25)
f_3	23	2^3	≢23 (Mod 25)
f_2	24	2^2	≢24 (Mod 25)
f_1	12	2^1	≢12 (Mod 25)
$\boldsymbol{f_0}$	6	2^0	≢6 (Mod 25)

"Then each image becomes a non-congruent residue"

Pseudoprime numbers have the same characteristic as prime numbers.

Alfa program with Python 3.9

This program builds the sequence of images of the cycle and ends in $f0$, the program. Analyze if $f0 = 1$ if it is true it will be a prime number otherwise it is composite. Recall that base 2 pseudoprimes are hidden among prime numbers and also pass the test.

```
# Argentest Alfa
# Pyton 3.9..Unknown pattern of prime numbers, Author Zeolla Gabriel Martín
# When the loop ends in 1,(number) is prime or with very low pseudo-
prime probabilities (Base 2 euler)

number = int(input("Enter Odd number = "))
numb = (number -1) / 2
if number % 2==0 or number<0:
        print("ERROR, THE NUMBER IS INCORRECT")
if numb % 4 == 1:
   z = (number - 1)// 2
   print("characteristic 1(Mod 4)")
elif numb % 4 == 2:
   z = (number - 1)// 2
   print(" characteristic 2(Mod 4)")
elif numb % 4 == 0:
   z = (number + 1)// 2
   print("characteristic 0(Mod 4)")
else:
   z = (number + 1)// 2
   print("characteristic 3(Mod 4)")
print(z)

counter=2
if z > 0:
     while counter != (((number-1)//2)+1):
        if z % 2:
            z = (z + number) // 2
        else:
            z //= 2
        print(z)
        counter=(counter +1)
        if (counter+1) > (((number-1)//2)+1):
                   break

print ("The ", number, "has a cycle of:",(number-1) //2 )
print("Is a possible prime number? ",z==1 )
```

Beta program with Python 3.9

Pseudoprimos removed

The most effective way to detect a pseudo prime is knowing that they form patterns within their cycle. But there are prime numbers like Mersenne's and others that also form patterns. Therefore, this program does not confirm the primality of these numbers, but it does manage to certify the primality of 70% of the set of prime numbers and 99% of the set of composite numbers.

This program builds the sequence of images of the cycle up to $i = 1$ if possible and stops, if there is no image with that value, it completes the cycle until $f0$.

The program parses and returns as a result:
$f0 = 1 \rightarrow Prime\ Number\ confirmed$, since it has no patterns within the loop.
$f0 \neq 1 \rightarrow Composite\ number\ confirmed$
$fx < Cycle \wedge cycle \equiv 0\ (Mod\ fx) \rightarrow Possible\ prime\ number$!!
Since it has Patterns in the cycle. So there is some possibility that it is pseudo-prime.

Reference

For more information on how to test primality without falling into the pseudo-prime trap, I expanded the research and did another work called:
New Argentest primality test algorithm
Download or read it online
https://www.academia.edu/51147288/Nuevo_algoritmo_de_prueba_de_primalidad_Argentest

```python
# Argentest Beta.  Professor Zeolla Gabriel Martín
# Primality test using functions that generate patterns.
#  (n) is PRIME CONFIRMATION or COMPOSITE NUMBER CONFIRMATION
# (n) WITHOUT CONFIRMATION and True  "it is prime or pseudoprime (Base 2 euler)

number = int(input("Enter Odd number = "))
numb = (number -1) / 2

if numb % 4 == 1:
   z = (number - 1)// 2
elif numb % 4 == 2:
   z = (number - 1)// 2
elif numb % 4 == 0:
   z = (number + 1)// 2
else:
   z = (number + 1)// 2
print(z)

counter=2
if z > 0:
     while z != 1:
        if z % 2:
            z = (z + number) // 2
        else:
            z //= 2
        print(z)
        counter=(counter +1)
        if (counter+1) > (((number-1)//2)+1):
                   break
else:
    print("The number entered is incorrect")

if ((number-1) //2)== (counter-1):
   print("Complete cycle ")
else:
      print("Incomplete cycle, has",((number-1) //2) /(counter-1 ),"patterns")

print ("The ", number, "has a cycle of:",(number-1) //2 )
r=numb % (counter-1)
print("Possible prime number!! " ,r==0 and z==1)
if numb / (counter-1) ==1 and z==1:
   print("PRIME NUMBER CONFIRMATION")
if (r==0 and z==1)== False:
   print("COMPOSTE NUMBER CONFIRMATION")
```

Fast Beta program with Python 3.9

Solve the same as in the previous one without spewing the sequence of images

```
# Argentest FAST.  Professor Zeolla Gabriel Martín
# Primality test using functions that generate patterns.
#  (n) is PRIME CONFIRMATION or COMPOSITE NUMBER CONFIRMATION
# (n) WITHOUT CONFIRMATION and True  "it is prime or pseudoprime (Base 2 euler)
number = int(input("Enter Odd number ="))
if number % 2==0:
      print("ERROR, only Odd numbers")
 numb = (number -1) // 2

if numb % 4 == 1:
   z = (number - 1)// 2
elif numb % 4 == 2:
   z = (number - 1)// 2
elif numb % 4 == 0:
   z = (number + 1)// 2
else:
   z = (number + 1)// 2

counter=2
if z > 0:
     while z != 1:
        if z % 2:
            z = (z + number) // 2
        else:
            z //= 2
        counter=(counter +1)
        if (counter+1) > (((number-1)//2)+1):
                   break
else:
    print("The number entered is incorrect")
if ((number-1) //2)== (counter-1):
   print("Complete cycle ")
else:
        print("Incomplete cycle, has",((number-1) //2) /(counter-1 ),"patterns")
        print ("The ", number, "has a cycle of:",(number-1) //2 )
r=numb % (counter-1)
print("Possible prime number!! " ,r==0 and z==1)
if numb / (counter-1) ==1 and z==1:
   print("PRIME NUMBER CONFIRMATION")
if (r==0 and z==1)== False:
   print("COMPOSTE NUMBER CONFIRMATION")
```

Conclution

After the function is executed, all prime numbers end in $f_0 = 1$. Which shows us a totally unknown and interesting new feature of prime numbers.
The Alpha and Beta program is an effective technological tool to test the primality of numbers, it helps us to understand and reflect on the behavior of their cycles and patterns.
The development of the Beta program for the elimination of pseudo-prime numbers is very interesting.
While the pseudo prime number sequences are expressed in the OEIS encyclopedia.
There is no paper or text that refers to this function as it is presented and developed in this document, nor did I find a primality test with these characteristics
This function is built based on the number 2. But we can build infinite sequences by changing the base.

Professor Zeolla Gabriel M
San Vicente, Buenos. Aires. Argentina
2021

Download the spreadsheet to check more numbers and play with prime numbers.
https://www.academia.edu/50803638

Other works on prime numbers of the author
https://independent.academia.edu/GabrielZeolla

References

BECKER, M. E.; PIETROCOLA, N. Y SÁNCHEZ, C. (2001); Aritmética, Red Olímpica, Argentina.
GRACIÁN, E. (2011); Los Números Primos, un Largo Camino al Infinito, Navarra: EDITEC.
GÓMEZ, J. (2011); Matemáticos, Espías y Piratas Informáticos, Codificación y Criptografía, Navarra: EDITEC
Papadimitriou, Christos H.: Computational Complexity. Sección 10.2: "Primality", pp.222–227. Addison-Wesley, 1era edición, 1993. (ISBN201-53082-1.)
Caldwell, Chris,Finding primes & proving primality [1]
Caldwell, Chris: The Prime Pages. Universidad de Tennessee. (Ver enlaces externos.)
Agrawal, Manindra; Kayal, Neeraj; Saxena, Nitin: "PRIMES is in P". Annals of Mathematics 160 (2004), no. 2, pp. 781–793.
H. W. Lenstra jr. and Carl Pomerance: "Primality testing with Gaussian periods".
Ball, W. W. R. and Coxeter, H. S. M. *Mathematical Recreations and Essays, 13th ed.* New York: Dover, p. 61, 1987.
Beiler, A. H. *Recreations in the Theory of Numbers: The Queen of Mathematics Entertains.* New York: Dover, 1966.