Collatz's Conjecture, proposal, solution and analysis

Daniel Olivares

Abstract

The Collatz conjecture, also known as the **3n+1** conjecture or Ulam conjecture, is a mathematical puzzle that has baffled researchers for decades. It was stated by the mathematician Lothar Collatz in 1937, and to this day it remains unsolved.

The conjecture is based on a simple operation applicable to any positive integer:

If the number is **even**, divide by 2.

1. If the number is **odd**, multiply it by 3 and add 1.

We repeat this operation successively, taking the result of each step as input for the next. The central question is: will we always reach the number **1**, regardless of the initial value of **n**.

Example:

- For n=13

The sequence is periodic: 13, 40, 20, 10, 5, 16, 8, 4, 2, 1.

- For n=6

The sequence has 8 steps: 6, 3, 10, 5, 16, 8, 4, 2, 1.

- For n=27

The sequence has 111 steps before reaching 9232 and finally descending to 1.

Introduction

The Collatz conjecture has baffled mathematicians for decades due to its apparent simplicity and the lack of a formal proof. In the next Paper we will address a possible solution to the conjecture by modifying it and the reasons for it and we will analyze determining factors for it. all conditions are met.

Proposal

To find a solution to the **3n+1** problem, we must change the way we perceive it, and I propose it in this way by making a modification that, although it does not affect the results obtained with **3n+1**, helps us understand the reason for the results.

We can represent 3n+1 as **3(n1)+1**, and this will not alter our results, however, to understand it we must leave the **"1"** aside, because by expecting the final result to remain **"1"**, we stagnate.

I propose the ability to replace **"1"** with any odd integer (Z), I chose to use the letter **"i"**, as it is the first letter of the word "**Impar**" odd in Spanish, leaving it like this:

3(ni) + i

With this modification, instead of always ending in **"1"**, we will now say that it always ends in "i".

This rule must be followed: since we are replacing **"1"** which is an odd number with **"i"**, then we say that "i" must be an odd integer (Z).

Example:

- For i = 1 and n = 13i

The sequence is: 13, 40, 20, 10, 5, 16, 8, 4, 2, 1.

We have completed the sequence of the original conjecture using the modified one without altering its results.

Now, we will use the same integer (Z) but negative.

- For i= -1 and n= 13i

The sequence is: -13, -40, -20, -10, -5, -16, -8, -4, -2, -1.

Let's check what happens when changing “i” for another odd integer:

- For i= 3 and n= 13i

The sequence is: 39, 120, 60, 30, 15, 48, 24, 12, 6, 3.

- For i= -3 and n= 13i

The sequence is: -39, -120, -60, -30, -15, -48, -24, -12, -6, -3.

Successfully ending in **“i”** in both cases.

We can observe that the order of even and odd when changing **“i”** was maintained, although in both we obtained different numbers, the order of even and odd did not change due to having used the same value of **“n”**. As can be seen below:

- For i = 1 and n=13i

The sequence is: 13, 40, 20, 10, 5, 16, 8, 4, 2, 1.
Order: odd, even, even, even, odd, even, even, even, odd.

- For i = 3 and n= 13i

The sequence is: 39, 120, 60, 30, 15, 48, 24, 12, 6, 3.
Order: odd, even, even, even, odd, even, even, even, odd.

Even changing **“i”** to huge numbers maintains the order.

- For i = (2^{256} - 1) and n = 13i

The sequence is:
1505297160085110540506422805112942802092509800653327332512948592102870685319155,
4631683569492647816942839400347516314130799386625622561578303360316525185597400,
2315841784746323908471419700173758157065399693312811280789151680158262592798700,
1157920892373161954235709850086879078532699846656405640394575840079131296399350.

5789604461865809771178549250434395392663499233282028201972879200395656
48199675,
1852673427797059126777135760139006525652319754650249024631321344126610
074238960,
9263367138985295633885678800695032628261598773251245123156606720633050
37119480,
4631683569492647816942839400347516314130799386625622561578303360316525
18559740,
2315841784746323908471419700173758157065399693312811280789151680158262
59279870,
1157920892373161954235709850086879078532699846656405640394575840079131
29639935.
Order: odd, even, even, even, odd, even, even, even, odd.

This makes us understand that we will always preserve the order of even and odd when changing **“i”** if we keep the same value of **“n”**. Therefore it gives strong veracity to affirm that the Collatz conjecture will always end in **“1”** or in our case **“i”** no matter how large the numerical values are.

Analysis

All results end in a loop when **“i”** is found in the sequence.

- For:

i= 3
n=3i
The sequence is: 9, 30, 15, 48, 24, 12, 6, 3.

By finding **“i”**, which is always odd, we force “i” to multiply by **3+i** which it is equal to **i4**, and since every number multiplied by an even number results in an even number and every odd number multiplied by 4 can be divided by 2 twice to reach the same odd number, the infinite loop **i4, i2, i** is created, in our case: **3x4=12, 12/2=6, 6/2=3.**

We always find **“i”** at the end of the sequence because by adding **“i”** each time we multiply **“3n”** we are converting **“n”** into a multiple of **“i”**.

With respect to the fluctuation and chaotic tendency to increase and reduce the numerical sequence, and then in the end fall to **“i”** in some numbers such as 27, it is conditioned by the limitation that multiplication has in conjecture that it can only happen one step at a time, that is, there must always be a division after each multiplication.

When the sequence increases, the most common case is that patterns similar to this one repeat for short periods:
Odd, even, odd, even, odd, even, odd, even, odd, even, odd.

But division always predominates because it can occur whenever the result is even without limitations.

Python code for testing

```
i = 1
n = 13 * i

def collatz(n):
    even_count = 0
    odd_count = 0
    sequence = []

    while n != i:
        sequence.append(n)
        if n % 2 == 0:
            even_count += 1
            n = n // 2
        else:
            odd_count += 1
            n = (3 * n + i)

    sequence.append(n)
    return sequence, even_count, odd_count

result, even, odd = collatz(n)

print(f"The Collatz sequence for i = {i} ends at {result}.")
print(f"Even numbers encountered: {even}")
print(f"Odd numbers encountered: {odd}")
```