REVISED COLLATZ GRAPH EXPLAINS PREDICTABILITY

Steven M. Tylock
Rochester, NY
stylock@tylockandcompany.com

September 9, 2021

ABSTRACT

The Collatz Conjecture remains an intriguing problem. Expanding on an altered formula first presented in "Bits of Complexity", this paper explores the concept that "3N + Least-Significant-Bit" of a number allows the complete separation of the step of "dividing by two on an even number" within the Collatz Conjecture. This alternate formula replaces the original Collatz on a one-for-one basis. The breadth and depth of graphs of the resulting path of numbers resolving with this new formula illustrate its fractal nature. Lastly, we explore the predictability of this data, and how the ultimate goal of reaching one prevented previous work from finding the key to understanding 3N+1.

Keywords Collatz Conjecture · Graph Theory · 3N+LSB

1 Introduction

The Collatz Conjecture has been described by Lagarias [2010] as an "extraordinarily difficult problem". Part of that difficulty depends on the integer's value, optionally dividing by two or multiplying by three and adding one.

This paper expands on the author's method to separate these two actions such that division by two is not performed until the resulting integer is equal to two to a power, or a perfect power of two (2^z). A graph of this new formula fully documents all predecessors; provides bounds for the breadth and width of this graph; illustrates the fractal nature of successive graphs; demonstrates the predictability of predecessors; and can be seen as a simplification of a graph of the same values under the original Collatz.

2 3N + Least Significant Bit

As shown in "Bits of Complexity" [Tylock, 2018], the original Collatz Conjecture method of dividing even numbers by 2 and multiplying odd numbers by 3 and adding 1 can be replaced with "3N + Least Significant Bit" without changing the outcome. The ending state becomes "Is the number 2^z?" Let us review.

First, consider an odd number. The original Collatz specifies 3N + 1. The new method also specifies 3N + 1 because, as an odd number, its Least Significant Bit (LSB)[1] is 2^0, or 1.

Next, consider a number that is divisible by 2 but not 4. Following the original Collatz Conjecture, the number is divided by two and 3N + 1 is applied to the resulting odd number. Following the new method, 3N + 2 is applied to the number because its LSB is 2, or one power of 2 (2^1). When that outcome is divided by 2, the result is identical to the original Collatz. Futher, since the new method converges to the same number as the Collatz for one multiple of 2, it can be seen that this happens for any multiple of 2. Collatz divides by two at the start and middle of operations, but 3N + LSB delays all divisions to the end. Once a number has reached 2^z, the number is "solved" to 1 after dividing by 2, Z times. 3N + LSB has an identical number of steps / iterations as Collatz, and so the 3N + LSB formula is a perfect replacement of the original.

[1]The Least Significant Bit of a number is defined as the smallest power of two that is present in the binary representation of the number. Example - the number 12 is represented in binary as 4 + 8 or two to the power of two plus two to the power of three; its least significant bit is 4 or two to the power of two.

3 Predecessors

Because every application of the formula increases the numerical value, we can now consider all possible numbers that are predecessors of Y, or all X such that 3X + LSB(X) = Y. Predecessors can be calculated with the following algorithm.

Algorithm to Identify Predecessors:

For any number Y.

Function Name: Predecessors
Input Variable: Any positive integer, Y (The number whose predecessors we are interested in)

```
BEGIN
    Integer i = log_2(LSB(Y)) - 1
    WHILE (i ≥ 0)
        Potential_Predecessor = (Y - 2^i) / 3
        IF (Remainder((Y - 2^i) / 3) == 0) AND (LSB(Potential_Predecessor) == 2^i)
            THEN Potential_Predecessor is a Predecessor of Y
        i = i - 1
    END WHILE
END
```

As explanation, first, we know that the LSB(X), where X is any predecessor of Y, must be less than the LSB(Y), so we can start evaluating numbers whose LSB is at least smaller than Y's. Then, if we reverse the 3N + LSB transaction by subtracting the LSB and dividing the remainder by three, we will have a predecessor if there is both no remainder when we divide by three, and the LSB is equivalent.

The predecessor is that number that is multiplied by three and has its LSB added. After evaluating all possible bits smaller than the LSB(Y), we will have identified the set of all numbers that are predecessors of Y.

A special case exists for a number 2^z such that 2^z = 3N+LSB(N) where N = 2^a and z = a+2. Symbolically, this looks like: $2^z = 3N+LSB(N) = 3\times2^a+LSB(2^a) = 3\times2^a+2^a = 4\times2^a = 2^{a+2}$. Or stated differently, any power of two has two-less of the power of two as a predecessor. If one does not want to consider this case, the algorithm above must specifically test for and exclude it.

It should also be noted at this point that any number that is a multiple of three can never have a predecessor (as shown in "Bits of Complexity").

4 Graph

Since this new method does not loop, a graph can be made to show how numbers simplify towards 2^z. To do this, consider any power of two, determine its predecessors, and recursively find each predecessor's predecessors.

This tree of predecessors can then be organized in a directed graph where each node represents a number and the edges represent the application of 3N + LSB.

The graph of 2^{15} is presented in Figure 1 on the next page. This graph shows the clear progression of all numbers that resolve at 2^{15} (32768).

5 Breadth

An analysis of all possible predecessors of a number reveals that there is one "root" predecessor, and that all successive predecessors include every bit of the root, any immediate prior predecessors, and $\frac{LSB(previous\_predecessor)}{4}$. These predecessors continue until the LSB is 2^1 or 2^0 (such that there is no integer value for $\frac{LSB}{4}$).

One can see this easiest by evaluating all numbers that directly transform into 352 (rewritten as $2^5 + 2^6 + 2^8$). The base for such a transformation is 112 (rewritten as $2^4 + 2^5 + 2^6$). The next value adds 4 (rewritten as 2^2), and the third adds 1 to the prior predecessor (rewritten as 2^0). Figure 2 below illustrates this.

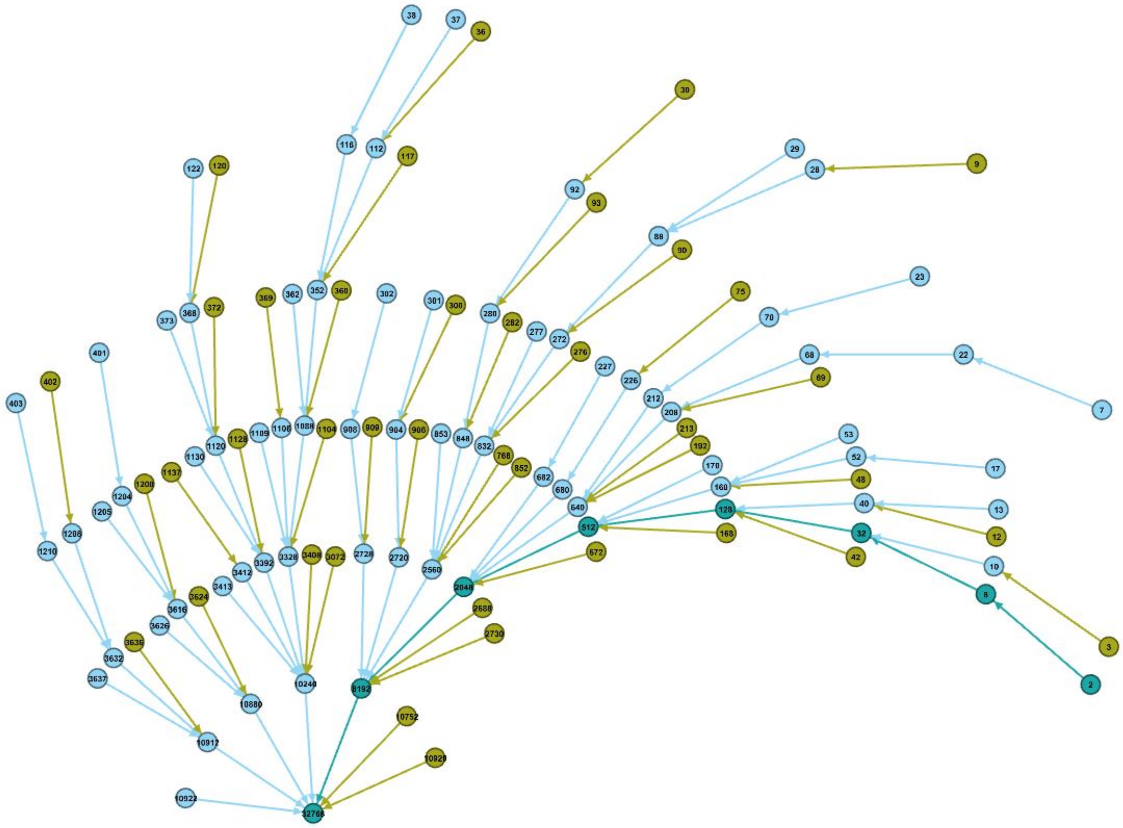

Figure 1: Directed graph of numbers that resolve at 2^{15}, or predecessors of 2^{15} (32768). Green indicates numbers divisible by 3. Dark Blue indicates perfect powers of 2.

	2^0 1	2^1 2	2^2 4	2^3 8	2^4 16	2^5 32	2^6 64	2^7 128	2^8 256
112	**0**	**0**	**0**	**0**	**1**	**1**	**1**		
116	0	0	1	0	**1**	**1**	**1**		
117	1	0	1	0	**1**	**1**	**1**		
	3N + LSB for each equal to:								
352	0	0	0	0	0	1	1	0	1

Figure 2: Table showing values that transform into 352 with 3N + LSB(N).

6 Depth

An analysis of the tree of all possible predecessors reveals a single bit predecessor (of a single bit number) retains the most space for further predecessors. The chain of predecessors $2^B \rightarrow 2^{B-2} \rightarrow 2^{B-4} \ldots\ 2^1$ or 2^0 contains the greatest quantity of values. One can generalize this to $\frac{B}{2}$ for an arbitrary power of 2.

Because we know that the predecessor of a number's LSB is smaller than its successor, the number of values in the chain of non-power-of-2 numbers is based on its Most Significant Bit and Least Significant Bit. It cannot have more predecessors than its $\frac{log_2(MSB)}{2}$ (based on the power of two calculations above), and it also cannot have more predecessors than its $\log_2$(LSB)-1.

7 Fractal

Another interesting aspect of the transformation is the repetitive nature of the graphs. This is because given a graph of predecessors for 2^N, the entire graph will be present in the graph of 2^{N+2} with each value replaced with itself multiplied by 4. New elements in the graph of 2^{N+2} will also include predecessors for every node that already has predecessor(s) (in the same formulation as explained above).

And so the graph of 2^{11} has the same shape as the graph of 2^9.

Referring to the original Collatz, the hailstone pattern of sequences has been considered part of the "interesting exercise" of the Collatz [Banks, 2012]. The separation of the division by 2 substantially alters the graph, removes the hailstone aspect, and allows the formula's progression to be more fully seen.

8 Predictability

Additional information can be revealed by evaluating the successive predecessors of two to the power. Collatz enthusiasts have long understood that most numbers have an easy solution – those can be seen in the incidents of repeat, successive predecessor, and numbers divisible by three (examples shown in Figure 3 below).

We can also chart the predecessors for each power of two on a log scale, see Figure 4 below with a logarithmic Y axis. The blue line indicates the count of all predecessors for each power. Interestingly, if you start a line at 2^{10} which has 11 predecessors and chart the growth at four thirds of that for each power after, that value (displayed in yellow) nearly exactly matches the growth of the actual number of predecessors in blue.

9 Obscuring Conclusions

The author's novel method provides a predictable polytree whose labels all ascend in value while the Collatz conjecture rises and falls as symbolized by the hailstone designation. Figure 5 contains the same numbers examined in Figure 1, but organized following the Collatz Conjecture. It can thus be seen that the original Collatz obscured the patterns within the 3N+1 formula by dividing by 2 stepwise. Examining Figures 1 and 5 side by side best illustrates the benefits of delaying the division by 2. But because the ultimate goal as stated by Collatz was to reduce the number to 1, altering this division was anathema to solving the puzzle. The predictability of 3N+LSB now points at a proof.

Power of Two	Two to the Power	$(4/3)^{\text{Power of Two}}$	Number of Predecessors	# Predecessors of Previous Power of Two	As a Percentage of Total Predecessors	Total New Predecessors	Successive Predecessors	As a Percentage of Total Predecessors	Predecessors with a New Bit Pattern	As a Percentage of Total Predecessors	Predecessors Divisible by Three	As a Percentage of Total Predecessors	Total Number of Endpoints	As a Percentage of Total Predecessors	Endpoints Not Divisible by Three
1	2		0												
3	8		0												
5	32		2	0	0%	2	0	0%	2	100%	1	50%	1	50%	0
7	128	3.6	4	2	50%	2	2	50%	0	0%	2	50%	3	75%	1
9	512	7.1	7	4	57%	3	2	29%	1	14%	2	29%	5	71%	3
11	2048	12.4	17	7	41%	10	3	18%	7	41%	5	29%	8	47%	3
13	8192	30.2	30	17	57%	13	10	33%	3	10%	13	43%	18	60%	5
15	32768	53.3	49	30	61%	19	13	27%	6	12%	17	35%	31	63%	14
17	131072	87.1	90	49	54%	41	19	21%	22	24%	30	33%	50	56%	20
19	524288	160.0	158	90	57%	68	41	26%	27	17%	52	33%	91	58%	39
21	2097152	280.9	281	158	56%	123	68	24%	55	20%	93	33%	159	57%	66
23	8388608	499.6	504	281	56%	223	123	24%	100	20%	174	35%	282	56%	108
25	33554432	896.0	884	504	57%	380	223	25%	157	18%	294	33%	505	57%	211
27	134217728	1571.6	1589	884	56%	705	380	24%	325	20%	524	33%	885	56%	361
29	536870912	2824.9	2828	1589	56%	1239	705	25%	534	19%	953	34%	1590	56%	637

Figure 3: Statistical information about Predecessors of odd powers of two.

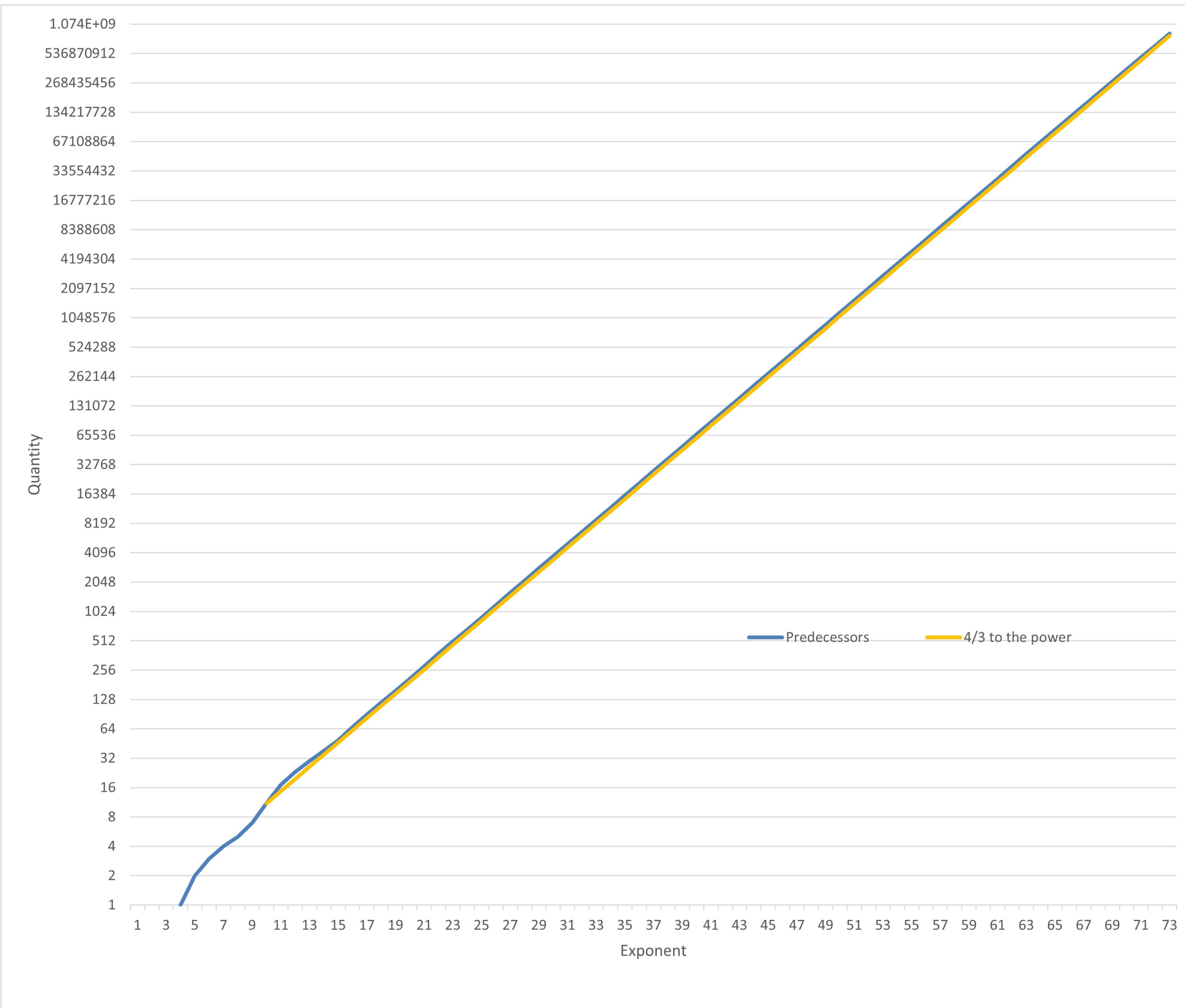

Figure 4: Log-Graph of the number of predecessors of two to the power (blue) and growth at four thirds (yellow).

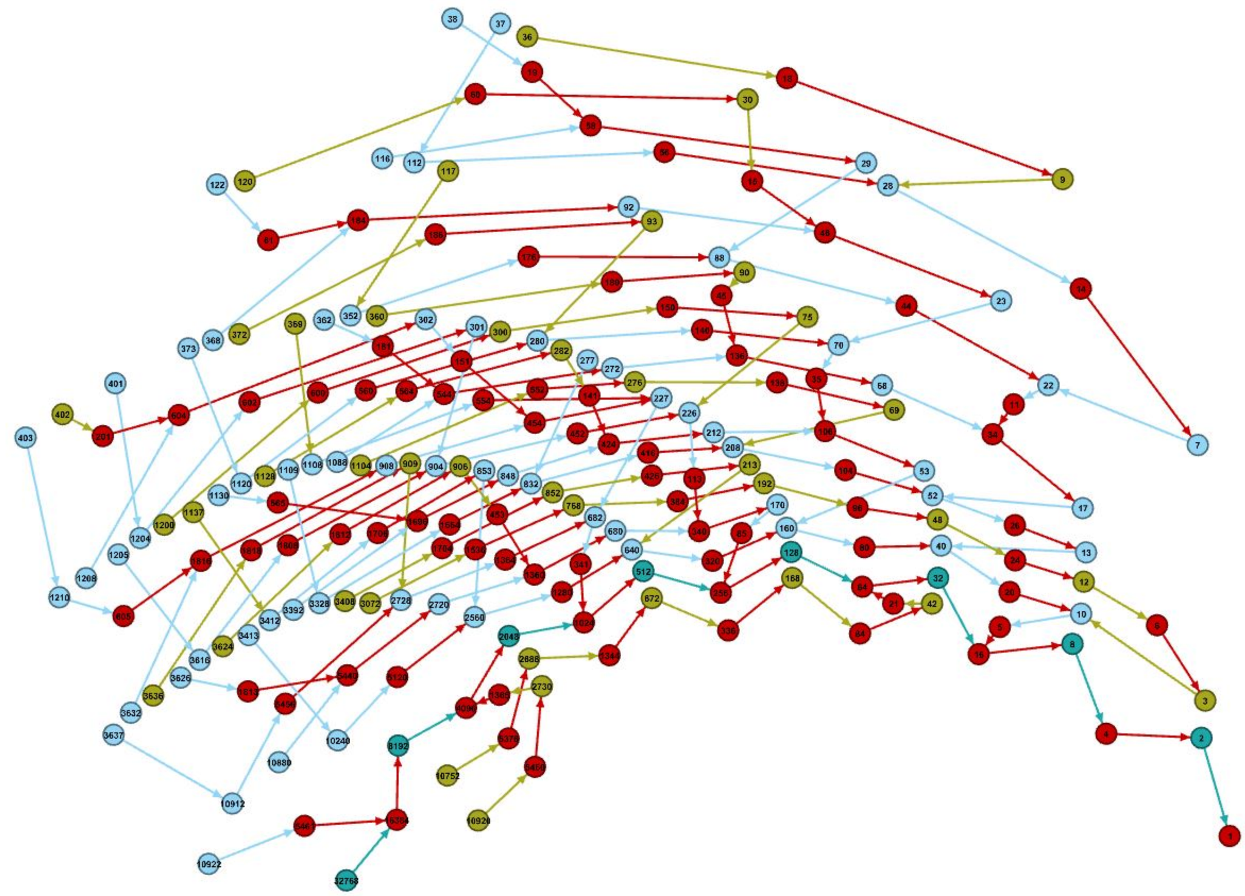

Figure 5: Graph of predecessors of 2^{15} (32768) under original Collatz. Identical nodes from Figure 1 with new red nodes inserted as needed after division by 2.

10 Appendix - Excerpt from Bits of Complexity with permission

A Replacement Function - Consolidation
Let us define a number's least significant bit as the smallest power of 2 that is added together to create its binary representation. The (decimal) number 12 can be considered in binary notation as 2^3 plus 2^2 or eight plus four; its least significant bit is 2^2.

Let us now construct a new Consolidation Function labeled CF. This function performs a single operation, multiplying a number n by three and adding n's least significant bit (LSB).

CF(n) = 3 * n + LSB(n)

Mapping

Now we will map the Consolidation Function (CF) to the Collatz (C). The Consolidation Function is equivalent to delaying all "divide by 2" steps of the Collatz Conjecture until a perfect power of 2 is reached. This can be illustrated with the following example of processing 3.

Collatz
C(3) = 3 * 3 + 1 = 10
C(10) = 10 / 2 = 5
C(5) = 3 * 5 + 1 = 16
C(16) = 16 / 2 = 8
C(8) = 8 / 2 = 4
C(4) = 4 / 2 = 2
C(2) = 2 / 2 = 1
Consolidation Function
CF(3) = 3 * 3 + 1 = 10
CF(10) = 3 * 10 + 2 = 32
$32 = 2^5$

Further, the power of 2 that the Consolidation Function stops at provides the number of "divide by 2" steps that would normally be a part of the Collatz. The total number of steps to process a number with the Collatz function is the same as the Consolidation Function steps to reach a power of 2, plus the power of 2 itself (5 in the case of processing 3 above).

Confirmation

Let us more formally show that to be true for the processing of an arbitrary number n greater than 1. Consider three cases of n, n is a perfect power of 2, n is odd, and n is even.

n = 2^x

Presents a special case, the number of iterations of "divide by 2" for the Collatz will be equal to x. The Consolidation Function will not process at all (as the ending state has already been reached), and it will also show that the number of steps to reach 1 will also be x.

n = odd

If n is an odd number, that means that the least significant bit of n is 1, or 2^0. The Consolidation Function's processing will be 3n + 1, which is exactly the same as the Collatz.

n = even

If n is an even number, we can represent it as $2^a * b$, where b is an odd number. The next set of iterations of the Collatz function will be to divide n by 2 exactly a times, and then perform the 3N + 1 operation (3b+1). For its counterpart, given the LSB of n is 2^a, the Consolidation Function will calculate 3N + LSB = $3n + 2^a$. That can be shown to be equivalent to $2^a * (3b + 1)$.

$n = 2^a * b$

$LSB(n) = 2^a$

$CF(n) = 3(n) + LSB$

$CF(n) = 3(2^a * b) + 2^a$

$CF(n) = 2^a * (3b) + 2^a * (1)$

$CF(n) = 2^a * (3b + 1)$

This value can also be divided by 2 exactly a times to reach the final value of the Collatz.

As any number of repetitions of the Collatz would operate in the same manner, at the conclusion of the Consolidation Function, it has completed the same number of 3N+LSB operations as the Collatz's 3N+1. The value of the exponent 2^x that the Consolidation Function has reached equals the number of divide by 2 steps the Collatz would have completed.

Further, if it can be shown that the Consolidation Function does in fact end at a power of 2 for all n, then the Collatz Conjecture also terminates for all n.

10.1 Lemma 1

Lemma 1 – No number that is divisible by 3 has a predecessor.

Establishing this helps understand the nature of the 3N + LSB function.

Let us take some number P that is divisible by 3. In order for P to have a predecessor R, there must be some LSB of R such that 3 * R + LSB(R) = P. We can also represent P instead as 3 * Q (by definition – because P is divisible by 3).

$P = 3Q$

$3 * R + LSB(R) = P$

$3 * R + LSB(R) = 3Q$

$LSB(R) = 3Q – 3R$

$LSB(R) = 3(Q – R)$

And this shows us that there can be no LSB of R meeting this equation – because the equation indicates that the only possible values for R's Least Significant Bit would also be multiples of three – and there are no powers of two that are also multiples of three.

10.2 Lemma 2

Lemma 2 – Any number having a predecessor with 0s in the bits representing 2^0 and 2^1, has one or more additional predecessors.

Let us consider some number S that has a predecessor T having this property. The bits of T representing 2^0 and 2^1 represent 1 and 2 in the binary scale, and if these are 0, we know that the smallest portion (LSB) of T is at least as large as 2^2 or 4.

When calculating 3 * T + LSB(T), we know that the LSB(T) is at least 2^2 or 4.

The number U that is equivalent to T + LSB(T)/4 will also be a predecessor of S.

Because the LSB(T) is at least 4, we know that there is at least one number fulfilling the binary representation of LSB(T)/4.

The calculation of 3 * T + LSB(T) is equivalent to 3 * U + LSB(U).

U = T + LSB(T)/4

LSB(U) = LSB(T/4)

Starting with 3 * U + LSB(U) and substituting for U and LSB(U)

3 * U + LSB(U) = 3 * (T + LSB(T)/4) + LSB(T)/4

3 * U + LSB(U) = 3 * T + 3 * LSB(T)/4 + LSB(T)/4

3 * U + LSB(U) = 3 * T + LSB(T)

The result of the Consolidation Function, CF(U) and CF(T) are identical.

10.3 Lemma 3

Lemma 3 – After two repetitions of the 3N + LSB function, for all cases of N, the new number's bits representing 2^0 and 2^1 will be 0.

Let us consider some number V whose bits representing 2^0 and 2^1 are both 1. On the first iteration of 3N + LSB, the LSB is 2^0 or 1. And so 3N + LSB will calculate a value of 4 (2^2) for that bit (3 * 1 + 1), and 2^0 will be 0 afterwards. On the other hand, the 2^1 bit that was set will still be set after being multiplied by 3 (becoming 2^1 plus 2^2).

And so after one iteration of 3N + LSB, the 2^0 bit will be 0, and the 2^1 will still be 1.

Repeating 3N + LSB will similarly clear the new LSB, or 2^1.

Given the Consolidation Function only multiplies by three and adds the Least Significant Bit, there is no opportunity for bits smaller than a number's LSB to ever hold data again. This state is irreversible.

11 Acknowledgement

The author would like to thank Rachael Tylock for review, comments, and LaTeX conversion support in finalizing this work.

References

Robert B. Banks. *Slicing Pizzas, Racing Turtles, and Further Adventures in Applied Mathematics*, Chapter 9: Great Number Sequences: Prime, Fibonacci, and Hailstone. Princeton University Press, 2012. ISBN 9780691154992.

Jeffrey C Lagarias, editor. *The Ultimate Challenge: the 3x + 1 problem*. American Mathematical Society, 2010. ISBN 9780821849408.

Steven M. Tylock. *Bits of Complexity: Collatz, Twinship, and Perspective – Discovering Relationships*, Appendix. Tylock and Company, LLC, 2018. ASIN: B07DYTF34R.