

Methodological Generalization of the Collatz Sequences to $(1 + 2^k)n + S_k(n)$, $n/2^k$ with Computational Verification for $k = 1$ up to $k = 20$

Ammar HAMDOUS 

September 20, 2026

Abstract

While attempts to prove the Collatz conjecture have so far culminated in the work of Terence Tao (2019), which establishes “Almost all orbits of the Collatz map attain almost bounded values” [3], several attempts have been made to generalize the Collatz sequences $3n + 1$, but many of them produced sequences that lack the essential structural properties of the original Collatz dynamics. Among these, the most promising known generalization is the one proposed in 2022 by Naouel Boulkaboul [2], which takes the form $3n + 3^k$ and leads sequences to converge toward 3^k . In this work, we propose a new methodological generalization of the Collatz sequences based on a two-part transformation $(1 + 2^k)n + S_k(n)$ if $n \bmod 2^k \neq 0$, and $n/2^k$ if $n \bmod 2^k = 0$, where $S_k(n)$ is a correction function preserving the generalized singularity previously revealed in [1]. This revised formulation ensures that all rank-1 branch beginnings exhibit the generalized singularity in binary form.

1 Introduction

In the present paper, we do not attempt to prove the Collatz conjecture. Instead, we introduce a methodological generalization of the classical Collatz $3n + 1$, building upon the structural findings of the singularity pattern described in a previous preprint [1]. The goal is to define a generalized transformation that preserves the binary singularity and the convergent behavior of

the original sequences while extending to a parameterized form. Initially, the generalization was defined using only the odd transformation $(1+2^k)n+S_k(n)$. However, this led to the appearance of rank-1 branch beginnings that do not exhibit the generalized singularity. To address this, we revised the g_k auxiliary function to $n/2^k$ if $n \bmod 2^k = 0$. This paper includes a computational investigation over a bounded domain, with $\mathbf{k}$ ranging from **1** to **20**. The observed numerical behavior indicates that the corresponding sequences remain bounded and eventually reach **1** within this range. These computations are presented as illustrative evidence, serving to test internal consistency and to motivate the proposed generalization, rather than as a proof or confirmation of convergence.

2 Reminder:

Let $\mathbb{N} = \{0, 1, 2, \dots\}$, and $\mathbb{N}^* = \mathbb{N} \setminus \{0\}$

In our previous work [1] we revealed a singularity for the $3n + 1$ Collatz sequence as follows:

The singularity consists in having or reaching a term in the sequence whose binary form satisfies the following condition:

- all bits at even indices are equal to **1**.
- and all bits at odd indices are equal to **0**.

In other words, the singularity term begins and ends with a bit equal to **1**, and each two consecutive bits equal to **1** are separated by one bit equal to **0**.

10 - - 0101

3 Generalization of the $3n + 1$ Singularity

We define a *generalized singularity* as having or reaching a term in the generalized Collatz sequences whose binary representation satisfies the following conditions:

- It begins and ends with k consecutive bits equal to **1**.
- Each two consecutive groups of k consecutive **1**s are separated by k consecutive **0**s.

- The number of groups of k consecutive bits equal to **1** is greater than or equal to 1.

$$1 \dots 110 \dots 00 \text{---} 1 \dots 11 \underbrace{0 \dots 0}_k \underbrace{1 \dots 1}_k$$

4 Deduction of f_k function from the singularity

In the previous preprint [1], we defined the auxiliary function f_3 as follows:

$$f_3 : \mathbb{N} \rightarrow \mathbb{N}$$

$$f_3(n) = 3n + 1$$

Now, how can a term **a** that qualifies as a generalized singularity be transformed into a power of 2^k by a single iteration of f_k function, which is the generalization of the function f_3 ?

To achieve this, we simply add to the term **a** the quantity $2^k \cdot a$, such that the resulting binary representation has a multiple of k bits equal to **1**. That is, the structure of **a** and $2^k \cdot a$ must be aligned in a way that all **1-bits** in the binary representation of **a** are complemented by **0-bits** in the binary representation of $2^k \cdot a$, and vice versa.

$$a = 1 \dots 110 \dots 00 \dots 001 \dots 11$$

$$2^k \cdot a = 1 \dots 110 \dots 001 \dots 11 \dots 110 \dots 00$$

$$a + 2^k a = (1 + 2^k)n = 1111111111111111 \dots 11111111$$

(the binary number consisting entirely of 1s)

The addition of **a** and $2^k \cdot a$ gives a natural number with a binary form composed only by a multiple of k number of bits equal to 1. Now to transform $a + 2^k a = (1 + 2^k) a$ into a power of 2^k , all we need is adding **1**, but we will expose later the real meaning of adding **1** to $(1 + 2^k) a$ that we have used here.

So we almost finished the definition of the auxiliary function f_k for the generalized Collatz sequences, as multiplying the odd term by $(1 + 2^k)$, then adding **1** to the result, but this definition will be adjusted later.

We have transformed a generalized singularity term **a** by just one iteration of f_k into a power of 2^k term, and we transformed the f_2 auxiliary function to

$$\frac{n}{2^k}$$

4.1 Adjusting the f_k function

4.1.1 Reason why we should adjust the f_k function

If we consider that f_k is defined by $f_k(n) = (1 + 2^k) n + 1$, then let us take $k = 2$ as an example. We will now examine the behavior of $f_k(n)$ for $n = 1$. The first 113 terms of the sequence are:

1st: 1	2nd: 6	3rd: 31
4th: 156	5th: 39	6th: 196
7th: 49	8th: 246	9th: 1231
10th: 6156	11th: 1539	12th: 7696
13th: 1924	14th: 481	15th: 2406
16th: 12031	17th: 60156	18th: 15039
19th: 75196	20th: 18799	21st: 93996
22nd: 23499	23rd: 117496	24th: 29374
25th: 146871	26th: 734356	27th: 183589
28th: 917946	29th: 4589731	30th: 22948656
31st: 5737164	32nd: 1434291	33rd: 7171456
34th: 1792864	35th: 448216	36th: 112054
37th: 560271	38th: 2801356	39th: 700339
40th: 3501696	41st: 875424	42nd: 218856
43rd: 54714	44th: 273571	45th: 1367856
46th: 341964	47th: 85491	48th: 427456
49th: 106864	50th: 26716	51st: 6679
52nd: 33396	53rd: 8349	54th: 41746
55th: 208731	56th: 1043656	57th: 260914
58th: 1304571	59th: 6522856	60th: 1630714
61st: 8153571	62nd: 40767856	63rd: 10191964
64th: 2547991	65th: 12739956	66th: 3184989
67th: 15924946	68th: 79624731	69th: 398123656
70th: 99530914	71st: 497654571	72nd: 2488272856
73rd: 622068214	74th: 3110341071	75th: 15551705356
76th: 3887926339	77th: 19439631696	78th: 4859907924
79th: 1214976981	80th: 6074884906	81st: 30374424531
82nd: 151872122656	83rd: 37968030664	84th: 9492007666
85th: 47460038331	86th: 237300191656	87th: 59325047914
88th: 296625239571	89th: 1483126197856	90th: 370781549464
91st: 92695387366	92nd: 463476936831	93rd: 2317384684156
94th: 579346171039	95th: 2896730855196	96th: 724182713799
97th: 3620913568996	98th: 905228392249	99th: 4526141961246
100th: 22630709806231	101st: 113153549031156	102nd: 28288387257789
103rd: 141441936288946	104th: 707209681444731	105th: 3536048407223656
106th: 884012101805914	107th: 4420060509029571	108th: 22100302545147856
109th: 5525075636286964	110th: 1381268909071741	111th: 6906344545358706
112th: 34531722726793531	113th: 172658613633967656	

As you can see, the sequence seems to grow to infinity, and there is no trivial cycle as in the original $3n + 1$ Collatz sequence.

4.1.2 Finding the Best Alternative to $+1$ part in f_k

The adjustment of the function f_k , which we have so far defined as $(1+2^k)n+1$, concerns only the $+1$ part, and It will be made by replacing this part with the best alternative that satisfies the following four conditions:

1st Condition: Offering the ability to integrate the Collatz sequence $3n + 1$ for a certain value of k .

2nd Condition: Maintaining the occurrence of the $+1$ part in f_k , which we previously used to transform $(1 + 2^k) a$ into a power of 2^k , when a is a generalized singularity term.

3rd Condition: There must be a generalized trivial cycle starting from 1.

4th Condition: Preventing the generalized Collatz sequences from diverging to infinity, in such a way that convergence to **1** remains possible

First let **a** be a generalized singularity term, and let apply the f_k iteration, the result is:

$$\begin{aligned} a &= 1 \dots 110 \dots 00 \dots 001 \dots 11 \\ 2^k \cdot a &= 1 \dots 110 \dots 001 \dots 11 \dots 110 \dots 00 \\ a + 2^k a &= (1 + 2^k)a = 1111111111111111 \dots 11111111 \\ &\quad (a \text{ binary number consisting entirely of } 1s) \end{aligned}$$

We have that the $(1 + 2^k) \cdot a$ par of f_k gives a $n \in \mathbb{N}^*$ whose binary form is composed only by bits equal to **1**, and the number of these bits is a multiple of k , so the $+1$ part of f_k will transform $(1 + 2^k) \cdot a$ into a power of 2^k , with a binary form composed of the same multiple of k bits equal to **0**, followed by a single bit equal to **1**. So, in the case of a generalized singularity term a , we have: $f_k(a) \equiv 0 \pmod{2^k}$ That is to say, the $+1$ part of f_k allows $f_k(a)$ to be divisible by 2^k .

But in the general case, where **n** is not necessarily a generalized singularity term, we have: $n \equiv 1 \pmod{2}$ and $(1 + 2^k) \equiv 1 \pmod{2}$ So $(1 + 2^k)n \equiv 1 \pmod{2}$ and by the adding of $+1$ part of f_k we get: $f_k(n) = (1 + 2^k)n + 1$ such that $f_k(n) \equiv 0 \pmod{2}$ Thus, the $+1$ part of f_k allows only to get an even term $f_k(n)$

We consider that the real meaning of $+1$ part of f_k is to make $f_k(n) \equiv 0 \pmod{2^k}$ and we can justify this by the fact that when we make $f_k(n) \equiv 0 \pmod{2^k}$ we are aiming to satisfy the 4th Condition, since any no congruent

to $\mathbf{0}$ modulo 2^k term $\mathbf{n}$ when it is multiplied by $(1 + 2^k)$ will be divisible by 2^{pk} with $p \in \mathbb{N}^*$ by adding what we call in Modular Arithmetic the smallest non-negative additive inverse of $(1 + 2^k)n \bmod 2^k$, and we represent it by the function: S_k

4.1.2.1 Definition of the S_k function

The smallest non-negative additive inverse of $(1 + 2^k)n \bmod 2^k$ can be defined as the smallest $s_k(n) \in \mathbb{N}^*$ that we need to add to $(1 + 2^k)n$ such that $(1 + 2^k)n + s_k(n)$ will be divisible by 2^k .

In general, we have:

$$S_k(n) = [2^k - ((1 + 2^k)n \bmod 2^k)] \bmod 2^k$$

But since $(1 + 2^k)n \bmod 2^k$ is always positive, because $(1 + 2^k)n$ is always in $\mathbb{N}^*$ then:

$$S_k(n) = 2^k - ((1 + 2^k)n \bmod 2^k), \text{ and since } (1 + 2^k)n = n + 2^k \cdot n, \text{ then}$$

$$S_k(n) = 2^k - (n \bmod 2^k)$$

S_k function statement:

Let $\mathcal{Q}$ be the set of all numbers in $\mathbb{N}^*$ that are not congruent to 0 modulo 2^k .

We can now define the function S_k , which gives the smallest non-negative additive inverse of $\mathbf{n}$ modulo 2^k , as follows:

$$S_k : \mathcal{Q} \rightarrow \{1, 2, \dots, 2^k - 1\}$$

$$S_k(n) = 2^k - (n \bmod 2^k)$$

4.1.2.2 Does S_k satisfy the 1st Condition?

If $k = 1$, then:

$$f_k(n) = (1 + 2^1)n + S_1(n) = 3n + S_1(n)$$

$$S_1(n) = 2^1 - (n \bmod 2^1) = 2 - 1 = 1$$

So if $k = 1$, then $f_k(n) = 3n + 1$, and the second auxiliary function for $k = 1$ will be $\frac{n}{2^1} = \frac{n}{2}$, then yes the $3n + 1$ Collatz sequences is integrated on the generalized Collatz sequences for $k = 1$.

So yes S_k satisfies the 1st Condition

4.1.2.3 Does S_k satisfy the 2nd Condition?

Let be $\mathbf{a}$ a generalized singularity term, when applying the f_k iteration $f_k(a) = (1 + 2^k)a + S_k(a)$ We know that the $(1 + 2^k) a$ part of f_k is a number in $\mathbb{N}^*$ whose binary form is composed only of bits equal to 1, and the number of these bits is a multiple of k , so if we add 1 we will get a power of 2^k .

So $S_k(a)$ is always equal to 1 when $\mathbf{a}$ is a generalized singularity term, which means that the occurrence of +1 part in f_k is always maintained when the term is a generalized singularity, that is to say S_k satisfies the 2nd Condition.

4.1.2.4 Does S_k satisfies the 3rd Condition?

We have:

$$\forall k \in \mathbb{N}^*, \forall n \in \mathbb{N}^*, \text{ if } n \bmod 2^k \neq 0 \text{ and } n < 2^k$$

then the term following $\mathbf{n}$ is obtained by an iteration via the auxiliary function f_k , and it is equal to:

$$f_k(n) = (1 + 2^k) \cdot n + s_k(n) = n + n \cdot 2^k + 2^k - n = (n + 1) \cdot 2^k$$

We also have:

$$\forall k \in \mathbb{N}^*, \forall n \in \mathbb{N}^*$$

if $(n + 1) \bmod 2^k \neq 0$, then the term following $(n + 1) \cdot 2^k$ is obtained via the second auxiliary function, and it is equal to:

$$\frac{(n + 1) \cdot 2^k}{2^k} = (n + 1)$$

So, from these two rules, and taking $\mathbf{1}$ as the first term of the generalized Collatz sequence, we get the following sequence:

$$\{1, 2 \cdot 2^k, 2, 3 \cdot 2^k, 3, \dots, 2^k \cdot 2^k, 2^k\}$$

And since the term that succeeds the term 2^k is:

$$\frac{2^k}{2^k} = 1$$

We conclude that there is a trivial generalized cycle of the form:

$$\{1, 2 \cdot 2^k, 2, 3 \cdot 2^k, 3, \dots, 2^k \cdot 2^k, 2^k\}$$

So yes S_k satisfies the 3rd Condition

4.1.2.5 Does S_k satisfies the 4th Condition?

By choosing the S_k function, we are aiming to satisfy the 4th Condition, by preventing the generalized Collatz sequences from diverging to infinity, in such a way that convergence to 1 remains possible, and to finally proof that the S_k is the best alternative to +1 part in f_k auxiliary function we need to proceed to a Computational Verification over a moderate range of the Generalized Collatz Conjecture $(1 + 2^k)n + S_k(n)$ if $n \bmod 2^k \neq 0$, and $n/2^k$ if $n \bmod 2^k = 0$ to verify if for all $k \in \mathbb{N}^*$ all the generalized sequences tested converge to **1**

5 Computational Verification over a moderate range of the Generalized Collatz Sequences $(1 + 2^k)n + S_k(n), n/2^k$

5.1 Efficiency Improvements of the algorithm

5.1.1 Early Convergence Criteria

Let $b \in \mathbb{N}^*$ such that $b > 1$.

And let

$\{1, 2, \dots, b\}$ be the set of the first b numbers in $\mathbb{N}^*$.

If all numbers in $\mathbb{N}^* \setminus \{1, 2, \dots, b\}$ converge to 1, then any number in $\mathbb{N}^*$ that converges to an element of that set, will converge to **1**.

This reflects the idea that a sequence doesn't need to reach **1** to validate the conjecture it's sufficient that it reaches a previously tested smaller value, which ensures it will eventually reach **1**.

To test the convergence to **1** of the generalized Collatz sequences for $k \in \{1, 2, 3, \dots, 20\}$

- $[1, 2^{30}]$ for $0 < k < 6$
- $[1, 2^{29}]$ for $k = 6$

- $[1, 2^{27}]$ for $k = 7$
- $[1, 2^{26}]$ for $k = 8$
- $[1, 2^{25}]$ for $k = 9$
- $[1, 2^{25}]$ for $k = 10$
- $[1, 2^{24}]$ for $k = 11$
- $[1, 2^{23}]$ for $k = 12$
- $[1, 2^{22}]$ for $k = 13$
- $[1, 2^{21}]$ for $k = 14$
- $[1, 2^{20}]$ for $k = 15$
- $[1, 2^{19}]$ for $k = 16$
- $[1, 2^{18}]$ for $k = 17$
- $[1, 2^{17}]$ for $k = 18$
- $[1, 2^{16}]$ for $k = 19$
- $[1, 2^{15}]$ for $k = 20$

we use the improved algorithms **GC Tester**, but for $k = 2$ and $k = 4$, the **GC Tester** detected that there is a sequence with 140 millions iterations without reaching a lower value so the program gives message telling that we must use **GC Tester2** which don't use the efficiency improvement, and to use this program we need first search the non-trivial cycles with a third program **Non Trivial Cycles**. for generating the histogram, we use the simpler **Histogram** with no improvement and it tests each $n \in \mathbb{N}$, odd and even.

5.2 Hardware Specifications Used for Computational Verification

The computational verification was performed on a personal computer with the following hardware specifications:

- **Processor:** Pentium(R) Dual-Core E5300 CPU 2.60GHz (2 cores)
- **RAM:** 2.5 GB DDR2

- **Storage:** 512 GB HDD
- **Operating System:** Windows 7 Integral 64-bit
- **Compiler:** Borland Delphi 7

To ensure that the program can handle very large numbers generated during the iterations of the generalized Collatz sequences, I chose the `Int64` data type. It allows for values up to $2^{63} - 1$, which provides a wide enough range to test billions of $n \in \mathbb{N}$ and accommodate exponential growth without immediate risk of overflow.

The tests for $k = 1$ took just **18 min**.
 The tests for $k = 2$ took **10 hours**.
 The tests for $k = 3$ took **10 hours**.
 The tests for $k = 4$ took **18 hours**.
 The tests for $k = 5$ took **11 hours**.
 The tests for $k = 6$ took **9 hours**.
 The tests for $k = 7, k = 8, k = 9$ took **4 hours**.
 The tests for $k = 10$ took **16 hours**.
 The tests for $k = 11$ took **12 hours**.
 The tests for $k = 12$ took **10 hours**.
 The tests for $k = 13$ took **18 hours**.
 The tests for $k = 14$ took **12 hours**.
 The tests for $k = 15$ took **10 hours**.
 The tests for $k = 16$ took **10 hours**.
 The tests for $k = 17$ took **8 hours**.
 The tests for $k = 18$ took **8 hours**.
 The tests for $k = 19$ took **10 hours**.
 The tests for $k = 20$ took **9 hours**.

5.3 Data extracted by each program

The data extracted by **GC Tester** program:

- **count:** the number of $n \in \mathbb{N}^*$ tested.
- **count1:** the number of $n \in \mathbb{N}^*$ validated because they converged to 1.
- **jt:** the total number of iterations via function g_k recorded by all the tested numbers in the whole large interval.

- **mt**: the total number of iterations via function f_k recorded by all the tested numbers in the whole large interval.
- **jmt**: the total number of iterations via functions g_k and f_k recorded by all the tested numbers in the whole large interval.
- **jmax**: the highest number of g_k iterations recorded by a number in the large interval.
- **mmax**: the highest number of f_k iterations recorded by a number in the large interval.
- **mjmax**: the highest number of g_k and f_k iterations recorded by a number in the large interval.
- **njmax**: the number that reached the highest number of g_k iterations in the large interval.
- **nmmax**: the number that reached the highest number of f_k iterations in the large interval.
- **nmjmax**: the number that reached the highest number of g_k and f_k iterations in the large interval.
- **Jav**: the average number of g_k iterations over the large interval.
- **mav**: the average number of f_k iterations over the large interval.
- **jmax**: the average number of g_k and f_k iterations over the large interval.
- **jti**: the total number of iterations via function g_k recorded by all the tested numbers in one of the 32 small intervals.
- **mti**: the total number of iterations via function f_k recorded by all the tested numbers in one of the 32 small intervals.
- **jmti**: the total number of iterations via functions g_k and f_k recorded by all the tested numbers in one of the 32 small intervals.
- **jmaxi**: the highest number of g_k iterations recorded by a number in one of the 32 small intervals.
- **mmaxi**: the highest number of f_k iterations recorded by a number in one of the 32 small intervals.

- **njmaxi**: the highest number of g_k and f_k iterations recorded by a number in one of the 32 small intervals.
- **njmaxi**: the number that reached the highest number of g_k iterations in one of the 32 small intervals.
- **nmmaxi**: the number that reached the highest number of f_k iterations in one of the 32 small intervals.
- **nmjmaxi**: the number that reached the highest number of g_k and f_k iterations in one of the 32 small intervals.
- **beg**: the starting value of one of the 32 small intervals.
- **Javi**: the average number of g_k iterations over one of the 32 small intervals.
- **mavi**: the average number of f_k iterations over one of the 32 small intervals.
- **jmavi**: the average number of g_k and f_k iterations over one of the 32 small intervals.

The data extracted by **GC Tester2** program:

- **Beg**: the start of each of the 32 small intervals.
- **Count**: the total number of even and odd numbers in $\mathbb{N}^*$ tested in each of the 32 small intervals and in the large interval.
- **count1**: the total number of even and odd numbers in $\mathbb{N}^*$ tested that have converged to **1** in each of the 32 small intervals and in the large interval.

The data extracted by **Non Trivial Cycles** program:

Each time it has tested 100 numbers, it will display the k , the number of cycles found, and each following line contains the number of elements in the cycle followed by all the elements of the cycle. The program runs continuously, but if you see that no new cycles are being found, you can choose to copy the result and paste it into the graphical interface of the **GC Tester2** program.

The data extracted by **Histogram** Program:

Figure 1: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 1$ and $n \in [1, 100]$ (top).

Figure 2: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 1$ and $n \in [501, 600]$ (middle).

Figure 3: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 1$ and $n \in [901, 1000]$ (bottom).

Figure 4: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 5$ and $n \in [1, 100]$ (top).

Figure 5: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 5$ and $n \in [501, 600]$ (middle).

Figure 6: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 5$ and $n \in [901, 1000]$ (bottom).

Figure 7: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 10$ and $n \in [1, 100]$ (top).

Figure 8: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 10$ and $n \in [501, 600]$ (middle).

Figure 9: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 10$ and $n \in [901, 1000]$ (bottom).

Figure 10: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 15$ and $n \in [1, 100]$ (top).

Figure 11: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 15$ and $n \in [501, 600]$ (middle).

Figure 12: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 15$ and $n \in [901, 1000]$ (bottom).

Figure 13: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 20$ and $n \in [1, 100]$ (top).

Figure 14: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 20$ and $n \in [501, 600]$ (middle).

Figure 15: Number of f_k iterations in red and number of g_k iterations in blue, for $k = 20$ and $n \in [901, 1000]$ (bottom).

5.4 Computational Results Report

- All the generalized Collatz sequences for $k \in \{1, 2, \dots, 20\}$ except for $k = 2$ and $k = 4$ over the moderate ranges converges to 1, and the program **GC Tester** have not detected any non-trivial cycle since there were no sequences with 140 millions iterations without reaching a previously tested smaller value.

- For $k = 2$, the only Non-Trivial cycle that has an element less than 1,073,741,824 is:
 $\{23, 116, 29, 148, 37, 188, 47, 236, 59, 296, 74, 372, 93, 468, 117, 588, 147, 736, 184, 46, 232, 58, 292, 73, 368, 92\}$
- For $k = 2$, out of the first 1,073,741,823 natural numbers tested, 145,959,493 produced a sequence that converges to 1, which represents 13.59% of the total, and all the others converged to the no-trivial cycle.
- For $k = 2$, for the initial terms equal to 456916631, 571145789, 713932237, 892415297, we obtain the negative terms. These negative terms can only be explained by an overflow. However, we checked the convergence starting from these terms using the **Windows 7** calculator, and after several iterations we obtained the terms, which are less than the initial terms. Using the Early Convergence Criterion, we conclude that these terms converge to **1** or an element of a non-trivial cycle.
- For $k = 4$ the only Non-Trivial cycle that have an element less than 1,073,741,824 is:
 $\{178, 3040, 190, 3232, 202, 3440, 215, 3664, 229, 3904, 244, 4160, 260, 4432, 277, 4720, 295, 5024, 314, 5344, 334, 5680, 355, 6048, 378, 6432, 402, 6848, 428, 7280, 455, 7744, 484, 8240, 515, 8768, 548, 9328, 583, 9920, 620, 10544, 659, 11216, 701, 11920, 745, 12672, 792, 13472, 842, 14320, 895, 15216, 951, 16176, 1011, 17200, 1075, 18288, 1143, 19440, 1215, 20656, 1291, 21952, 1372, 23328, 1458, 24800, 1550, 26352, 1647, 28000, 1750, 29760, 1860, 31632, 1977, 33616, 2101, 35728, 2233, 37968, 2373, 40352, 2522, 42880, 2680, 45568, 2848\}$
- For $k = 4$
 Out of the first 1,073,741,823 natural numbers tested, 971,669,489 produced a sequence that converges to 1, which represents 90.49% of the total, and all the others converged to the no-Trivial cycle.

All data generated in the computational verification are available in a supplementary dataset.

The results are fully reproducible using the source code provided by the author.

5.5 Conclusion of the Computational Verification

For k ranging from 1 to 20, except for $k = 2$ and $k = 4$ our computations did not reveal any nontrivial cycles, and all tested trajectories reached 1.

6 Generalized Collatz sequences $(1+2^k)n+S_k(n), n/2^k$

We first defined the two auxiliary functions:

$$g_k : \mathbb{E} \rightarrow \mathbb{N}^*$$

$$g_k(n) = \frac{n}{2^k} \quad \text{where } \mathbb{E} \text{ is the set } n \in \mathbb{N}^* \text{ that are} \\ \textbf{congruent to zero modulo } 2^k$$

$$f_k : \mathbb{L} \rightarrow \mathbb{E}$$

$$f_k(n) = (1 + 2^k)n + S_k(n) \quad \text{where } S_k(n) = 2^k - (n \bmod 2^k) \\ \text{and } \mathbb{L} \text{ is the set of } n \in \mathbb{N}^* \text{ that are not} \\ \textbf{congruent to zero modulo } 2^k$$

k is called the **Exponent parameter**.

We have:

$$\mathbb{L} \cup \mathbb{E} = \mathbb{N}^*$$

and

$$\mathbb{N}^* \cup \mathbb{E} = \mathbb{N}^*$$

so:

The generalized Collatz sequence is then defined as follows:

$$C_k : \mathbb{N}^* \rightarrow \mathbb{N}^*$$

$$C_{n+1} = \begin{cases} f_k(C_n) & \text{if } C_n \not\equiv 0 \pmod{2^k} \\ g_k(C_n) & \text{if } C_n \equiv 0 \pmod{2^k} \end{cases}$$

7 Conclusion

In this work, we introduced a methodological generalization of the Collatz sequence using the transformation $(1 + 2^k)n + S_k(n), n/2^k$, where $S_k(n)$ is the smallest non-negative additive inverse of $\mathbf{n}$ modulo 2^k ensuring that the result is divisible by 2^k . This generalization was designed to preserve key dynamical features of the original $3n + 1$ sequence, including the appearance of generalized singularities and the existence of a trivial cycle. Importantly, this work emphasizes the role of the singularity revealed in our previous preprint [1], by extending its structural logic to the generalized sequences. The transformation was carefully constructed so that the generalized singularities characterized by specific binary patterns remain central to the convergence dynamics across all tested values of k . It is worth noting that the generalized transformation considered here is related, for the particular choice $d = 2^k$, to the generalized Collatz mappings introduced by Carnielli [4]. The present construction, however, was developed from a structural reverse-engineering of the classical Collatz sequence, starting from the binary singularity and extending its underlying structural logic to a parameterized family. A comprehensive computational verification was conducted for k ranging from 1 to 20. The results confirm that, for all $k \neq 2, k \neq 4$, every tested sequence eventually converges to 1, without encountering any non-trivial cycles within the tested ranges. In the special cases of $k = 2$ and $k = 4$, a small number of non-trivial cycles were detected, suggesting a fundamental difference in behavior for this specific value.

These results are purely empirical and are restricted to the finite computational domain explored in this study. They are presented solely to illustrate the behavior of the associated natural number sequences and to motivate further theoretical investigation of the underlying structure of this family of generalized Collatz-type iterations.

References

- [1] Ammar Hamdous, *Revealing a Singularity in Collatz Sequences*, Zenodo, 2025.
- [2] Naouel Boulkaboul, *$3n + 3^k$: New Perspective on Collatz Conjecture*, arXiv, 2022.
- [3] Terence Tao, *Almost all orbits of the Collatz map attain almost bounded values*, arXiv, 2019.

- [4] W. Carnielli, *Some Natural Generalizations of the Collatz Problem*, Applied Mathematics E-Notes, 15 (2015), 207–215.

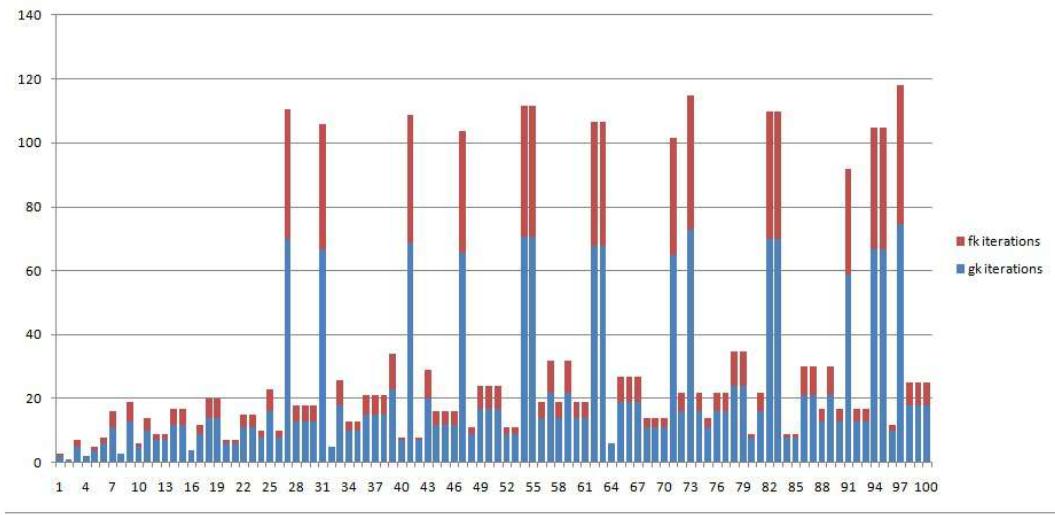


Figure 1: Number of f_k iterations in red and number of g_k iterations in blue For $k = 1$, for $n \in [1, 100]$ (top). 19

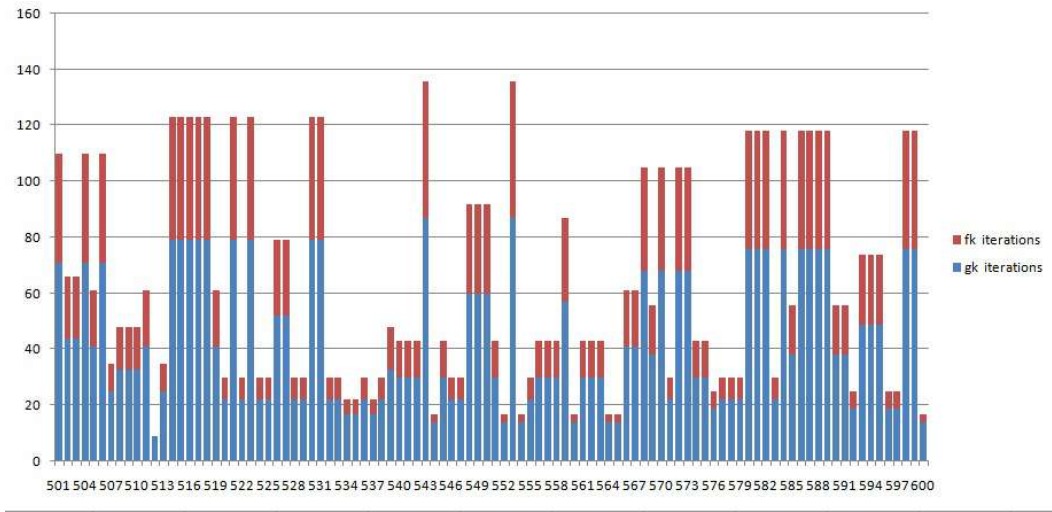


Figure 2: Number of f_k iterations in red and number of g_k iterations in blue
For $k = 1, n \in [501, 600]$ (middle). 20

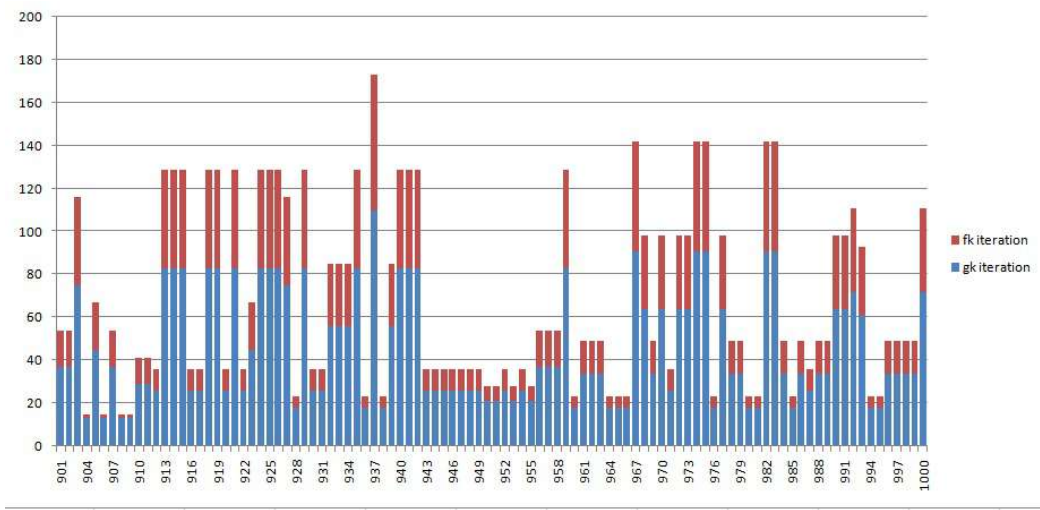


Figure 3: Number of f_k iterations in red and number of g_k iterations in blue For $k = 1$, and $n \in [901, 1000]$ (bottom).

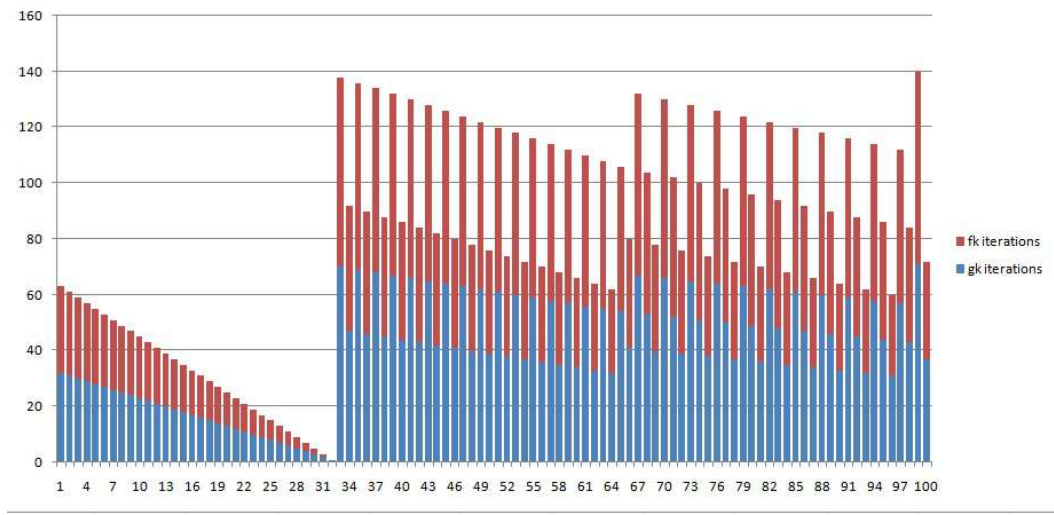


Figure 4: Number of f_k iterations in red and number of g_k iterations in blue
For $k = 5$, for $n \in [1, 100]$ (top). 22

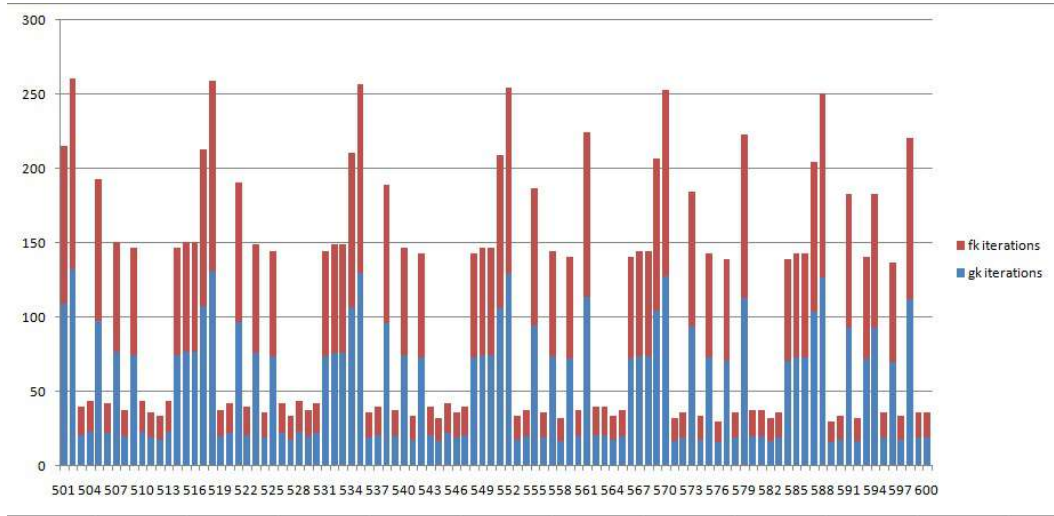


Figure 5: Number of f_k iterations in red and number of g_k iterations in blue
For $k = 5$, $n \in [501, 600]$ (middle). 23

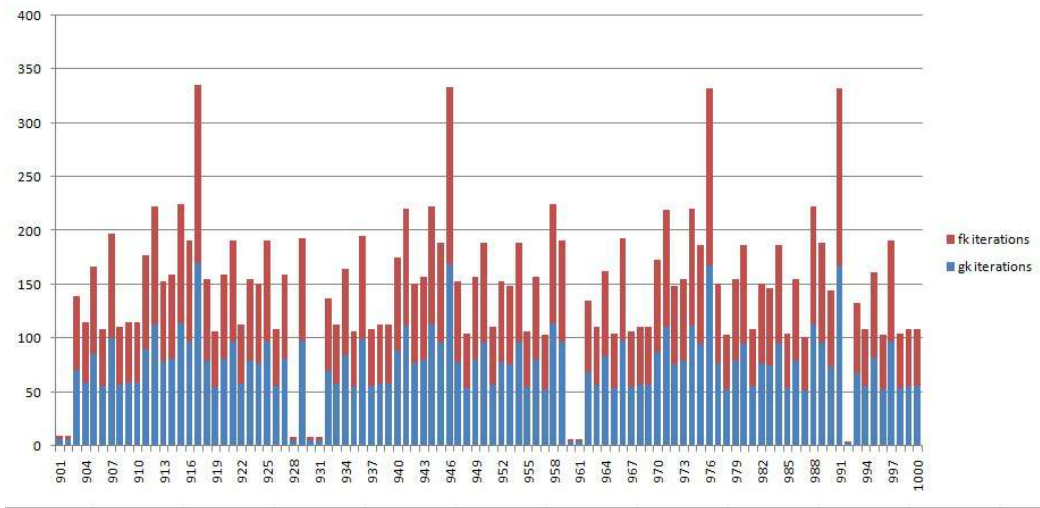


Figure 6: Number of f_k iterations in red and number of g_k iterations in blue For $k = 5$, and $n \in [901, 1000]$ (bottom).

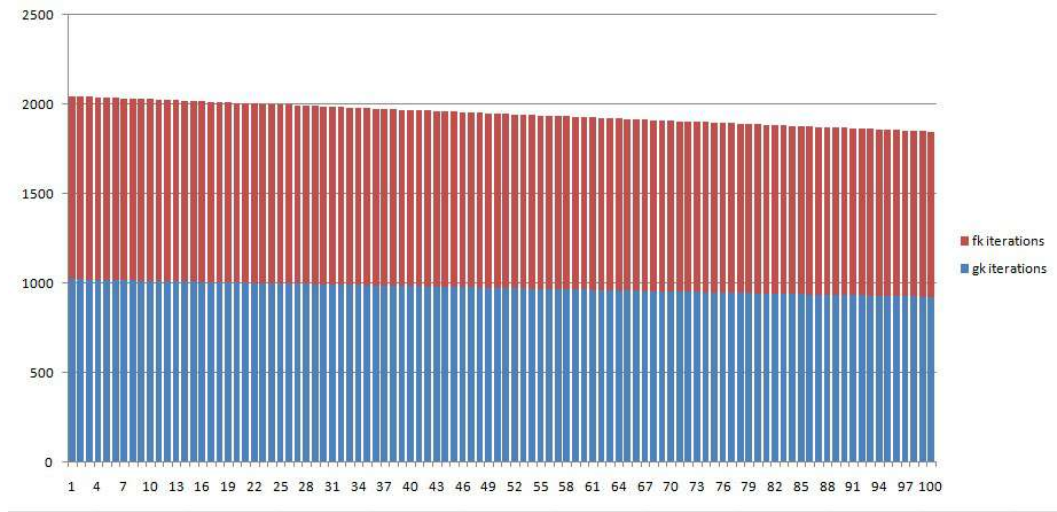


Figure 7: Number of f_k iterations in red and number of g_k iterations in blue
 For $k = 10$, for $n \in [1, 100]$ (top). 25

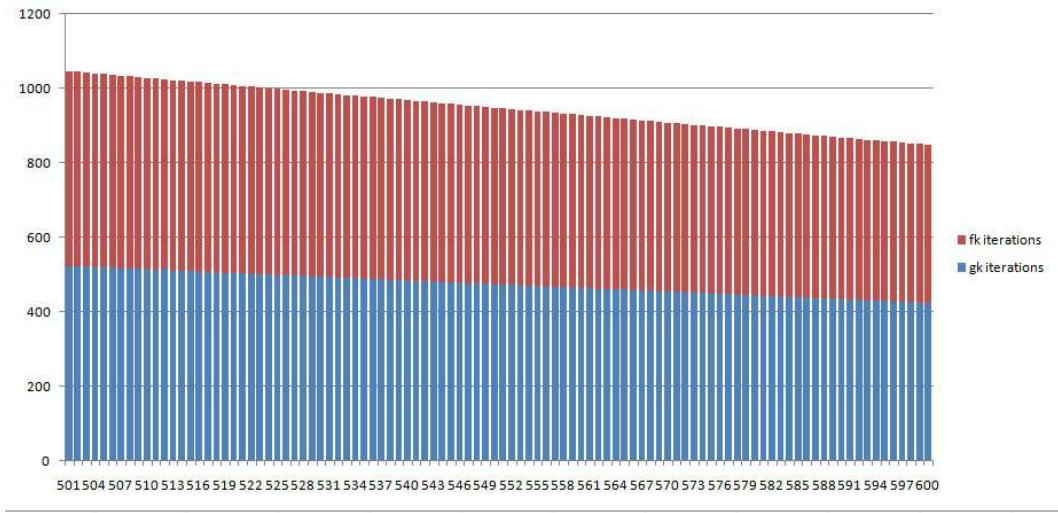


Figure 8: Number of f_k iterations in red and number of g_k iterations in blue For $k = 10$, $n \in [501, 600]$ (middle). 26

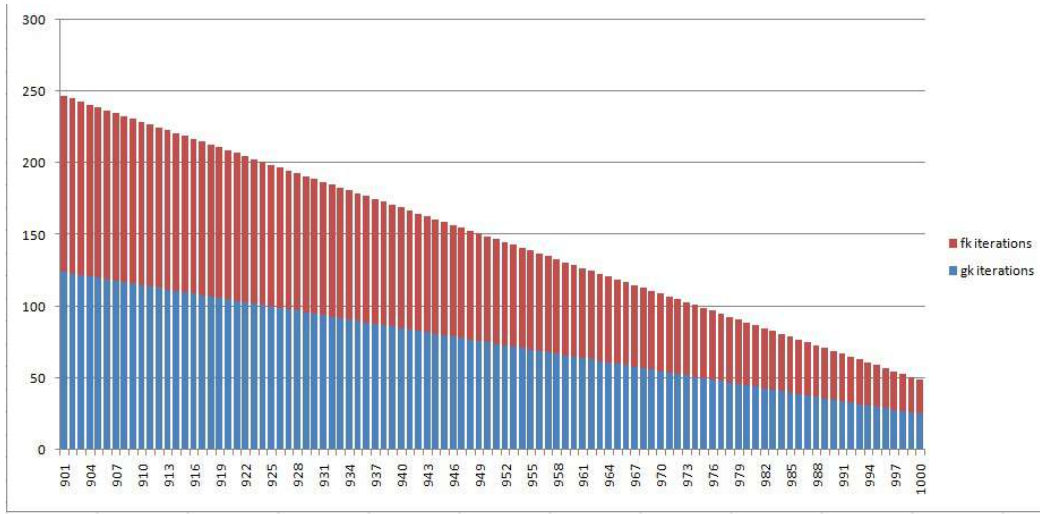


Figure 9: Number of f_k iterations in red and number of g_k iterations in blue For $k = 10$, and $n \in [901, 1000]$ (bottom).

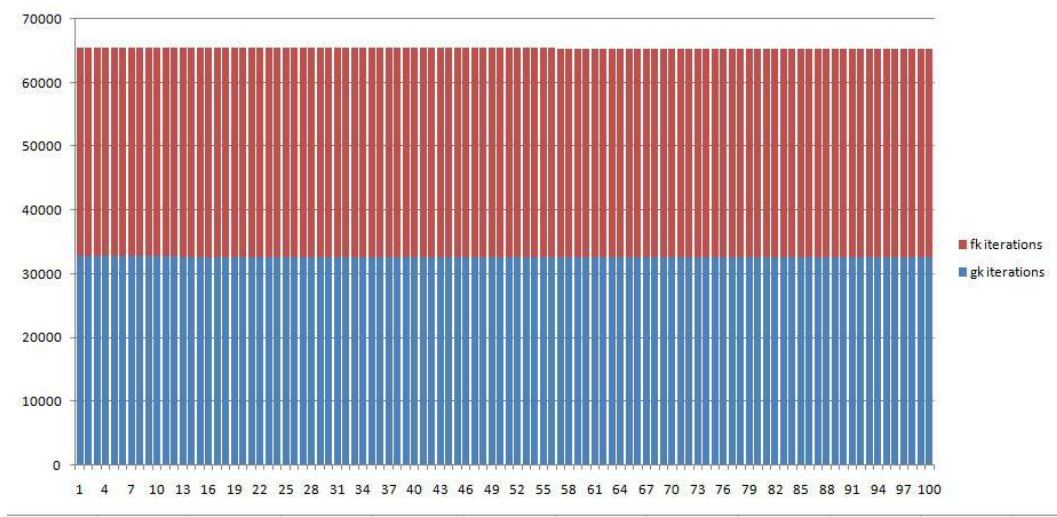


Figure 10: Number of f_k iterations in red and number of g_k iterations in blue For $k = 15$, for $n \in [1, 100]$ (top). 28

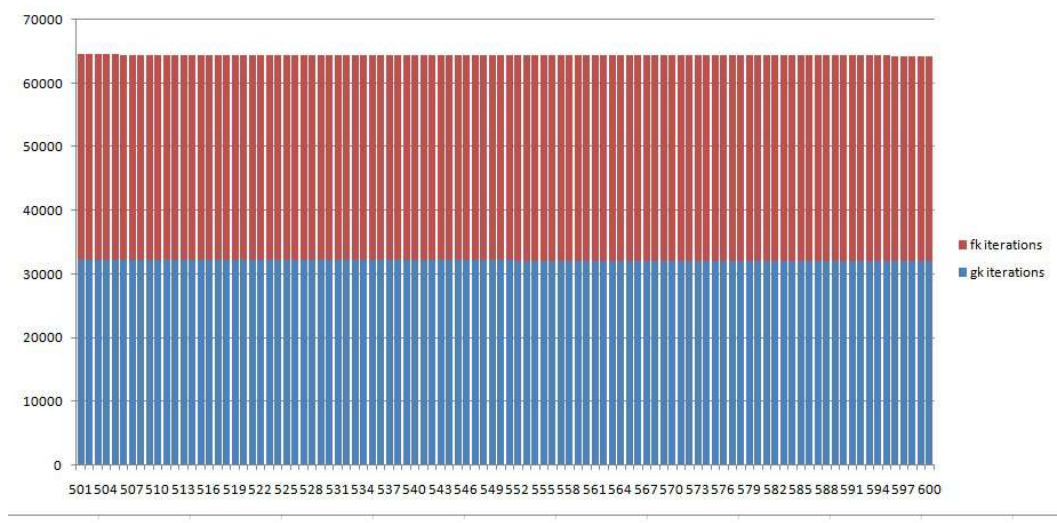


Figure 11: Number of f_k iterations in red and number of g_k iterations in blue
For $k = 15$, $n \in [501, 600]$ (middle). 29

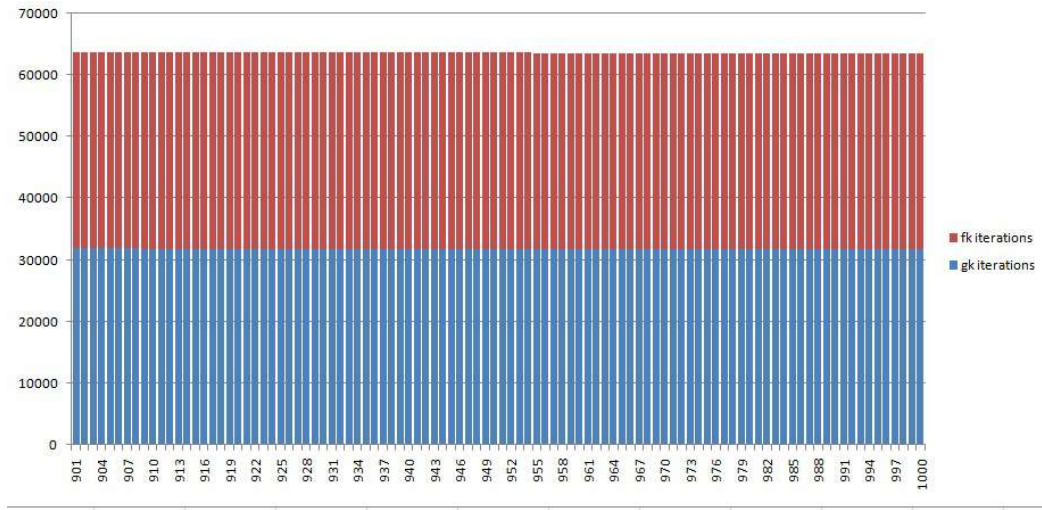


Figure 12: Number of f_k iterations in red and number of g_k iterations in blue For $k = 15$, and $n \in [901, 1000]$ (bottom).

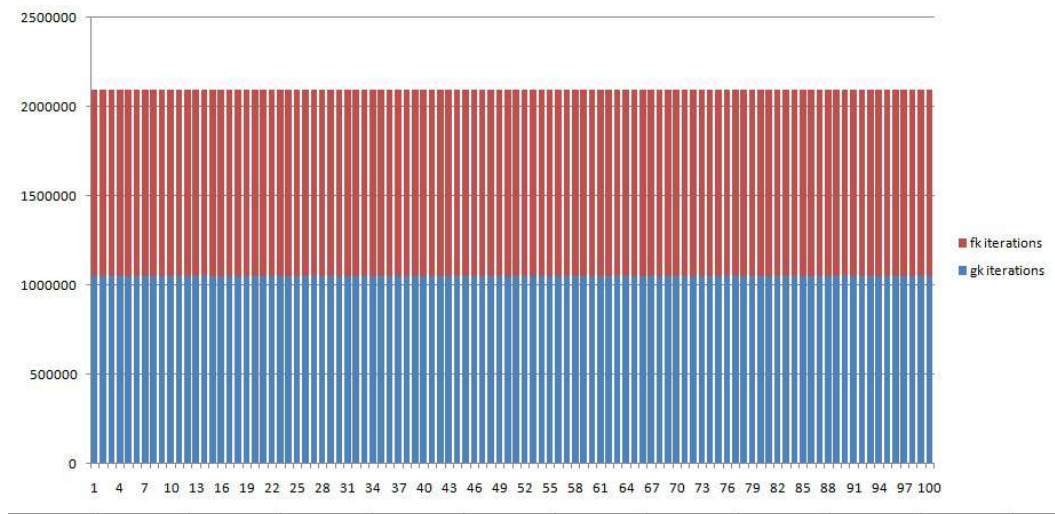


Figure 13: Number of f_k iterations in red and number of g_k iterations in blue
For $k = 20$, for $n \in [1, 100]$ (top). 31

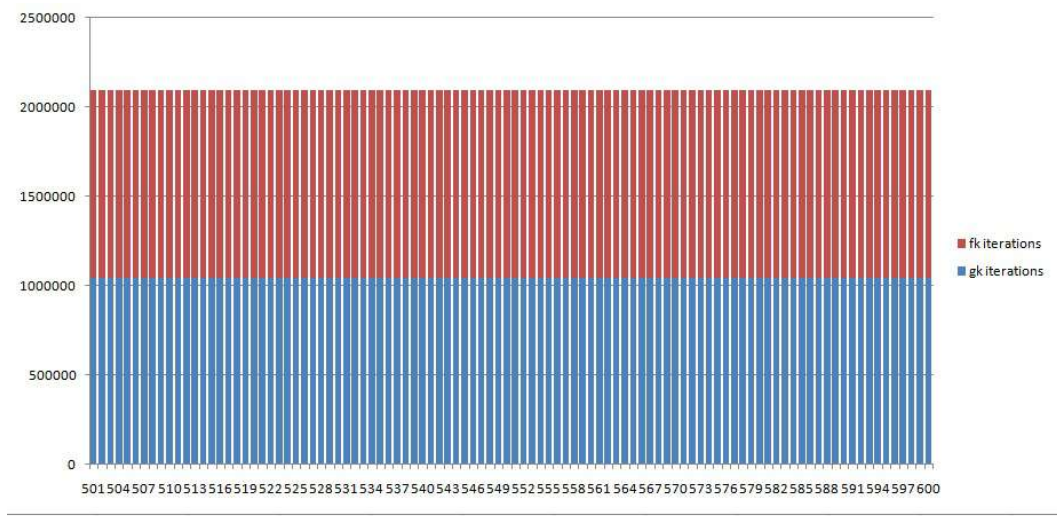


Figure 14: Number of f_k iterations in red and number of g_k iterations in blue
For $k = 20$, $n \in [501, 600]$ (middle). 32

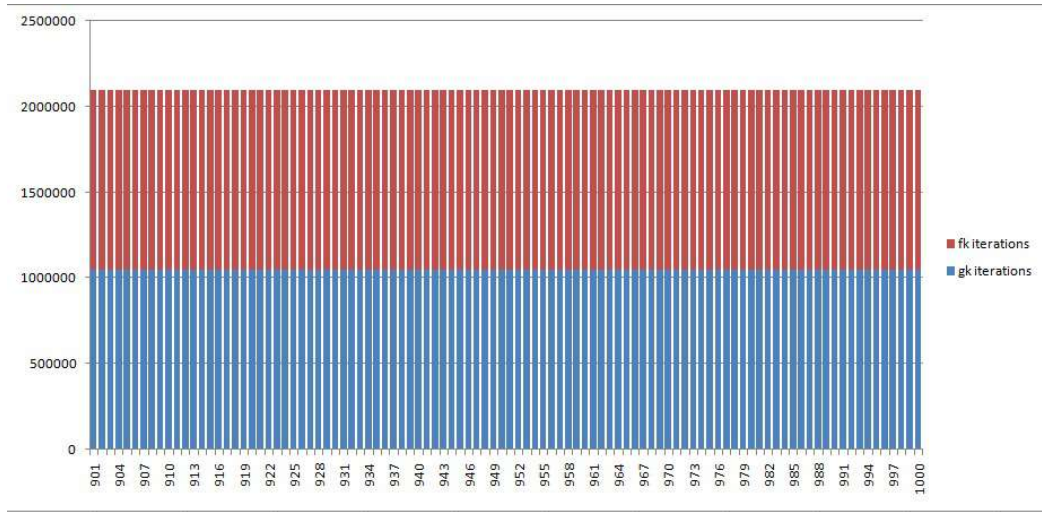


Figure 15: Number of f_k iterations in red and number of g_k iterations in blue For $k = 20$, and $n \in [901, 1000]$ (bottom).