

# From The Collatz Function, Selected Arithmetic Sequences (mod $m$ ) Are Sorted To Form Array Structures That Illustrate Tiling Patterns

2020 Mathematics Subject Classification: Primary 11B25; Secondary 11B50, 52C20, 00A27

Keywords: Number Theory, Arithmetic Progressions

Sequences (mod  $m$ ), Tiling in 2 dimensions (tessellations), Open Problems

Ihor Jakowec, [ask@ijakowec.info](mailto:ask@ijakowec.info)

Thursday 13<sup>th</sup> November, 2025 00:04 GMT

Creative Commons License CC-BY-SA-4.0



## Abstract

Usual approaches to this problem involve exhaustive algorithmic, or probabilistic, methods. Moreover, when set as a universally quantified problem, the Collatz conjecture is classified as undecidable. In the arithmetical hierarchy, it is  $\Pi_2^0$ -complete. This situation motivated the development of a different approach in investigating the conjecture by using the conjecture's results as data. The data is assembled as primitive data structures, which in turn are reassembled to form more composite data structures that create patterns that gain further insight about the Collatz conjecture. These data structures are outlined as follows:

Positive integers are viewed in their binary form. The definition of parity is extended by introducing a concept called multiple parity (abbreviated as  $m$  parity). The variable  $m$  refers to the 2-adic valuation of a positive integer. The *three plus one* operation is shown to be able to generate all possible  $m$  parities.

The *ruler sequence* (OEIS entry: A001511) is extended to sequences with differing minima (called *ruler harmonic sequences*). Next,  $m$  parities, of integer arithmetic progressions (with varying common differences and origins), are classified as either ruler harmonic sequences of various orders, or constant sequences.

Borrowing from earlier nomenclature: A *hailstone function* inputs a positive integer and returns its triple plus 1 (from an odd input), or returns half (from an even input). The initial iteration is called the *seed*. Later iterations are called *moves*. More than one iteration is called a *path*. Outputs are called *heights*. All are used to create these composite structures:

Two-dimensional arrays of paths, called *cascades*, are assembled to hold data in manifestations of: heights, offsets (height differences of a given move between different paths),  $m$  parities, and ones-phobic binaries (a bit represents a rise or fall in the height between successive moves). Paths, of a cascade, all have the same length. Flutes are a power of two subdivision of a cascade column, whose exponent is set by its move number ( $2^0$  for the seed column,  $2^1$  for the column of the first move, etc.). At the same relative flute position (from flutes of successive moves), paths are selected to create a tier. A continuous selection of columns from a tier is called a region. Regions are classified as either: coherent, ruler harmonic, or dissonant. A coherent region's column has identical  $m$  parities (and offsets). In the ruler harmonic region,  $m$  parities follow a ruler harmonic distribution. The dissonant region has no simple patterns of  $m$  parities. However, sequences cycle, but at longer cycles for successive columns.

In a single cascade, a row-wise sort of  $m$  parities, per column, forms *tiers* called *quilts*. Except for offset magnitudes, tiers within a cascade, have the same properties as tiers among successive cascades. The structure of a cascade's quilts leads to establish that all possible ones-phobic binaries occur, within the confines of the cascade's dimensions ( $2^n$  rows by  $n$  columns).

Key Takeaway: By the properties of a ruler harmonic sequence, we show that a sorted and collated cascade forms a quilt that places limitations on the magnitude and arrangement of  $m$  parities.

# Contents

<b>1</b>	<b>Nomenclature Convention For: Arithmetic Progression and Binary Sequence</b>	<b>1</b>
<b>2</b>	<b>Introduction</b>	<b>1</b>
2.1	Defining The Collatz Conjecture . . . . .	1
<b>3</b>	<b>Extending The Definition Of Parity</b>	<b>2</b>
3.1	How <i>TP1</i> Sets Multiple Parity . . . . .	2
<b>4</b>	<b>The <math>m</math> Parity Distribution Pattern From Positive Integers</b>	<b>4</b>
4.1	Frequency of a Given $m$ Parity in Consecutive Integers (Binary, Bit Length $n$ ) . . . . .	4
4.2	Ruler Harmonics . . . . .	4
4.2.1	Deriving Ruler Harmonics in Positive Integer Arithmetic Progressions. . . . .	5
<b>5</b>	<b>Classifying Sequences With Respect to Their <math>m</math> Parity Distribution Type</b>	<b>5</b>
5.1	The Frequency Distribution of $m$ Parities in an $m$ Parity Sequence	6
5.2	Sequential Ordering of $m$ Parities in an $m$ Parity Sequence . . . .	6
5.3	Requirements Needed to Determine if a Sequence is a Ruler Harmonic Sequence . . . . .	7
5.4	Procedure to Determine if a Sequence is a Ruler Harmonic Sequence	7
<b>6</b>	<b>Fundamental Property Of <math>m</math> Parities</b>	<b>8</b>
<b>7</b>	<b>Fundamental Properties of <math>m</math> Parity Sequences</b>	<b>9</b>
7.1	Effect of Varying Origin and Varying Jump Increment, on $rhs^k$ . .	9
7.1.1	Multiplying by Two to the Power of $n$ . . . . .	9
7.1.2	Changing the Origin in a Ruler Harmonic Sequence of Order 0	9
7.1.3	Changing the Origin in a Ruler Harmonic Sequence of Order $k$	9
7.1.4	$m$ Parity Clamp . . . . .	10
7.1.5	No Clamping . . . . .	10
7.2	Multiplying a Sequence By an Odd Positive Integer . . . . .	11
7.3	An Index Advance by $3^k$ . . . . .	11
7.4	An Index Advance by $6^k$ . . . . .	12

7.4.1	How a Sequence Transitions From Being Clamped to Being Un-clamped . . . . .	13
7.4.2	Sequence Transition From Clamped to Un-Clamped (Illustrated in Binary) . . . . .	13
<b>8</b>	<b>Cascades, Seed Ranges, Flutes, Offsets, and Tessellation</b>	<b>14</b>
8.1	The Cascade . . . . .	14
8.2	Seed Range . . . . .	14
8.3	Flute . . . . .	14
8.4	Offsets . . . . .	14
<b>9</b>	<b>Placements and Landings</b>	<b>15</b>
9.1	Using Modulo Arithmetic to Generate Ruler Harmonic Sequences . . . . .	15
9.5	Illustrating Inter-Flute Tessellation In a Cascade . . . . .	18
9.6	Why Cascade Flutes Tessellate Column-wise (With Respect to Their Constituent $m$ Parities). . . . .	18
9.6.1	Generating Arithmetic Progressions For the Column of the Next Move . . . . .	20
9.7	Generalized Evaluation of Offsets . . . . .	25
9.7.1	Generating Offsets for Flutes at the Column of the First Move . . . . .	25
9.7.2	Generating Offsets for Flutes for the Column of the Second Move . . . . .	25
9.7.3	Repeated Compositions (Generating an Offset for a Flute of Size $2^n$ ) . . . . .	26
9.7.4	Introducing the Ones-Phobic Binary . . . . .	27
9.8	Relating an Offset Cascade to a Ones-Phobic Cascade . . . . .	28
9.9	Offset Pair With the Same Exponent . . . . .	29
9.10	Offset Pair With Different Exponents . . . . .	29
<b>10</b>	<b>Frequency Effects on Ones-Phobic Binaries</b>	<b>29</b>
10.1	The Effect of Prepending a Zero . . . . .	30
10.2	The Effect of Prepending a One . . . . .	30
10.3	Starting With the Least Significant Bit . . . . .	30

<b>11 Demonstrating the Existence of a Tier</b>	<b>38</b>
11.1 Building a Tier . . . . .	38
11.1.1 Why Uniform Range Spacing, Within a Tier? . . . . .	39
11.2 Properties of Paths in a Tier . . . . .	42
11.2.1 A Reason for Move Synchronization, in a Tier . . . . .	42
11.3 More Properties of Ruler Harmonic Sequences . . . . .	43
<b>12 Regions: Coherent, Ruler Harmonic, and Dissonant</b>	<b>44</b>
<b>13 The Tiara</b>	<b>48</b>
13.1 Extra Structures Arising From (Formed by), Tiers . . . . .	48
13.2 Extending The Ruler Harmonic Sequence Definition to Include Tiaras	48
13.2.1 Tiara Points of Equal Length Create a Tier . . . . .	48
13.2.2 Illustrating What Follows Tiara Points . . . . .	49
13.2.3 Extending a Tier by One Bean . . . . .	49
13.2.4 Building a Ruler Harmonic Sequence From Beans . . . . .	49
<b>14 The Quilt</b>	<b>52</b>
14.1 Components of the Quiltwork Structure . . . . .	52
14.2 Quilt Formation (By Sorting an $m$ Parity Cascade) . . . . .	53
14.2.1 Stacks (of Quilt Rectangles) . . . . .	53
14.3 The Other Offset: Tier Offsets . . . . .	54
14.4 How a Quilt Rectangle Leads To a Next Generation Stack . . . . .	55
14.5 A Quilt Margin and Fill Sufficiency . . . . .	56
<b>15 Enumeration of a Mosaic</b>	<b>61</b>
15.1 Sorting the Ruler Harmonic Region to Form a Powers of Two Staircase	62
<b>16 Bringing Together Remarks</b>	<b>63</b>
16.1 Longer and Longer Paths . . . . .	63
16.1.1 Exception: Singular Paths . . . . .	64
16.1.2 Conclusion . . . . .	64
<b>A How and Why Fibonacci Numbers Are Used as a Basis in a Numeral System</b>	<b>66</b>

<b>B Succession in One's Phobic Binaries</b>	<b>66</b>
<b>C Why there is a Fibonacci Staircase in A Mosaic</b>	<b>70</b>
C.1 Building an Enumerative Grove . . . . .	70
C.2 Properties of an Enumerative Grove . . . . .	71
C.2.1 Enumerative Groves are self similar . . . . .	71
C.2.2 Fibonacci Property Associated With An Enumerative Grove	72
C.3 Building the Next Larger Enumerative Grove . . . . .	72
<b>D A Bounding Estimate <i>Using Pseudo Gravity</i></b>	<b>77</b>
D.1 Modelling an Imaginary Solar System . . . . .	77
D.2 The Effect of Rises and Falls on The Balance . . . . .	77
D.2.1 A Transform to Encode Balance . . . . .	77
D.2.2 Tethering Rises to Falls In Ones-Phobic Binaries . . . . .	77
D.2.3 Balance Encoding To Find A Sink Location . . . . .	78
D.2.4 A Shorter Method to Locate a Sink in a Path . . . . .	79
D.3 Falls and Rises and Ratios . . . . .	80
D.4 Modelling A Pseudo Solar System (Analogy) . . . . .	81
D.5 Longer Paths Approach a Ruler Harmonic Frequency Distribution	81
<b>E The Binary Approach</b>	<b>86</b>
E.1 Establishing Ruler Harmonics for Offsets of Six . . . . .	86
E.2 Distribution of <i>L parity</i> in positive integers. . . . .	88
E.3 Relating L parity to Offsets Whose base is Six . . . . .	88
E.3.1 Frequency of <i>L parity</i> for a given m in Bn . . . . .	88
E.3.2 Leading and Trailing Parity . . . . .	88
<b>F Addendum of Observations</b>	<b>92</b>
F.1 Tabulating Results: Multiplying Index by $2^k$ With Varying Origin	92
F.1.1 Increment Advance by 2 . . . . .	92
F.1.2 Increment Advance by 4 . . . . .	92
F.1.3 Increment Advance by 8 . . . . .	93
F.2 The m Parities of an Index Multiplied by a Power of Three . . . .	94
F.3 Aside: Illustrating the Effect of Sequential Multiplication of a Constant and/or Varying the Origin, on Sequences, by Using Tables	94

<b>G Two Algorithms To Evaluate <math>m</math> Parity</b>	<b>98</b>
G.1 Description of The Source Code . . . . .	98
G.1.1 How the mp_bitwise Function Yields $m$ Parity. . . . .	98
G.1.2 How the mp_modular Function Yields $m$ parity. . . . .	98
<b>Bibliography</b>	<b>106</b>

## List of Tables

1	Arithmetic Progression and $m$ Parity Notations Alongside Their Expansions . . . . .	22
2	Arithmetic Progressions With Their $m$ Parity Sequences Within a Flute For a Given Move . . . . .	23
3	CS Frequency Distribution of Each Order At a Given Move From a Cascade . . . . .	24
4	RHS Frequency Distribution of Each Order At a Given Move From a Cascade . . . . .	24
5	Tier Path Heights . . . . .	46
6	Corresponding m-parities of Tier Paths . . . . .	46
7	Corresponding Cascade Offsets of Tier Paths . . . . .	46
8	Ones-Phobic Encoding of Tier Paths . . . . .	47
9	Tier Path Heights Followed By Tier Offsets . . . . .	47
10	Seed Values That Generate a Path Containing Both: a Height Maxima and a Path Length Maxima . . . . .	80
11	Pseudo Escape Velocities . . . . .	83
12	Varying the Origin for Index Common Difference of Six. . . . .	96
13	Varying the Origin for Index Common Difference of 36 . . . . .	97

## List of Figures

1	Obtaining the First Few Multiple Parities Using $TP1$ . . . . .	2
2	Why an odd input to the $TP1$ operator always leads to an even result. . . . .	3
3	$TP1$ Inputs That Form Even $m$ parities . . . . .	3
3	$TP1$ Inputs That Form Odd $m$ parities . . . . .	3
4	How any magnitude of even or odd $m$ parity can be Obtained From $TP1$ . . . . .	3

5	Generalization (in Binary): Adding an Origin to an Index . . . . .	10
6	Multiplication by and Odd Integer, Leaves $m$ parity Unchanged. (Shown as Repeating an Odd number of Additions of a Binary to Itself) . . . . .	11
7	Ruler Harmonics The First Few Powers Of Six . . . . .	32
8	Comparison of Heights With Offset Encoding . . . . .	33
9	Quilt Pattern in Offset, and Binary Encoded, Cascade . . . . .	34
10	A Mosaic (All Unique Ones-Phobic Binary Paths From a Cascade)	35
11	Cascade of Heights With it's Corresponding Cascade of Offsets and Cascade of $m$ Parities . . . . .	36
12	Sorted and Unsorted $m$ Parity Cascade, After a MultiStage Sort a Quilt Pattern is Revealed) . . . . .	37
13	Paths Forming a Tier Within and Outside a Cascade . . . . .	41
14	Synchronization Transitioning to Dissonance . . . . .	42
15	A Single Insertion Breaks Ruler Harmonics . . . . .	44
16	A Tiara (Highlighted Tier After Sequences) . . . . .	51
17	Quilt Margins . . . . .	60
18	Region Transition: From Coherent to Quilt to Dissonant. . . . .	62
19	Succession in a Ones-Phobic Numeral System . . . . .	68
20	Succession of Ones-Phobic Binaries in Nested Mosaics . . . . .	69
21	An Enumerative Grove of Magnitude 5 (Magnitude 6 With Red Vertices) . . . . .	74
22	An Enumerative Grove of Magnitude 6 (Magnitude 7 With Red Vertices) . . . . .	74
23	An Enumerative Grove of Magnitude 7 (Magnitude 8 With Red Vertices) . . . . .	75
24	An Enumerative Grove of Magnitude 7 (Magnitude 8 With Red Vertices) . . . . .	76
25	Using Balance to Find A Sink Location. . . . .	78
26	Multiples of 6 with their binary representation . . . . .	87
27	An Icicle Sector, Abridged From Figure 28. . . . .	89
28	Showing How Modulo Arithmetic Can Generate Ruler Harmonics .	90
29	Ruler Harmonics in Powers of Six With Landings and Placements .	91
30	Ruler harmonics, for $m$ parities and $L$ parities. . . . .	95

## Listings

1	An Algorithm Generating a Hailstone Path Where Rises and Falls are Counted for A Balance . . . . .	84
2	Table Generator Showing The Effect of Ratios for Pseudo Planets .	85
3	Algorithm to obtain m parity (bitwise approach) . . . . .	98
4	Algorithm to obtain m parity (modular arithmetic approach) . . .	98
5	Procedure to establish whether a subsequence is a ruler harmonic sequence of order zero. . . . .	99
6	CS and RHS Frequency Distributions of Each Order At a Given Move From a Cascade. . . . .	99

# 1 Nomenclature Convention For: Arithmetic Progression and Binary Sequence

Introduced for brevity and disambiguation:

- Let  $k \cdot \mathbb{N}_i$  denote an arithmetic progression of positive integers, with a common difference of  $k$ , and origin  $i$  (the  $k$  prefix is optional when  $k = 1$ ).
- Let  $\mathbb{B}_n$  be a binary string of bit length  $n$ .

## 2 Introduction

### 2.1 Defining The Collatz Conjecture

Some earlier jargon from the conjecture's investigation is borrowed here, namely: height, path, seed, move and hailstone. [Hay84] Let  $hs_x \in \mathbb{N}_1$  be called a *height*. A sequence of heights is called a *path*. The metaphor involves hailstone dynamics implements an analogy which relates changes in the varying oscillating motion of a hailstone to the mimicking output of the defining function of the Collatz conjecture. [Hay84]

Here, this function is called the hailstone function, its following iterative form is used to obtain the height of the next move in a path.

$$hs_{x+1} = \begin{cases} 3 \cdot hs_x + 1 & \text{if } 1 \equiv hs_x \pmod{2} \\ hs_x/2 & \text{if } 0 \equiv hs_x \pmod{2} \end{cases} \quad (1)$$

The conditional operation  $3 \cdot hs_x + 1$  is called *TP1*.

The conditional operation  $\frac{hs_x}{2}$  is called *DV2*.

The conjecture raised is: What ultimately happens to height(s) in a path as  $x$  of  $hs_x$  increases for a given seed value? Specifically: Do all paths for any given seed eventually attain a height of 1? Is the cycle of 4, 2, 1,  $\dots$  the only cycle found in paths?

The initial height, denoted as  $hs_0$ , is called the *seed* [Hay84]. In general an element of a path is called a *move*. A move has manifestations which can refer to: a height (the output of the hailstone function after a given iteration); an *m parity* (a short form for the 2-adic valuation of an integer); *offset* (the difference between heights from different paths but at the same move; a bit or glyph (which denotes a change in height, either a rise or a fall). The bit or glyph represents an element of a *ones phobic binary*. A ones-phobic binary is used to make visual patterns of a path's change in direction more immediately apparent (than would an integer) This is even evident with the appropriate glyph as seen in Figure 18.

This sans serif font is used to indicate explanatory annotation, or as the introduction of a concept or (sub)topic.

Next an extension to parity is introduced, called *m* parities. (Used here for a more concise working nomenclature for the 2-adic valuation of integers.) This is followed by the properties of *m* parities arising from arithmetic progressions with common differences as one variable and the origin as a second variable .

### 3 Extending The Definition Of Parity

**Definition 3.1** (*m* parity). Where *m* is number of consecutive trailing zeros from the least significant end of the binary representation of an integer in  $\mathbb{N}_1$ , which is called it's *m* parity (for multiple parity). Note: *0* parity is an integer with no trailing zeros, *1* parity is an integer with 1 trailing zero, etc.

Aside: *DV2* is not expounded upon here as is *TP1*. Since each time the *DV2* operator is applied to a positive even integer, it simply lowers that integer's *m* parity by 1.

#### 3.1 How *TP1* Sets Multiple Parity

Here we show how a any given number of trailing zeros (from the binary form of an integer), can be obtained; by selecting the appropriate input to the operator *TP1*.

Note:  $? \dots ?$  denotes a string of unspecified bits. (They are used as placeholders, whose contents do not pertain to the immediate discussion).

Below tables of Figure 2 illustrate the result, in binary, after applying *TP1* to an odd positive integer.

Figure 1: Obtaining the First Few Multiple Parities Using *TP1*

(a) 1 parity	(b) 2 parity	(c) 3 parity
$? \dots ?11$	$? \dots ?001$	$? \dots ?1101$
$? \dots ?11$	$? \dots ?001$	$? \dots ?1101$
$? \dots ?11$	$? \dots ?001$	$? \dots ?1101$
$\begin{array}{r} + \quad 1 \\ \hline ? \dots ?10 \end{array}$	$\begin{array}{r} + \quad 1 \\ \hline ? \dots ?100 \end{array}$	$\begin{array}{r} + \quad 1 \\ \hline ? \dots ?1000 \end{array}$
(d) 4 parity	(e) 5 parity	(f) 6 parity
$? \dots ?00101$	$? \dots ?110101$	$? \dots ?0010101$
$? \dots ?00101$	$? \dots ?110101$	$? \dots ?0010101$
$? \dots ?00101$	$? \dots ?110101$	$? \dots ?0010101$
$\begin{array}{r} + \quad 1 \\ \hline ? \dots ?10000 \end{array}$	$\begin{array}{r} + \quad 1 \\ \hline ? \dots ?100000 \end{array}$	$\begin{array}{r} + \quad 1 \\ \hline ? \dots ?1000000 \end{array}$

Ellipsis located *in between* specified bits denote repetition. The specified bits immediately preceding this ellipsis repeat. Any ellipsis immediately above (or below), another ellipsis repeats it's preceding bits the same number of times.

Figure 2: Why an odd input to the *TP1* operator always leads to an even result.

$$\begin{array}{r}
 m \quad ? \dots ? 1 \\
 2m \quad ? \dots ? 10 \\
 + \quad \quad 1 \\
 \hline
 3m + 1 \quad ? \dots ? 10
 \end{array}$$

Figure 3: *TP1* Inputs That Form Even *m* parities

$? \dots ? \mathbf{001}$  Using the top augends from Figures:  
 $? \dots ? \mathbf{001}01$  **1b**, **1d** and **1f**. The leading pair of specified  
 $? \dots ? \mathbf{001}0101$  bits is always **001**, and that 01 is appended to  
 $\vdots$  get the next entry. The bottom row illustrates  
 $? \dots ? \mathbf{001} \dots 01 \dots 01$  the generalization of *m* parity (even).

Figure 3: *TP1* Inputs That Form Odd *m* parities

$? \dots ? \mathbf{11}$  Using the top augends from Figures: **1a**, **1c**  
 $? \dots ? \mathbf{11}01$  and **1e**. Note that the leading pair of specified  
 $? \dots ? \mathbf{11}0101$  bits is always **11**, and that 01 is appended to  
 $\vdots$  get the next entry. The bottom row illustrates  
 $? \dots ? \mathbf{11}01 \dots 01 \dots 01$  the generalization of *m* parity (odd).

Figure 4: How any magnitude of even or odd *m* parity can be Obtained From *TP1*

(a) Showing that an entry of Figure 3 has *m* parity. (odd)  
(*h* = height)

$$\begin{array}{r}
 h \quad ? \dots ? 1101 \dots 01 \dots 01 \\
 2h \quad ? \dots ? 11010 \dots 10 \dots 10 \\
 + \quad \quad 1 \\
 \hline
 3h + 1 \quad ? \dots ? 100000 \dots 00 \dots 00
 \end{array}$$

(b) Showing that an entry of Figure 3 has *m* parity. (even)  
(*h* = height)

$$\begin{array}{r}
 h \quad ? \dots ? 001 \dots 01 \dots 01 \\
 2h \quad ? \dots ? 0010 \dots 10 \dots 10 \\
 + \quad \quad 1 \\
 \hline
 3h + 1 \quad ? \dots ? 100000 \dots 00 \dots 00
 \end{array}$$

## 4 The $m$ Parity Distribution Pattern From Positive Integers

That is, how the distribution pattern of  $m$  parity values occur, as yielded by  $\mathbb{N}_1$ , and the infinite subsets of  $\mathbb{N}_1$ . (Specifically, the arithmetic progressions  $k \cdot \mathbb{N}_i$ .) Of interest here is how varying  $i$  and  $k$  affect the resulting  $m$  parity distribution so as to classify them. First we look at only the frequencies of  $m$  parities in  $\mathbb{N}_1$  integers. Then we look at how these  $m$  parities are distributed within the sequences.

### 4.1 Frequency of a Given $m$ Parity in Consecutive Integers (Binary, Bit Length $n$ )

(The plain binary is not to be confused with a ones-phobic binary.)

**Lemma 4.1.** *By definition of  $m$  parity,  $(m+k)$  parity has  $k$  more consecutive trailing zeroes than  $m$  parity. Let both of the above be in the set  $\mathbb{B}_n$ . That is, both have the same number of bits in total. Where,  $\mathbb{B}_n$  contains all of the consecutive integers from  $00 \cdots 00$  to  $11 \cdots 11$  (All possible bit values in a string, or sequence, of length  $n$ .)*

*Proof.* Since,  $m$  parity specifies an  $m$  number of consecutive trailing zero bits with the leading end demarcated by a set bit or 1. There are  $m + 1$  specified bits. Conversely, there are  $n - (m + 1)$  unspecified bits in  $\mathbb{B}_n$ . Thus, there are  $2^{[n-(m+1)]}$  values of  $\mathbb{B}_n$  having that  $m$  parity.

Similarly, there are  $n - (m + 1 + k)$  unspecified bits for the ones-phobic binaries in an  $(m+k)$  parity in  $\mathbb{B}_n$ . Also, there are  $2^{[n-(m+1+k)]}$  values of  $\mathbb{B}_n$  having  $(m+k)$  parity. Taking the ratio of these amounts:

$$\frac{2^{[n-(m+1)]}}{2^{[n-(m+1+k)]}} = 2^k \quad (2)$$

Consequently, there are  $2^k$  times as many ones-phobic binaries having  $m$  parity as there are having  $(m+k)$  parity, (in  $\mathbb{B}_n$ ).  $\square$

### 4.2 Ruler Harmonics

**Definition 4.1** (*ruler harmonics*). Let indices refer to an interval of consecutive positive integers  $k \cdot \mathbb{N}_i$  (arithmetic progression). More specifically, let the indices range from  $i$  to  $i + 2^n$ , such that  $i \geq 1$  and  $n \in \mathbb{N}_1$ . The index length is arbitrary but must be at least two. (The reason for this covered in Section 5.3). A length of  $2^n$  will contain a segment which repeats every  $2^n$  elements, with the exception of the transversal. (explained in Figures 28 and 27). Now focus on the  $m$  parity of these indices. The odd indices have an  $m$  parity of zero. At a given even starting point, every second index has *at least* a 1 parity (the even integers). At the same starting point, each fourth value is *at least* 2 parity. Similarly, each eighth value is *at least* 3 parity. In general, each  $2^k$ th value is *at least*  $k$  parity; etc. (This is proved in Lemma 4.1.) The exact distributions of  $m$  parities in the sequence depends on the origin of the range looked at, here labeled as  $i$ . A bar chart can be drawn to illustrate a spatial pattern. (Look at Figure 7) Here, the length of the bars correspond to the degree of  $m$  parity. The placement of bars of a given

length are reminiscent of scale marks (on a ruler). This is why the term **ruler harmonics** is used to describe the frequency distribution pattern, over a given range of positive integers.

Note: Ruler harmonic sequences are primarily derived from indices, namely an arithmetic progression that is a sub-sequence of  $\mathbb{N}_1$ . Ruler harmonic sequences can be indirectly derived from elements of the dissonance start column of a tier. Here, instead of cascade offsets, tier offsets are used. Tier offsets still form arithmetic progressions, albeit with a different common difference, which still lead to a ruler harmonic sequence. (Refer to Section 12. Ruler harmonic sequences can also be indirectly derived from the beans of a tiara. (Refer to Section 13.)

**Definition 4.2** (*ruler subharmonics*). Subsets of  $\mathbb{N}_1$ , namely  $6^n \cdot \mathbb{N}_1$ , where  $n \geq 2$ , can also form a pattern similar to ruler harmonics, defined here as **ruler subharmonics**. These are illustrated, with their own colours in Figure 7 and Figure 30. Note, **ruler harmonics** exhibit the self symmetry property, in particular, scale invariance (zoom out only dilation).

**Definition 4.3** (*order, of a ruler harmonic series*). The *order* of a ruler harmonic series is the magnitude of the smallest element(s) in the series. For an example refer to the note in Figure 7. Note: *ruler subharmonics* is a generalization of all ruler harmonic series having an order greater than zero.

A ruler harmonic series of order  $k$  is abbreviated as:  $rhs^{\underline{k}}$ . Note: To distinguish that the superscript is an order, of the ruler harmonic series, and not an exponent; the superscript is underlined. Addition can occur with superscripts but they do not have the properties of exponents:  $rhs^{\underline{k}+2} \neq [rhs^{\underline{k}}]^2$

#### 4.2.1 Deriving Ruler Harmonics in Positive Integer Arithmetic Progressions.

**Theorem 4.2.** *A an arithmetic progression  $k \cdot \mathbb{N}_i$ , where  $k = 1$  and  $k \in \mathbb{N}_1$ , which as indices, will have their  $m$  parities follow ruler harmonics. The origin, of the index range is arbitrary, but  $\geq 1$ . The length of the index range is greater than 1.*

*Proof.* Regard  $\mathbb{N}_1$  (as binary). To obtain the next successive integer a 1 is added to the current integer. In binary addition, a lesser significant bit, 0 is replaced by a 1. While a 1 is replaced by a 0, with a 1 to carry to the next significant bit. Repeatedly adding 1 effectively creates a binary odometer. Where the lesser significant bit toggles between 0 and 1, twice as often as the next more significant bit, etc. Consequently,  $m$  parity frequency in  $\mathbb{N}_1$  (binary), halves with each increase in  $m$  parity. This explains the *ruler harmonics* for  $m$  parity, as illustrated in Figure 7. This argument is corroborated by Section 4.1, to establish ruler harmonics in integers.  $\square$

## 5 Classifying Sequences With Respect to Their $m$ Parity Distribution Type

We next need to look at what happens when we select every  $n$ th term in a ruler harmonic sequence. So we tabulate the categories of relevant sequences in terms

of sequence notation. <sup>1</sup> A function called  $mp$  is introduced to disambiguate the term *ruler function* (which is used to generate either sequence A007814 or sequence A001511 in OEIS) <sup>2</sup>.

The  $mp$  function acts upon an index (the input). The index ( $\in k \cdot \mathbb{N}_i$ ), must be an [arithmetic progression](#). The index origin must be  $i \geq 1$ . Also, the common difference must be  $k \geq 1$ . Function  $mp$  returns the  $m$  parity of  $m \in \mathbb{N}_i$ . Source code for function  $mp$  is shown in Appendix G. These index derived  $m$  parity sequences are distinguished into two categories:

1. A [constant sequence](#), with constant  $k$  (from  $n \in \mathbb{N}_i$ ), is denoted as  $cs^k$ .
2. A [ruler harmonic](#) sequence of order  $k$  (from  $n \in k \cdot \mathbb{N}_i$ ), is denoted as  $rhs^k$ .

## 5.1 The Frequency Distribution of $m$ Parities in an $m$ Parity Sequence

The frequency distribution of  $rhs^k$ ,  $k \in \mathbb{N}_0$ , is a mapping of the elements in  $rhs^k$ , enumerated by  $m$  parity, to a powers of two distribution.

*Remark 5.1.* Given two  $m$  parity sequences, each finite but arbitrarily large (approaching infinity), in length. Let the first sequence be  $cs^k$  and the second  $rhs^{k+1}$ . Then the combined  $m$  parity frequency distribution of both sequences will approach the  $m$  parity distribution of  $rhs^k$ . Why? There is a bijective mapping between each  $m$  parity of  $rhs^k$  with the combined sequence pair of  $cs^k$ ,  $rhs^{k+1}$ . Since, by definition,  $k$  is the minimum in  $rhs^k$  and  $k+1$  is the minimum in  $rhs^{k+1}$ . Then, combining  $cs^k$  with  $rhs^{k+1}$  introduces a new minimum  $k$  to the combination. Where  $rhs^{k+1}$  maps to every  $m$  parity in  $rhs^k$  except for the minima  $k$ . However, this is accounted for by mapping the minima  $k$  from  $cs^k$ .

## 5.2 Sequential Ordering of $m$ Parities in an $m$ Parity Sequence

Not only does the frequency distribution of  $m$  parities matter; the sequential ordering of  $m$  parities in a given sequence also matters. To verify ordering of a given sequence we can compare it to the [standard sequence](#), denoted as  $\mathbf{S}$ . Now  $\mathbf{S}$  is generated, by the function  $mp$ , from an index, called the [main index](#), which is  $\mathbb{N}_1$ . (This is by Definition 1 an arithmetic progression whose origin and common difference is 1.) The resulting  $\mathbf{S}$  is a ruler harmonic sequence of order zero. In other words, the contents of  $\mathbf{S}$  is the 2-adic valuation derived from the monotonic increasing ordering of  $n \in \mathbb{N}_1$ .

By design<sup>3</sup>, we overload the  $\lesssim$  relation, restricting it's application to  $\mathbf{S}$ ,  $cs^k$ , and  $rhs^k$ , since we are only concerned with the following distinctions:

- i. We say two sequences are  $\lesssim$  to each other, when both are ruler harmonic sequences; and the sequence to the left of the relation,  $\lesssim$ , is equal or *higher* in order than the sequence to the right of that relation.

$$\dots rhs^{k+2} \lesssim rhs^{k+1} \lesssim rhs^k \lesssim \mathbf{S} \quad \text{such that } n \in \mathbb{N}_0 \quad (3)$$

<sup>1</sup>Sequences are denoted by sub-scripted parenthesis  $()_{i \in \mathbb{N}}$ .

<sup>2</sup>THE ON-LINE ENCYCLOPEDIA OF INTEGER SEQUENCES®, founded in 1964 by N.J.A. Sloan, <https://oeis.org>

<sup>3</sup>By convention, whenever  $A$  is a subsequence of sequence  $B$ . The relation of  $A$  to  $B$ , being a preorder, is denoted as:  $A \lesssim B$

- ii. A constant sequence,  $cs^k$ , can still be  $\lesssim$  to another constant sequence even if each sequence contains different constants, and/or lengths.
- iii. We only distinguish between a constant sequence and a ruler (sub)harmonic sequence of any order.

Then always in this context:

$$cs^k \not\lesssim rhs^k \quad \text{such that } n \in \mathbb{N}_0 \quad (4)$$

The standard sequence is written as:

$$\mathbf{S} = \left( mp(n) \right)_{n \in \mathbb{N}_1} \quad (5)$$

Where by definition,

$$rhs^k \lesssim \mathbf{S} \quad \text{such that } n \in \mathbb{N}_0. \quad (6)$$

Also by definition,

$$cs^k \not\lesssim \mathbf{S}. \quad (7)$$

### 5.3 Requirements Needed to Determine if a Sequence is a Ruler Harmonic Sequence

First the sequence to be matched must be derived from an index (that is an input to the  $mp$  function). The index must be an [arithmetic progression](#) whose length is at least two. Apart from the minimum length stipulation, the definition for  $rhs^k$  does not have any other specific length restriction (a defining property  $rhs^k$ , refer to Section 4.2). The minimum length requirement ensures that sequence being matched will contain at least one minimum. A [ruler subharmonic](#) sequence requires that every other (second), element in the sequence be an  $m$  parity minimum. (Sub)sequences can match anywhere along the standard sequence. The term match means that a subsequence exists in a larger sequence.

### 5.4 Procedure to Determine if a Sequence is a Ruler Harmonic Sequence

We can establish whether a given sequence  $A$  is, or is not, a ruler (sub)harmonic sequence. (Proviso:  $A$  must be generated from an arithmetic progression that is the index [input], to the  $m$  parity function.) Let  $k$  be a global minima in  $A$ . Convert  $A$  to a zero order ruler harmonic series by performing an element wise subtraction of  $k$  from  $A$ . Call the result  $W$ .

How to check if  $W \lesssim \mathbf{S}$ .

- i. Find the location of the first overall maximum in  $W$ . Call this location,  $mw$  (the distance from the origin of  $W$ ).
- ii. Let  $piv$  be a distance to the origin of an element in  $\mathbf{S}$ . Let  $piv$  always point to a location in  $\mathbf{S}$  whose element has the same value as the value pointed to by  $mw$ . Set  $piv$  to point to the first such location in  $\mathbf{S}$ .

- iii. If  $piv < mw$ , find and reset  $piv$  to point to the next location residing in  $\mathbf{S}$  whose value is the same as the maximum in  $W$ .
  - iv. Repeat (iii.) until the  $piv \geq mw$ .
  - v. Now all of the elements of  $W$  can be compared with a subsequence of  $\mathbf{S}$  that is the same length as  $W$ .
  - vi. If an ordered pairwise comparison of  $W$  with that subsequence of  $\mathbf{S}$  shows pairwise equality then, in this context, we say that  $W \lesssim \mathbf{S}$ .
- Then we can say that  $A$  is a ruler harmonic series of order  $k$ .

Note: adding a constant positive integer,  $c$ , as an  $m$  parity, to each element of a ruler harmonic (sub)sequence, of order  $k$ , yields a ruler harmonic (sub)sequence of order  $c + k$ . (By definition 4.3.)

An example using a builtin method for matching a substring to a longer string is shown in G.1.2 of Appendix G.

## 6 Fundamental Property Of $m$ Parities

First an identity is established.

**Theorem 6.1.** *Let the constants be:  $a, b \in \mathbb{N}_1$ . Then:*

$$mp(a \cdot b) = mp(a) + mp(b) \quad (8)$$

$$mp\left(\frac{a}{b}\right) = mp(a) - mp(b) \quad (9)$$

*Proof.* The property of 8 is established by using the binary form of an integer along with the property of binary multiplication. Specifically, the sum of the trailing zeroes of two binary numbers is the number of trailing zeros in their product. Trailing zeros can be factored out of a number as a power of two. Since the exponents of a power of two are the constituents of an  $m$  parity. We can use the identity:

$$a2^x \cdot b2^y = ab2^{x+y} \quad (10)$$

Re-writing the exponents of identity 10 in terms of the  $mp$  function yields equation 8.

By similar reasoning, going in reverse, multiplication can be replaced by division. Since the exponents of a power of two are the constituents of an  $m$  parity. We can use the identity:

$$a2^x \div b2^y = \frac{a}{b} \cdot 2^{x-y} \quad (11)$$

Re-writing the exponents of identity 11 in terms of the  $mp$  function yields equation 9. □

**Corollary 6.2.** *A special case. From equation 8, let  $a = 2^k$ . Then:*

$$mp(2^k \cdot b) = k + mp(b) \quad \text{where constants } k, b \in \mathbb{N}_1 \quad (12)$$

*Proof.* Let  $a = 2^k$  in equation 8. This forms:

$$mp(2^k \cdot b) = mp(2^k) + mp(b) \quad \text{whose constants are } b \text{ and } k \in \mathbb{N}_1 \quad (13)$$

Now  $mp(2^k)$  is the number of trailing binary zeros in  $2^k$ , which is  $k$ . Then equation 13 becomes equation 12  $\square$

## 7 Fundamental Properties of $m$ Parity Sequences

First: we look at multiplication by  $2^k$ . Second: we look at what happens to the  $m$  parities of a sequence, when the origin is changed. Then examples are used illustrate the derived results.

### 7.1 Effect of Varying Origin and Varying Jump Increment, on $rhs^k$

**Theorem 7.1.** *Let  $n$  be an element of the main index. Let  $j$  be the index origin,  $k$  the order in  $rhs^k$ . The function  $mp$  is described in G. Where  $j, k \in \mathbb{N}_1$ . Then the origin determines whether the  $m$  parity series is a ruler harmonic series or a constant series. That is:*

$$\begin{aligned} \left(2^k \cdot mp(n)\right)_{n \in \mathbb{N}_j} &\lesssim rhs^k && \text{if } 0 \equiv j \pmod{k} \\ \left(2^k \cdot mp(n)\right)_{n \in \mathbb{N}_j} &\lesssim CS && \text{if } 0 \not\equiv j \pmod{k} \end{aligned}$$

*Proof.* The proof is subdivided into the following subsubsections:

#### 7.1.1 Multiplying by Two to the Power of $n$

Sequentially, multiplying an index by  $2^y$  increases the number of trailing zeros ( $m$  parity), of each element, by  $y$ . That is, the series  $rhs^k$  when multiplied by  $2^k$  yields the result is  $rhs^{k+y}$ . (By the property of equation 12.)

#### 7.1.2 Changing the Origin in a Ruler Harmonic Sequence of Order 0

Changing the origin involves adding a positive integer, the new origin, to each element of the index. Adding a non-zero constant to each index element of  $rhs^0$  is the same as selecting a subsequence of  $rhs^0$  starting at that new origin. Alternatively stated,  $rhs^0$  is truncated to the new origin. By definition 4.1, the new subsequence is still of the type  $rhs^0$ .

#### 7.1.3 Changing the Origin in a Ruler Harmonic Sequence of Order $k$

Suppose, instead we have a  $rhs^k$ , then each index is composed of an odd factor and a even factor of  $2^n$ . (This being the trailing zeros common to every index value.)

Adding an origin to an index element, of  $rhs^k$  can set the  $m$  parity series type in two ways.

#### 7.1.4 $m$ Parity Clamp

There exists an  $m$  Parity Clamp if the origin addend is less than the  $2^n$  factor. Look at the integers in binary form (in Figure 5a). If the least significant 1's bit position, of the origin addend is of lesser, or equal, significance than the most significant zero of the  $2^n$  factor of the augend; then the least significant 1's bit position of the origin addend sets the  $m$  parity of the result. Moreover, this origin addend sets (clamps), the  $m$  parity of each element in the resulting series. Since each index value, the augend, has more trailing zeros than the addend.

Since the origin addend has a lower  $m$  parity than the index augend; the index origin  $j$  can not divide into  $k$ , the order of  $rhs^k$ . Then the condition of no clamping can be expressed as:

$$0 \not\equiv j \pmod{k} \quad \text{where } j, k \in \mathbb{N}_1$$

.

Since clamping sets every  $m$  parity of a sequence to the same constant value, this can be written as:

$$\left(2^k \cdot mp(n)\right)_{n \in \mathbb{N}_j} \lesssim CS \quad \text{provided } 0 \not\equiv j \pmod{k} \quad \text{and } j, k \in \mathbb{N}_1 \quad (14)$$

Figure 5: Generalization (in Binary): Adding an Origin to an Index

index:		
origin:	(a) Clamping	(b) No Clamping
	$\begin{array}{r} \text{?} \dots \text{?} 100 \dots 0 \\ + \text{?} \dots \text{?} \text{?} 10 \dots 0 \\ \hline = \text{?} \dots \text{?} \text{?} 10 \dots 0 \end{array}$	$\begin{array}{r} \text{?} \dots \text{?} \text{?} 100 \dots 0 \\ + \text{?} \dots \text{?} 1000 \dots 0 \\ \hline = \text{?} \dots \text{?} \text{?} 100 \dots 0 \end{array}$

#### 7.1.5 No Clamping

The elements of sequence  $rhs^k$  can be factored into two series of factors:  $rhs^0$  and the constant series  $(2^k)$ , where  $k \in \mathbb{N}$ . This is done by element-wise application of the property from equation 12. In the series of Figure 26 these two factors are highlighted as different coloured columns. The factors forming a constant series, here as  $2^1$ , is illustrated as a column of least significant zeros is highlighted in gold. The columns forming the  $rhs^0$  factors are highlighted in increasing significance as: red, blue, and violet.

When, the origin addend has a greater, or equal,  $m$  parity than index augend, then no  $m$  parity clamping by the origin addend occurs. In essence the origin addend is only added to the  $rhs^0$  factor, while skipping over the  $(2^k)$ , factor. The result is the same as in Section 7.1.2.

Since, the origin addend is of higher, or equal,  $m$  parity than the index augend; the index origin  $j$  divides into,  $k$  the order of  $rhs^k$ . Then clamping can be paraphrased as:

$$0 \equiv j \pmod{k} \quad \text{and } j, k \in \mathbb{N}_1$$

Now looking just at the  $rhs^0$  factor: Adding a constant,  $j$ , to  $rhs^0$  changes the origin of the sequence, as in Section 7.1.2, the result is:

$$\left( mp(n) \right)_{n \in \mathbb{N}_j} \quad (15)$$

Re-introduce  $2^n$  by multiplying this factor to each term of 15. Then definition 6 yields:

$$\left( 2^k \cdot mp(n) \right)_{n \in \mathbb{N}_j} \lesssim rhs^k \quad \text{provided } 0 \equiv j \pmod{k} \quad \text{and } j, k \in \mathbb{N}_1 \quad (16)$$

□

## 7.2 Multiplying a Sequence By an Odd Positive Integer

**Theorem 7.2.** If  $\left( mp(n) \right)_{n \in \mathbb{N}_j} \lesssim rhs^k$ , then  $\left( p^k \cdot mp(n) \right)_{n \in \mathbb{N}_j} \lesssim rhs^k$ , where  $p$  is an odd positive integer.

*Proof.* Switching to binary mode. When repeatedly adding a positive integer an odd number of times the least significant ones bit will remain at the same placeholder (highlighted in red, look at Figure 6).

Figure 6: Multiplication by and Odd Integer, Leaves  $m$  parity Unchanged. (Shown as Repeating an Odd number of Additions of a Binary to Itself)

(a) $n$ parity	(b) 0 parity
$?\dots ?\mathbf{1}0\dots 0$	$?\dots ?\mathbf{1}$
$\vdots$	$\vdots$
odd number of additions	odd number of additions
$\vdots$	$\vdots$
$?\dots ?\mathbf{1}0\dots 0$	$?\dots ?\mathbf{1}$
+	+
$?\dots ?\mathbf{1}0\dots 0$	$?\dots ?\mathbf{1}$

Consequently, sequentially multiplying a sequence by an odd positive integer leaves the  $m$  parity of the sequence unchanged. □

## 7.3 An Index Advance by $3^k$

Let  $j, k, n \in \mathbb{N}_1$  also  $k$  can be equal to zero.

$$\left( mp(3^k \cdot n) \right)_{n=j} = \left( mp(3^k) + mp(1^k \cdot n) \right)_{n=j} \quad (17)$$

$$= \left( 0 + mp(1^k \cdot n) \right)_{n=j} \quad \text{since } 3^k \text{ is odd} \quad (18)$$

$$= \left( mp(n) \right)_{n=j} \lesssim rhs^0 \quad (\text{by Section 5}) \quad (19)$$

The rationale for equations 17 through 18:

- (17) By equation 12.
- (18) Now  $3^k$  is odd since the product of its factors is odd. Then sequentially multiplying a sequence by  $3^k$  leaves the  $m$  parity of the sequence unchanged. That is, the  $3^k$  term adds in an  $m$  parity of 0. (by Lemma 7.2)
- (19) Note: changing the origin does not affect the sequence order or sequence type. Section 7.1.2. The relation is defined in equation 5 and equation 6.

For an example, look at Appendix F.2.

## 7.4 An Index Advance by $6^k$

Since  $3^k$  is a composite of the factors  $1^k$  and  $3^k$ , the properties of these factors determine the sequence type:

Let  $j, k, n \in \mathbb{N}_1$  also  $k$  can be equal to zero.

$$\left( mp(6^k \cdot n) \right)_{n=j} = \left( mp(2^k 3^k \cdot n) \right)_{n=j} \quad (20)$$

$$= \left( mp(3^k) + mp(2^k \cdot n) \right)_{n=j} \quad (21)$$

$$= \left( 0 + mp(2^k \cdot n) \right)_{n=j} \quad \text{since } 3^k \text{ is odd} \quad (22)$$

$$= \left( mp(2^k \cdot n) \right)_{n=j} \lesssim rhs^k \quad \text{if } 0 \equiv j \pmod{k} \quad (23)$$

$$\text{or } \left( mp(2^k \cdot n) \right)_{n=j} \lesssim CS \quad \text{if } 0 \not\equiv j \pmod{k} \quad (24)$$

The rationale for equations 21 through 24:

- (21) By equation 12.
- (22) Now  $3^k$  is odd since the product of its factors is odd. Then sequentially multiplying a sequence by  $3^k$  leaves the  $m$  parity of the sequence unchanged. That is, the  $3^k$  term sequentially adds in an  $m$  parity of 0. (by Lemma 7.2)
- (22) Sequentially multiplying a sequence by an even positive integer, adds  $k$  to the parity of the sequence (it's order). The addend to the parity,  $k$ , is introduced by factoring out  $2^n$ . (By Section 7.1.1)
- (23) Varying the origin of an  $rhs^k$  will either leave the order of the sequence unchanged or convert it to a constant sequence. This depends on whether or not the origin  $m$  parity clamps that sequence. (refer to Section 7.1.4 and Section 7.1.5)
- (24) A defining property, refer to Lemma 7.1.

For an example look at Table 12. Here, the origin corresponds to a row entry that is either  $rhs^1$  or  $cs^0$ .

#### 7.4.1 How a Sequence Transitions From Being Clamped to Being Un-clamped

Consequently, An arithmetic progression, whose common difference is  $6^k$ , used as an index to  $mp$  will either be:  $rhs^k$  (if no clamping occurs), or  $CS$  (if clamping occurs). By equations 23 and 24.

Sequentially dividing an  $m$  parity sequence by 2 reduces one trailing zero from each sequence element, provided  $m \geq 1$  (recall equation 9). For the following, refer to Sections: 7.1.5 and 5a. The series index common difference  $6^k$  can be factored into  $2^{k+1}$  and  $3^k$ . These factors can each designate a series of their own.

#### 7.4.2 Sequence Transition From Clamped to Un-Clamped (Illustrated in Binary)

In the series of Figure 26 these two factors are highlighted as different coloured columns. In this figure, the factors forming a constant series, is the factor sequence  $2^1$ , illustrated as a column of least significant zeros (highlighted in gold). The columns forming the  $3^k$  factors are highlighted, as bit pairs, in increasing significance as: red, blue, and violet. Here, the origin is 6. Applying the condition to determine clamping (from Lemma 7.1) yields:  $0 \equiv 6 \pmod{6}$ , which indicates no clamping. Yielding the series:  $rhs^1$ . Figure 26 shows what the order of the series  $rhs^1$  looks like.

Returning to the generalization of using  $6^k$  as an index common difference.

Upon division by 2 the least significant 1's bit position, of the origin addend, becomes closer, in significance, to the  $3^k$  factor series. Eventually after repeated division, the least significant 1's bit position, of the origin addend, is *inside*, in terms of significance to the  $3^k$  factors. Once *inside*, the  $3^k$  factors, the least significant 1's bit position no longer clamps the sequence (by Section 7.1.5). Then the previously clamped sequence transforms from a constant sequence to a  $rhs^k$  sequence.

#### Summarizing This Section With Concluding Remarks

*Remark 7.1* (Generating, Constant or Ruler Harmonic, Sequences From Powers of Six). Begin with an index generated  $m$  parity sequence. (That is generated from an arithmetic progression  $j \cdot \mathbb{N}_i$ , with  $j = 6^n$ , and  $n, i, j \in \mathbb{N}_1$ .) Even if this sequence is a constant sequence. That is of the type  $cs^k$ , recall  $k$  is the order which is the  $m$  parity (the number of trailing zeros). Continued division by 2 (on the entire sequence), will eventually yield a  $cs^0$  sequence, whose index is an arithmetic progression where the common difference is  $3^k$ ,  $k \in \mathbb{N}_0$ , since the  $2^k$  is factored out. For the next move, instead of  $DV2$ ,  $TP1$  is applied to each sequence element. Ultimately, this yields  $rhs^0$  (by equation 19).

*Remark 7.2* (Division of Ruler Harmonic Sequences by  $2^n$ ). Division by  $2^n$  (where  $n \in \mathbb{N}_0$ ) is allowed when performing division on  $rhs^k$ , only when  $k > 0$ . Moreover, division by  $2^n$  is deemed to be undefined, the resulting series can not have a lower order than  $rhs^0$ . (Related: Remark 9.2)

## 8 Cascades, Seed Ranges, Flutes, Offsets, and Tessellation

### 8.1 The Cascade

Figure 8a is an example of a **cascade** (a two dimensional array consisting of vertically aligned flute(s), in each column). The focus now is on what happens when the *columns* are traversed.

Whether they show heights,  $m$  parities, offsets, or ones-phobic binaries; cascade columns are labeled by their move, as follows: A column of seed values is column 0. The column of the first move is column 1. The column of the second move is column 2, etc.

### 8.2 Seed Range

**Definition 8.1** (seed range). Within the seed column (the first column labeled 0 of Figure 8a), the difference between successive seed values is called the *seed range*. Between successive even values (or successive odd values), the seed range is 2, denoted as:

$$\Delta r = 2 \quad (25)$$

### 8.3 Flute

**Definition 8.2** (*flute*). The length of a flute is a seed range that is a power of 2. Flutes are demarcated by a thicker coloured border as shown in Figure 8. A flute is half as long as the one in the column to the right of it. Column-wise flutes can immediately follow each other (without gaps). However, they do not overlap. A horizontal collection of progressively longer flutes is called a pan flute template or Q template as illustrated in Figure 8a. (The tops of these flutes align at a single row. Flutes are shown as thicker cell borders illustrated with colours that vary column by column refer to Figure 8b).

### 8.4 Offsets

**Definition 8.3** (*offset*). Within this context, an offset refers to the difference in a pair of heights. Where, both heights are in *successive* flutes of the same length (within a column). More specifically, each height is in the same location, within their flutes, relative to the top their flutes. (That is, having the same **phase shift**.) Both sub-figures of Figure 11, are used for the following example:

Refer to the **Cascade of Heights** diagram of Figure 11 column 3, row 7. The height is 34. In the same corresponding position (**phase shift**), within the next lower flute of column 3 row 15 the height is 70. The difference in heights is  $70 - 34 = 36$ . The result is recorded in column 3 row 15, from the adjacent diagram **Cascade of Offsets**. Note: Flutes are delineated by borders with a disparate colour, to better distinguish flute lengths for each column.

Types of element tessellation among column-wise flutes are illustrated using icicles within landings (constant sequences), and traversals within landings (ruler harmonic sequences). The following section takes an algebraic approach using interleaving arithmetic progressions whose  $m$  parity sequences are either constant or ruler harmonic.

## 9 Placements and Landings

### 9.1 Using Modulo Arithmetic to Generate Ruler Harmonic Sequences

The main reference in this section is Figure 28, this figure is explained by a key which is shown in Figure 27.

The  $m$  parity for an integer in  $\mathbb{N}_1$  can be obtained by using modulo arithmetic. Specifically, counting the number of divisors of successive powers of 2 starting with an exponent of 1. Since the number of trailing zeros of an integer is also the number of powers of 2 that divide that integer. Refer to the notes of Figure 28.

When evaluating the remainders, for a given modulus, of successive values of  $\mathbb{N}_1$ , the resulting a least residue system will tessellate. Why? For succeeding integers, every time a multiple of a modulus is reached the remainder is repeated (by the property modulo arithmetic).

Suppose an output of row-wise tessellating least residue systems based on  $(\text{mod } 2^n)$  are arranged as rows in an array (one row for each exponent  $n$ ). Refer to Figure 27.

Note that all ordered least residue systems contain a remainder of zero, (due to the analogue clock property of modulo arithmetic). In other words, since the index is monotonically increasing by a constant factor, landings can be shifted (translated) along a sequence; such that the last placement of a landing is zero.

*Theorem 9.2* (Changing Landing Boundaries Along an  $m$  Parity Sequence, by Translation).

Suppose there are contiguous landings of equal length of  $2^k, \in \mathbb{N}_1$  partitioning an infinitely long  $m$  parity sequence. Within each landing, the number of disparate  $m$  parities (represented as identical icicles, including their traversal), will remain the same, provided the landing length of  $2^k$  where  $k \in \mathbb{N}_1$ , is kept the same everywhere along that sequence.

*Proof.*

The translation property of the congruence relation is:

$$a + c \equiv b + c \pmod{n}, \quad \text{where } c \text{ is any integer} \quad (26)$$

Let  $a - b$  be the length of a landing which is the modulus  $n$ . Let  $a$  be the index value at position  $c$  within a landing. Let  $b$  be the index value at the same relative position  $c$  within the next landing. Then, even while repeatedly being incremented by  $c = 1$ ,  $a$  and  $b$  have a common remainder. Consequently, the placements as a least residue system, tessellate from landing to landing. Observe tessellation along any  $(\text{mod } n)$  row in Figure 28.

Placement tessellation allows for the following: When a landing is translated (shifted left or right), by one move, along a  $(\text{mod } n)$  row, a placement's value that is lost at one end of a landing is introduced at another end of that landing. Regard several contiguous landings as a block. Then placements, as several successive least residue systems, tessellate as a block. Similarly, when a block is translated by one move, the placement's value that is lost at one end is introduced at the other end of that block.

An *icicle sector* is delimited by contiguous rows containing blocks of landings that are confined within the span of the pivot row. The landing length within contiguous rows ranges in this order:

$$(\text{mod } 2^1), (\text{mod } 2^2), \dots, (\text{mod } 2^n)$$

Here  $2^n$  is the length of the landing within the pivot row.

For an *icicle sector* (Figure 27), when the remainder is zero and the zero remainders line up (at the end of a landing), the formation is called an icicle (also shown in Figure 28).

Just as translation occurs individually among landings and blocks along a row. The translation can occur simultaneously among rows, provided landings align with blocks containing shorter landing. (Two landings align with a twice as long landing in a subsequent row.) Then an icicle formation will tessellate from one icicle segment to the next. Although the length of the transversal varies from one icicle sector to the next. (The length being the  $m$  parity at the associated index.) Notice that the distance between traversals is the length of the pivot landing. For the moment we are only concerned with the length of the traversal up to and including (but not past), the pivot landing. In this way we can say that the icicles along with the traversal tessellate from one icicle sector to the next. Shifting simultaneously all landing boundaries within the icicle segment while keeping their lengths the same is a translation. Enough shifts will move the relative position of a traversal within a landing (from end to the other). A leading icicle (or traversal), of a given length (within the confines of the icicle sector), would disappear from one end of the icicle sector but would reappear at the other end. This process maintains the same number of disparate icicles or traversals within an icicle sector (provided the span of the icicle sector is a power of two). The length of an icicle (or traversal), is the  $m$  parity of the associated index for the residing column. Therefore, the number of disparate (unique in  $m$  parity) values remains the same for a given fixed span of  $2^n$  where  $n \in \mathbb{N}_1$ , regardless of where that span lies along the  $rhs^0$ .  $\square$

*Theorem 9.3* (Enumeration of the Number of Disparate  $m$  Parities Within a Power of Two Seed Range (Along a Ruler Harmonic Sequence of Order One)).

Given an icicle sector with a length, or seed range, of  $\Delta r = 2^k$  where  $k \in \mathbb{N}_1$ . This icicle sector will contain icicles with a traversal that define  $k + 1$  disparate  $m$  parities.

*Proof.* Doubling the icicle sector span,  $\Delta r$ , introduces a new row with a pivot landing twice as long as the previous pivot landing. One of the two traversals will not reach the new pivot landing. It will be classified as an icicle, increasing the icicle count. The other traversal will reach or go beyond the new pivot landing. Then the traversal count stays the same. This is due to the property of  $rhs^0$ .

Since: By definition of  $rhs^0$  every 2nd  $m$  parity is a 0 parity, every 4th  $m$  parity is a 1 parity, every 8th  $m$  parity is a 2 parity. In general, every  $(2^k)$ th  $m$  parity is a  $(k - 1)$  parity. Then a landing of  $\triangle r = k$  will contain the following  $m$  parities:  $0, 1, 2, \dots, k, t$ . Where  $t$  (the traversal occurs once), and  $0, 1, 2, \dots, n$  (are the icicles).

Since the other introduced icicles due to the doubling of the icicle sector span are the same length, as those in the landing of half size, they are not counted towards the total number of disparate icicles (whose lengths represent  $m$  parities). The number of traversals stays as 1. Then in general, the number of disparate  $m$  parities for a given landing length of  $\triangle r = 2^k$  is:  $k + 1$ .  $\square$

*Theorem 9.4* (Enumeration According to  $m$  Parity Within a Seed Span of  $2^k$ ). For  $k \in \mathbb{N}_1$ . There are  $2^{k-1}$  0 parities,  $2^{k-2}$  1 parities,  $2^{k-3}$  2 parities,  $\dots$  1  $(k + 1)$  parity; within a span of  $2^k$ .

*Proof.* The following criteria are used to enumerate the frequency distribution of each disparate  $m$  parity in a span of  $2^k$ , such that  $k \in \mathbb{N}_1$ :

- i. Given that within a span of  $2^k$ , such that  $k \in \mathbb{N}_1$  the number of unique  $m$  parities is  $k + 1$ .
- ii. For a span of  $2^k$ , such that  $k \in \mathbb{N}_1$  the frequency distribution is as follows: A successive  $(m + 1)$  parity has half the frequency as the previous  $m$  parity. (This is a defining property of ruler harmonics Section 4.2)
- iii. The right hand side of the following powers of two sequence is used to find the frequency distribution of  $m$  parties for the given span of  $2^k$ , such that  $k \in \mathbb{N}_1$ :

$$1 + \sum_{i=0}^n 2^{i-1} = 2^n$$

Using criteria i through iii yields:

For  $k \in \mathbb{N}_1$ . There are  $2^{k-1}$  0 parities,  $2^{k-2}$  1 parities,  $2^{k-3}$  2 parities,  $\dots$  1  $(k + 1)$  parity; within a span of  $2^k$ . (Illustrated by example in: Figure 28)  $\square$

Here a remark is used to encapsulate several theorems, to be referenced later as a single remark.

*Remark 9.1* ( $m$  Parity Enumeration With Regards to Sub-Sequence Span and Translation). Combining the results of Theorem 9.2 and Theorem 9.3 yields the following: Given a sub-sequence  $W$  where  $W \lesssim S$  with  $W$  having a span of  $2^k$ , such that  $k \in \mathbb{N}_1$ . Then, these properties hold:  $W$  has  $k + 1$  unique  $m$  parities. In particular, for these  $m$  parities,  $m \in 1, 2, 3, \dots, k, (k + 1)$  (that is the consecutive integers from 1 to  $(k + 1)$ ).

Here,  $(k + 1)$  is the magnitude of a traversal and the number of unique  $m$  parities, which holds *wherever*  $W$  is located along  $S$ .

Also, for  $k \in \mathbb{N}_1$ . There are  $2^{k-1}$  0 parities,  $2^{k-2}$  1 parities,  $2^{k-3}$  2 parities,  $\dots$  1  $(k + 1)$  parity; within its span of  $2^k$ .

This is also inferred by the self symmetry property, specifically scale invariance (zoom out only dilation), of ruler harmonics. (definition 4.2) For a visual example

look at Figure 7 one sub-range (the seed range from 2 to 32), of this diagram is illustrated in Figure 15 (space permits the inclusion of 0 parities here, which were omitted by space constraints in Figure 7).

This sub-range is included in diagram 12 look at the 5 parity, at the second column from the left with the seed value from 106 to 126 (sorted cascade only).

*Remark 9.2 ( $m$  Parity Properties Involved When Changing The Common Difference of an Arithmetic Progression).* Dividing an arithmetic progression by  $2^k$  decreases the common difference by  $2^k$ . (Refer to Lemma 9.1 specifically the *DV2* part). Multiplying an arithmetic progression by  $2^k$  increases the common difference by  $2^k$ . (Refer to Lemma 9.1 apply a similar approach as in *TP1* except use  $2^k$ .)

Multiplying the index of a ruler harmonic sequence by  $2^{\pm k}$  will change the order of the ruler harmonic sequence by  $\pm k$ . (Refer to Section 7.1.1, specifically Theorem 6.1 and Corollary 6.2.) Therefore, changing the common difference of an arithmetic progression by  $2^{\pm k}$  changes the order of a ruler harmonic series by  $\pm k$ , provided the order does not drop below zero. Here division by 2 is only defined for even integers greater than zero. Changing the common difference, of an arithmetic progression whose  $m$  parities form a constant sequence, leaves the constant sequence unchanged. (Here, a finite sequence length is not taken into account.)

To illustrate by example: Look at Figure 27 and Figure 28, traversals (by design in demarcation), are located last within a landing. The order depends on the landing length. The common difference,  $\delta = 2^k$ , can be viewed as the [landing](#) length; since one [traversal](#) is selected from each successive landing. Observe that selecting the traversal from successive landings forms a ruler harmonic sequence of a given order.

For examples illustrating the order of  $k$ th selected ruler harmonic sub-sequences refer to Figure 7 and Figure 29.

## 9.5 Illustrating Inter-Flute Tessellation In a Cascade

Column-wise inter-flute tessellation is more readily comprehended with diagrams. First compare the **Cascade of Heights** and the **Cascade of Offsets** diagrams in Figure 11. Note the columns which are equidistantly delineated into flutes. Move 0, (or column 0, in these diagrams), is the seed value column. The seed value column being  $\mathbb{N}_1$  manifests as  $rhs^0$  (refer to Section 7.1).

Then [landings](#) and [traversals](#) can be utilized when describing column 0. The concept of landings and traversals are developed in the comments of Figure 27 and Figure 28. Landings are also mentioned in the comments of Figure 29.

In Figure 11 the orientation of landings is columnar and part of a cascade; these landings are now referred to as flutes.

## 9.6 Why Cascade Flutes Tessellate Column-wise (With Respect to Their Constituent $m$ Parities).

The  $m$  parities do not actually tessellate column-wise from flute to flute. However, when looking at the arithmetic progressions that represent the phase shift congruent

$m$  parities (column-wise), the *sequences of a given type* (whether *rhs* of *cs*) and specified by order do tessellate. This is later shown in Table 2.

Since  $m$  parities of the seed column follow  $rhs^0$  (by Section 4.2.1). The  $m$  parities tessellate as described in Section 9.5. The  $m$  parities in the column for the first move are determined by *DV2* or *TP1* operations on the previous column. When either of these operations act on an arithmetic progression with a common difference, the result is also an arithmetic progression with a different common difference, as shown by:

**Lemma 9.1.** *Beginning with an arithmetic progression having a common difference of  $\Delta h$ . If exclusively either DV2 or TP1 operators are applied, then the result is in turn an arithmetic progression with the following common difference.*

*Proof.*

Using *DV2*, as an iterative form:

$$h_{i+1} = \frac{h_i}{2}$$

Evaluating the change in  $h$  (the common difference of the resulting arithmetic progression).

$$\Delta(h_{i+1}) = \Delta\left(\frac{h_i}{2}\right) \quad (27)$$

$$= \frac{1}{2}\Delta h_i \quad (28)$$

Applying *DV2* to an arithmetic progression, of even values, results in another arithmetic progression whose values and common difference are decreased as shown by equation 28.

Using *TP1* (as iterative form):

$$h_{i+1} = 3(h_i) + 1$$

Evaluating the change in  $h$  (the common difference of the resulting arithmetic progression).

$$\Delta(h_{i+1}) = \Delta(3h_i + 1) \quad (29)$$

$$= 3(\Delta h_i) + \Delta 1 \quad (30)$$

$$= 3\Delta h_i \quad (31)$$

Applying *TP1* to an arithmetic progression results in another arithmetic progression whose values and common difference increased by equation 31.  $\square$

As such, the origin of an arithmetic progression determines the  $m$  parity series type. When the origin  $i$ , has the property  $i \equiv 0 \pmod{\delta}$ , where  $\delta$  is the common difference. The series type is  $rhs^k$ . When  $i \not\equiv 0 \pmod{\delta}$  then the series type is  $cs^k$ . (The reasoning for this is explained in Lemma 7.1.)

Note that the sequence of offsets contained in a flute form a periodic pattern (observed in Figure 8). This pattern is repeated in the flutes of the same length that are consecutively tiled column-wise end to end (from below the first template

flute). Note: Only the offsets in the Q template need to be evaluated using equation 1, with the method shown in Section 8.4. To save on computation, the remaining offsets from the Q template flutes are copied to the next successive flute (going column-wise). This is allowed because the contents of each flute tessellate column-wise.

For disambiguation, the phrase “standard parity” is introduced. It is restricted to the dictionary entry of parity, and is not a general term for all other types of parity mentioned here. Moreover, standard parity is specified as either: even parity (trailing zero(s)), or odd parity (no trailing zero(s)).

### 9.6.1 Generating Arithmetic Progressions For the Column of the Next Move

Column 0 is partitioned by standard parity into two arithmetic progressions. Even parities are acted upon by the *DV2* operator. Odd parities are acted upon by *TP1*. In more general terms the  $m$  parity sequence of the seed column can be partitioned into two types  $m$  parity sequences: The odd parities manifest as  $cs^0$ , while the even parities manifest as  $rhs^1$ . (This is listed in Table 2. The key to explaining this table is in Table 1. Here, for a given row, in the last column, resides an abridged height sequence with its associated abridged  $m$  parity sequence. The respective shorthand notation is in the same row of the first column.)

Now to focus on the  $m$  parities where  $m > 0$  (the even parities). These will be  $m - 1$  parities in the next move (by the *DV2* operation). The  $rhs^1$ , acted upon by *DV2*, becomes  $rhs^0$  (since division by 2 decreases the order of the ruler harmonic sequence by 1, refer to Theorem 6.1 and equation 28).

What is less straight forward is what happens to the 0 parities (the odd parities), in the next move. Observe the 0 parities in column 0 (in the right diagram of Figure 11). The *TP1* operator multiplies the common difference of an arithmetic progression by three (refer to equation 31. Effectively, after *TP1* the resulting arithmetic progression of  $2 \cdot \mathbb{N}_3$  becomes  $6 \cdot \mathbb{N}_{10}$ . Also, the *TP1* operator adds 1 to each element of an arithmetic progression and in the process toggling the odd parity sequence to an even parity sequence. The  $m$  parity sequence type is set to  $rhs^1$ . Since:

- i. These  $m$  parities of odd integers can be described in terms of  $L$  parities (look ahead parities, defined in Section E.1). How  $L$  parity sequences are generated is shown in Figure 30 and also in the orange bars in Figure 7. This last figure, along with its accompanying notes, illustrates graphically the effect of how choosing the common difference and origin, of an arithmetic progression, determines the  $m$  parity sequence type and order.
- ii. In general, partitioning every alternate element of an arithmetic progression results in two arithmetic progressions (refer to Lemma 9.1); their  $m$  parity sequence type is either  $rhs^k$  or  $cs^k$ . Which type depends on the origin of their input arithmetic progression and it's common difference. (Refer to Section 7.)

In what follows, excerpts from Figure 11 are used to form Table 1 which in turn is used to explain how Table 2 is built. The frequencies of  $cs^a$  and  $rhs^b$  for  $a, b \in \mathbb{N}_0$  for

each move of this Table 2, along the frequencies of later moves are summarized in Table 3 and Table 4.

Note that the flute lengths in the Cascade of  $m$  Parities Figure 11, are twice as long as in the Cascade of Heights and the Cascade of Offsets. This is because flute tessellation occurs half as often in the Cascade  $m$  Parities as in other cascades. The reason:  $m$  parities determine the manifestations of height and offset for the *next* move which is in a height or offset cascade flute that is twice as long as the previous flute. By design, the number of interleaving arithmetic progressions is doubled for each successive move (thereby doubling the flute length from move to move). This allows for all allowable possible permutations of standard parities, *ones phobic binaries*, to be used as inputs. The number of allowable distinct permutations is  $\leq 2^n$  (the cascade seed range), where  $n$  is the number of moves (or the path length of the cascade). Two odd parity moves are not allowed to follow each other. In this case, even parity moves supplant successive odd parity moves. This is illustrated in Figure 11. The actual number of distinct ones-phobic paths is  $\mathfrak{F}_{n+2}$  the  $(n+2)$ th Fibonacci number, refer to Table 10 and Appendix C. To summarize, the arithmetic progressions which are made from each phase shift congruent element from successive column-wise flutes in each column of Figure 11 are tabulated in Table 2.

How entries in Table 2 are derived can be shown by comparing the entries of move 2 from Table 2 with the Cascade of Heights and the Cascade of  $m$  Parities Figure 11. Here, the seed range, or row range, (the length of a flute for the Cascade of Heights and the Cascade of Offsets, but not the Cascade of  $m$  Parities), is  $\Delta r = 4$ .

In Table 1, the first column entry (of the top part), whose subscript 2 is the first phase shift of the top flute of the Cascade of Heights of Figure 11 (also the first element of the adjacent heights sub-sequence of the second column). The leading coefficient 6 is the first phase shift of the top flute of the Cascade of Offsets. The first column entry (of the bottom part), is the  $m$  parity sequence (also the first element of the adjacent  $m$  parity sub-sequence of the second column).

In the second column (of Table 1), the entries (of the top part), interleave to form the move 2 column of the Cascade of Heights of Figure 11. That is, the first element of the four sequences form the first four elements of the Cascade of Heights (move 2). The arrows in the Figure 11 point from an entry from a previous move. The arrows split to point two entries that are partitions of the previous entry. These partitions are interleaving sub-sequences of the sequence of the previous move. Then a succeeding arrow points to the entry of the next move (here the *TP1* or *DV2* operator has been applied).

The interleave approach is used to build the remaining columns of Table 2. That is, a cascade column is partitioned into interleaving sequences to generate the next column. Note how interleaving doubles the arithmetic progression common difference. Why? The arithmetic progression is a sequence of phase shift congruent heights. When forming an interleaved sequence, every alternating height forms one sub-sequence; while the remaining heights form the other. Essentially, an offset is skipped from each sub-sequence. This would cause sub-sequence heights to be less than what they should be. In order to compensate, for skipping each alternate offset, the offsets are doubled.

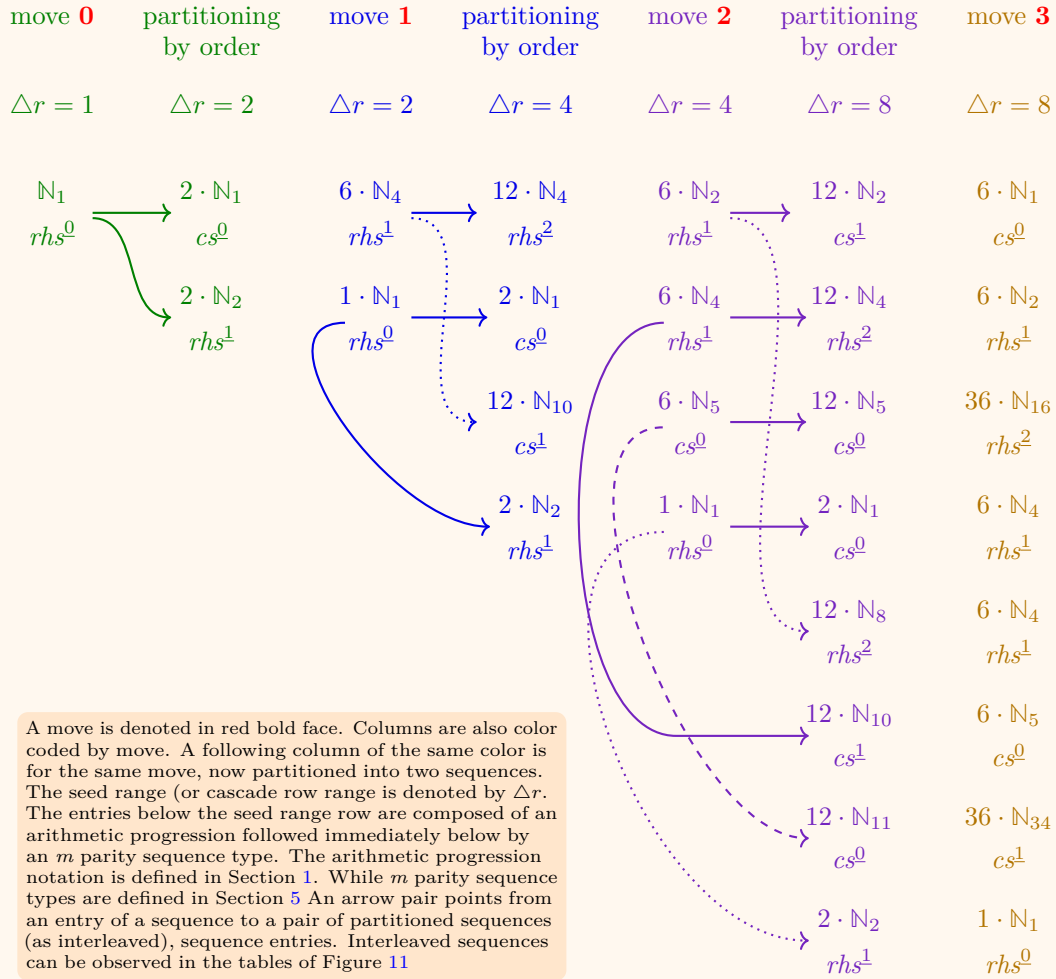
The doubled offsets are changed again (as shown by succeeding that point to the next move column). Either *TP1*, or *DV2*, changes the leading coefficient again.

Which operator used depends on the standard parity (of the sub-scripted entry of the preceding partition column). The operator adjusted offsets form the flute of the next move (in the Cascade of Offsets).

Table 1: Arithmetic Progression and  $m$  Parity Notations Alongside Their Expansions

move <b>2</b> $\triangle r = 4$	
arithmetic progression with its $m$ parity sequence	their expansion
$6 \cdot \mathbb{N}_2$ $rhs^1$	$2, 8, 14, 20, 26, \dots, 92$ $1, 3, 1, 2, 1, \dots, 1$
$6 \cdot \mathbb{N}_4$ $rhs^1$	$4, 10, 16, 22, 28, \dots, 94$ $2, 1, 4, 1, 2, \dots, 2$
$6 \cdot \mathbb{N}_5$ $cs^0$	$5, 11, 17, 23, 29, \dots, 95$ $0, 0, 0, 0, 0, \dots, 0$
$1 \cdot \mathbb{N}_1$ $rhs^0$	$1, 2, 3, 4, 5, \dots, 16$ $0, 1, 0, 2, 0, \dots, 5$

Table 2: Arithmetic Progressions With Their  $m$  Parity Sequences Within a Flute For a Given Move



Remark 5.1 is used in the following: Note, with regards to the approaching  $m$  parity frequency distribution for finitely large seed ranges: In move **0**, combining the occurrences of  $cs^{k-1}$  with the occurrences of  $rhs^k$ , yields the frequency of occurrence of  $rhs^{[k-1]}$ . In move **1**, combining the occurrences of  $cs^{k-1}$  with the occurrences of  $rhs^k$ , yields the frequency of occurrence of  $rhs^{[k-1]}$ .

**The key for the iterative forms below each table is as follows:**  
 For both tables:  $A_n$  is column 0,  $B_n$  is column 1,  $D_n$  refers to column 2 or a succeeding column,  $C_n$  refers to the column preceding  $D_n$ . While the iteration subscript  $n$  is also the exponent in  $2^n$  whose result is the range or the flute length, which resides in the cascade column of move  $n$ .

Table 3: CS Frequency Distribution of Each Order At a Given Move From a Cascade

CS Order (red text)											
	0	1	2	3	4	5	6	7	8	9	10
move											
0	0	0									
1	0	0									
2	1	0									
3	2	1	0								
4	5	3	0								
5	10	8	1	0							
6	21	18	4	0							
7	42	39	12	1	0						
8	85	81	30	5	0						
9	170	166	69	17	1	0					
10	341	336	150	47	6	0					
11	682	677	316	116	23	1	0				
12	1365	1359	652	266	70	7	0				
13	2730	2724	1329	582	186	30	1	0			
14	5461	5454	2688	1234	452	100	8	0			
15	10922	10915	5412	2563	1034	286	38	1	0		
16	21845	21837	10866	5251	2268	738	138	9	0		
17	43690	43682	21781	10663	4831	1772	424	47	1	0	
18	87381	87372	43618	21529	10082	4040	1162	185	10	0	
19	174762	174753	87300	43310	20745	8871	2934	609	57	1	0
20	349525	349515	17467	86928	42274	18953	6974	1771	242	11	0

Iterative form:  $A_n = 2A_{n-1} + (n \bmod 2)$ ,  $B_n = B_{n-1} + A_{n-1}$ ,  $D_n = D_{n-1} + C_{n-2}$

Table 4: RHS Frequency Distribution of Each Order At a Given Move From a Cascade

RHS Order (red text)												
	0	1	2	3	4	5	6	7	8	9	10	11
move												
0	1	0										
1	1	1	0									
2	1	2	0									
3	1	3	1	0								
4	1	4	3	0								
5	1	5	6	1	0							
6	1	6	10	4	0							
7	1	7	15	10	1	0						
8	1	8	21	20	5	0						
9	1	9	28	35	15	1	0					
10	1	10	36	56	35	6	0					
11	1	11	45	84	70	21	1	0				
12	1	12	55	120	126	56	7	0				
13	1	13	66	165	210	126	28	1	0			
14	1	14	78	220	330	252	84	8	0			
15	1	15	91	286	495	462	210	36	1	0		
16	1	16	105	364	715	792	462	120	9	0		
17	1	17	120	455	1001	1287	924	330	45	1	0	
18	1	18	136	560	1365	2002	1716	792	165	10	0	
19	1	19	153	680	1820	3003	3003	1716	495	55	1	0
20	1	20	171	816	2380	4368	5005	3432	1287	220	11	0

(Pascals Triangle Left Justified With Succeeding Columns Shifted Down One Cell)

Iterative form:  $A_n = 1$ ,  $B_n = n$ ,  $D_n = D_{n-1} + C_{n-2}$

## 9.7 Generalized Evaluation of Offsets

### 9.7.1 Generating Offsets for Flutes at the Column of the First Move

Note: In the calculus of finite differences  $\Delta k = 0$ , where  $k$  is a constant (no change in a constant).

First, equations are generated for the first flute (column 1). The range is  $\Delta r = 2^{move} = 2^1 = 2$ . This is the length of a flute by Definition 8.3. Next, proceed from seed values to the column holding the first move.

If the input is even, apply the *DV2* rule to obtain the offset:

$$\Delta \left( \frac{r}{2} \right) = \frac{1}{2} (\Delta r) = \frac{1}{2} (2) = 1 \quad (32)$$

If the input is odd, apply the *TP1* rule to obtain the offset:

$$\Delta (3r + 1) = 3\Delta r + \Delta 1 = 3(2) + 0 = 6 \quad (33)$$

Note: Successive values in the seed column increment by 1 (by design). Successive values in column 1 in figure 8a increment by the alternate offsets of 1 or 6. Results from equations (32) and (33) are the offsets for the first flute.

### 9.7.2 Generating Offsets for Flutes for the Column of the Second Move

Continuing in a similar fashion to obtain the general equations for the flute of the second move column. The range  $2^{move} = 2^2$  (labeled as 2). using (25) in composition:

$$\Delta(\Delta r) = \Delta^2 r = 2^2 = 4 \quad (34)$$

Since, an odd move cannot lead to an odd move. (By the rationale of Section 3.1 which is by design from equation 1). Then, in general, there are three possible outcomes. However, there are four possible inputs for  $\Delta^2 r = 4$ , two even and two odd. The even and odd heights alternate (as a consequence of the two flutes of the previous column that are used as input (they contain the preceding moves), to a flute in the current column.

Equations 32 and 33 can be reused through composition. Then equations for the second move (column labeled 2 in figure 8a) are:

Even height followed by even height,

$$\Delta \frac{\left( \frac{\Delta r}{2} \right)}{2} = \frac{\frac{\Delta^2 r}{2}}{2} = \frac{\frac{4}{2}}{2} = 1 \quad (35)$$

Odd height followed by even height,

$$\Delta \left( 3 \frac{\Delta r}{2} + 1 \right) = \frac{3}{2} \Delta^2 r + \Delta 1 = \frac{3}{2}(4) + 0 = 6 \quad (36)$$

Even height followed by odd height,

$$\Delta \frac{(3\Delta r + 1)}{2} = \frac{3}{2} \Delta^2 r + \Delta \frac{1}{2} = \frac{3}{2}(4) + 0 = 6 \quad (37)$$

Generating Equations for the Flute of Column 2. equations in this order: (35), (36), (37), and again (36); generate the offsets that are demarcated by the next larger flute. there are an equal number of even and odd first stage inputs (seed values). that is why (36) repeats.

### 9.7.3 Repeated Compositions (Generating an Offset for a Flute of Size $2^n$ )

More generally for repeated compositions of (25), we define the difference between values whose difference is a power of two, as:

$$\Delta(\Delta(\Delta(\dots r \dots))) \triangleq \Delta^n r = 2^n \quad (38)$$

For repeated compositions of (33). The 3 prefix is kept as a repeat multiplier,  $a$  times, for repeated compositions (applications), by the  $\Delta^a$  operator. The isolated  $+1$  term remains constant, consequently, the  $\Delta$  of a constant is zero, (no change in a constant). Let  $a$  = number of odd compositions, using (38) yields:

$$\Delta^a (3r + 1) = \Delta^a(3r) + \Delta^a 1 = 3^a \Delta^a r + 0 = 3^a \Delta^a r \quad (39)$$

For repeated compositions of (32). The 2 is kept as a divisor by the  $\Delta$  operator. Let  $b$  = number of even compositions, correspondingly:

$$\Delta^b \left( \frac{r}{2} \right) = \frac{1}{2^b} \Delta^b r \quad (40)$$

Combining (39) and (40) for an arbitrary fluctuating path, through repeated compositions.

$$\frac{3^a}{2^b} (\Delta^b r^{a+b}) = \frac{3^a}{2^b} (2^{a+b}) = 6^a \quad (41)$$

To illustrate with an example (which is checked in Section 9.7.4). Begin with the table coordinate (column labeled 5, row labeled 87) of figure 8a. The height is 592. Proceed to the corresponding position at the next lower flute located

directly below. (Note: The fifth column after the seed column is being used.) Thereby, we let  $n = 5$  in equation (38). The seed range becomes  $\triangle^5 r = 2^5$ . The coordinates are: (column labeled 5, row labeled 119). Here the height is 808. The offset between these coordinates is,

$$808 - 592 = 216 = 6^3 \quad (42)$$

Summarizing with a conclusion to this section.

*Remark 9.3* (All Offsets Are a Power of Six and They Tessellate Within Flutes). The result of equation 41 demonstrates that all offsets are a power of six.

The ordered offsets of a flute, of a given length  $2^n$ , can be evaluated (developed in Section 8.1 up to and including Section 9.7.3). As a consequence, the offsets tessellate (as a collection among column-wise flutes which tile end to end).

#### 9.7.4 Introducing the Ones-Phobic Binary

To further distinguish a binary from a ones-phobic binary the following symbols are used:  $\bullet$  for 1 and  $\circ$  for 0. In a cascade, for a given path following a seed, encode odd values (or 'rises'), as  $\bullet$ . Encode even values (or 'falls'), as  $\circ$ . Ones-phobic binary symbols are used in figure 8. Also, to further differentiate patterns, differently shaded squares are used (as in 18).<sup>4</sup>

Note that no two ones are next to each other. This due to the fact that *TP1* always produces an even result, explained in Section 3.1.

Ones-phobic binaries are displayed in figure 9, figure 10, figure 16, and are discussed in appendix D.

Excluding the header, call a row of figure 2, a *ones phobic binary*, which is denoted by  $\mathbb{P}_n$ . Where,  $\mathbb{P}_n$  refers to a ones-phobic binary consisting of  $n$  bits. That is  $n$ , the number of bits, corresponds to the number of moves, or path length, of a cascade row (or part of a row); this includes the seed column. To more readily evaluate the offset at a given cell in a cascade re-implement equation (41). That is, the result in (41) is determined by the exponent  $a$ . This is the number of *TP1* operations (rises or  $\bullet$ 's in  $\mathbb{P}_n$ ), which re-implements as:

$$\mathbb{O}(\mathbb{P}_n) = 6^{\sum \mathbb{P}_n} \quad (43)$$

By design, the  $n$ , in  $\mathbb{P}_n$ , designates the columnar position of the offset. These results, using equation (43), are shown in figure 8b. By definition,  $\sum \mathbb{P}_n$  is the sum of the bits in  $\mathbb{P}_n$ . Example:

Using the path, of five moves, starting with the seed value, in the row labeled 87 up to the column labeled 5.

---

<sup>4</sup>This allows for easier visual pattern recognition.

Here,

$$\mathbb{P}_5 = 10101 \quad (44)$$

substituting in,

$$\mathbb{O}(\mathbb{P}_5) = 6^{\sum \mathbb{P}_5} = 6^3 = 216 \quad (45)$$

The result of (45) checks with (42).

## 9.8 Relating an Offset Cascade to a Ones-Phobic Cascade

The generalization of an arbitrary row from the left sub-figure of figure 9 in terms of powers of 6, looks something like:

$$6^n \quad 6^n \quad \dots \quad 6^{n+1} \quad 6^{n+1} \quad \dots \quad 6^{n+2} \quad 6^{n+2} \quad \dots \quad (46)$$

The focus is now on a subsequence of offsets with the same exponent in (46). Then all have the same offset. However, the offset of the next move (in the next column to the right), is incremented half as often as the offset in the column of the previous move (demonstrated in Section 8.1). In a cascade, a path height is determined by it's associated offset and the number of times that offset is incremented. This increment frequency of the offset is determined by the position of the containing flute, relative to other flutes, within that column. This is more readily observed (and explained), by delineating a column with flutes. recall the definition of a flute in Section 8.3. A *flute* contains consecutive offsets in a sub-column whose length is a power of two. The length requirement ensures that a *flute* will tessellate (repeat), along the entire length of the column of the full cascade. That is, for a given *flute*, another *flute* below (or above, if it exists), in the same column, has the same pattern of offsets. (This is observed, column-wise, by the *flutes* in 8a. The general case is covered in Remark 9.3.) This tessellation is illustrated in Figure 28 and Figure 27

For an arbitrary selected column, *flutes* in the neighbouring column to the right are twice as long as those in the selected column (by definition, in Section 8.3). This means that, at a given row, an offset occurs twice as often as that of the neighbouring rightward offset. The offset frequency can be used to determine inequality relationships between row-wise adjacent pairs of offsets.

Initial heights (those prior to offset addition), set the direction of the inequality. Refer to equations 47 and 49. Initial heights are less than the offset that is added to them. Otherwise, they would contain an offset (that was added previously). The effect of the initial heights, on the inequality relation, decreases by repeated addition of offsets along with the offset magnitude (determined by  $n$  in  $6^n$ ). The initial heights remain constant; by themselves they do not alter the inequality relationship of heights between row-wise adjacent pairs. (To avoid obfuscation, the initial heights are not shown below in equations 47 and 49.)

## 9.9 Offset Pair With the Same Exponent

Let a pair of heights, in the same row and in adjacent columns, be expressed as offsets (from an offset cascade). Let the leading height, of the pair, be larger than the trailing height (heights are descending). These heights can be factored into a power of six and another factor  $k$ . Where  $k$  is the offset increment count. A flute in the column of a following move is twice as long as a flute in the previous column. Consequently,  $k$  is half as large in the following move. In general, let's focus on a row-wise contiguous pair of offsets. Looking at a pair of offsets that are the same.

$$k6^n > \frac{k}{2}6^n \quad (47)$$

## 9.10 Offset Pair With Different Exponents

Now let's focus on an offset pair, contiguous along a row, where the latter offset has a factor being a higher exponent.

$$k6^n < \frac{k}{2}(6^{n+1}) \quad (48)$$

$$k6^n < 3k6^n \quad (49)$$

Note to summarize: As evident in equations (47) and (49), order relations of are preserved by successive additions of offsets. Since in equation (47), the left hand side is greater than the right hand side. A falling move has occurred. then, by equation 1, the leading height is even. Let an even height be represented by 0. Conversely in (49), the left-hand side is less than the right-hand side. A rising move has occurred (which is represented by 1). In this way a ones-phobic cascade can be directly derived from an offset cascade (without referring to heights directly).

Consequently, each offset in a full cascade can be replaced with it's ones-phobic counterpart. Seed values and columnar headers (as powers of two) are used as coordinate reference points. The result is in the right sub-figure of figure 9.

## 10 Frequency Effects on Ones-Phobic Binaries

Begin with a ones-phobic binary of length  $k$  within a  $2^k$  seed range. Section 4.1 and appendix Section E.2 are an introduction to the rest of this section.

## 10.1 The Effect of Prepending a Zero

Prepend a zero bit to a ones-phobic binary to extend the length from  $k$  to  $k + 1$ . The leading parity of the ones-phobic binary changes from  $m$  to  $(m + 1)$ . Because of its length the resulting ones-phobic binary is in  $\mathbb{B}_{(k+1)}$ . According to equation (2),  $\mathbb{B}_{(k+1)}$  has a seed range twice as large as  $\mathbb{B}_k$ . Ordinarily this would double the frequency of occurrence. since, appending a zero increases the parity, from  $m$  to  $(m + 1)$ . However,  $(m+1)$  parity occurs half as often as  $m$  parity, (refer to 4.1). this cancels out the seed range doubling. thus, the frequency stays the same.

## 10.2 The Effect of Prepending a One

Prepend a 1 bit to a ones-phobic binary to extend the length from  $k$  to  $k + 1$ . This can only occur, in those paths where the pending 1 will precede as 0. (So as not to violate the defining characteristic of a ones-phobic path). The leading parity of the ones-phobic binary changes from  $m$  to  $l$ . However by the defining property of  $l$  parity, has  $m$  consecutive trailing zeroes (refer to appendix E.2). So:

$$m = l$$

That is, the parity stays the same. Consequently, the frequency of occurrence stays the same. However, as in Section 2, the seed range doubles, thereby increasing the rate of incidence of the given  $l$  parity. Doubling the seed range while maintaining the same rate of occurrence, doubles the frequency.

## 10.3 Starting With the Least Significant Bit

Initially, no bit is pre-pended to an existing binary. To begin a ones-phobic binary, the  $m$  parity of the seed values are used. The seed values are either even or odd. So correspondingly, the  $m$  parity is set. There is no change in frequency due to the least significant bit.

Generalize the dimensions of cascade as: path length  $n$  and seed range  $2^n$ . A formula for the frequency of occurrence of  $\mathbb{P}_n$  in a cascade becomes:

$$f(\mathbb{P}_n) = 2^{\sum \mathbb{P}_{(n-1)}} \quad (50)$$

Note: For the, left to right, indexing convention, 1 is the most significant bit,  $n$  is the least significant bit. Essentially, sum the bits of, a given ones-phobic binary ( $\mathbb{P}_n$  in  $\mathbb{N}_n$ , where  $\mathbb{N}_n$  is the seed range  $2^n$ ). Excluding the least significant bit in the summation.

Example:

Using the same path as in the last example, of five moves, starting with the seed value, in the row labeled 87 up to the column labeled 5. here,

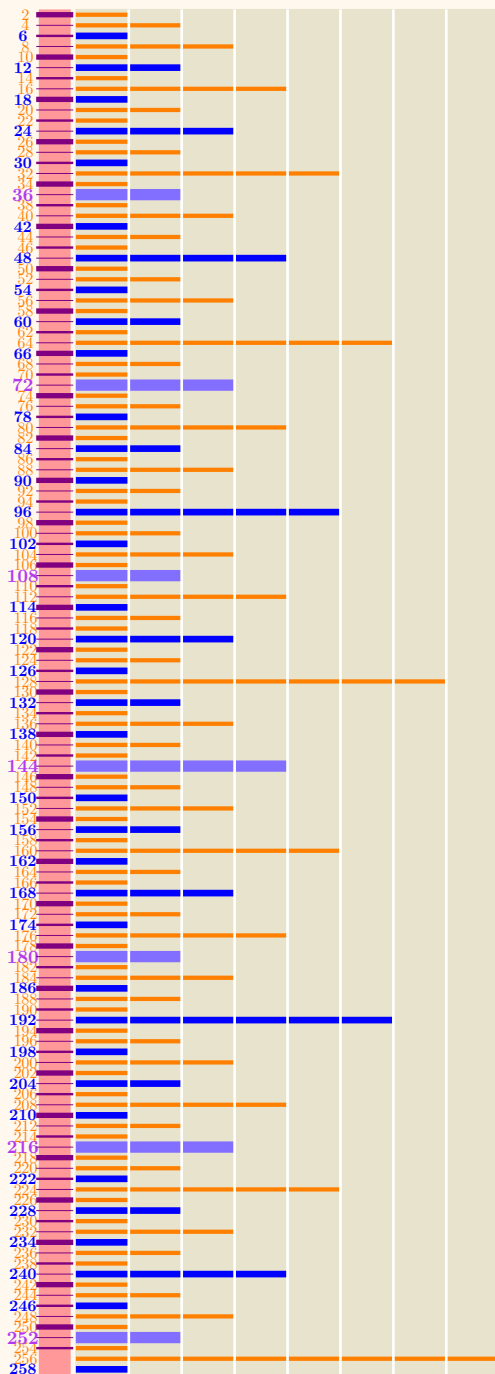
$$\mathbb{P}_5 = 10101 \quad (51)$$

exclude the last bit,

$$\mathbb{P}_4 = 1010 \quad (52)$$

substituting in:

$$f(\mathbb{P}_5) = 2^{\sum \mathbb{P}_4} = 2^2 = 4 \quad (53)$$



#### Illustrating ruler harmonics with a bar chart:

We have two rulers facing each other. One is pastel coral (on the left), the other pastel burlywood (on the right). The scale marks of both rulers line up with a column of numbers located left of both rulers. These numbers represent index values. The end to end horizontal segmented lines are ruler scale marks (of the right ruler). The number of end to end bars represent the  $m$  parity of the aligning index value. The term **ruler harmonics** describes this sequence of  $m$  parities. With regard to the index values: The orange bars are multiples of two. The thicker blue bars are multiples of 6. The thickest bars, slate blue, are multiples of 36. Each multiple, of the powers of six shown, forms it's own ruler. Where each ruler is indicated by a colour of it's own. The red line segments crossing over the coral ruler are there to delineate landings. The length of the landing regions, on the coral ruler, are a power of two. Note the ordering of the scale marks of the burlywood ruler. Observe the specific end to end lengths (the  $m$  parity), with regards to a landing. Note how this ordering repeats down along successive landings. That is, the  $n$ th **placement** within a landing will have the same  $m$  parity as the same placement within another landing. With one exception, the maximum  $m$  parity within a landing will not be a constant, it will exhibit **ruler subharmonics**. Note, the odd index values are omitted from the illustration. Their  $m$  parity is zero. As such, their scale marks are of length zero, and would be shown as blank spaces. The composite of all the scale marks illustrates a ruler harmonic sequence of order 1. The blue scale marks illustrate a ruler harmonic sequence of order 1. The thickest scale marks (slate blue) illustrate a ruler harmonic sequence of order 2. The magnitude of the order is the value of the smallest element in the series.

Figure 7: Ruler Harmonics The First Few Powers Of Six

Figure 8: Comparison of Heights With Offset Encoding

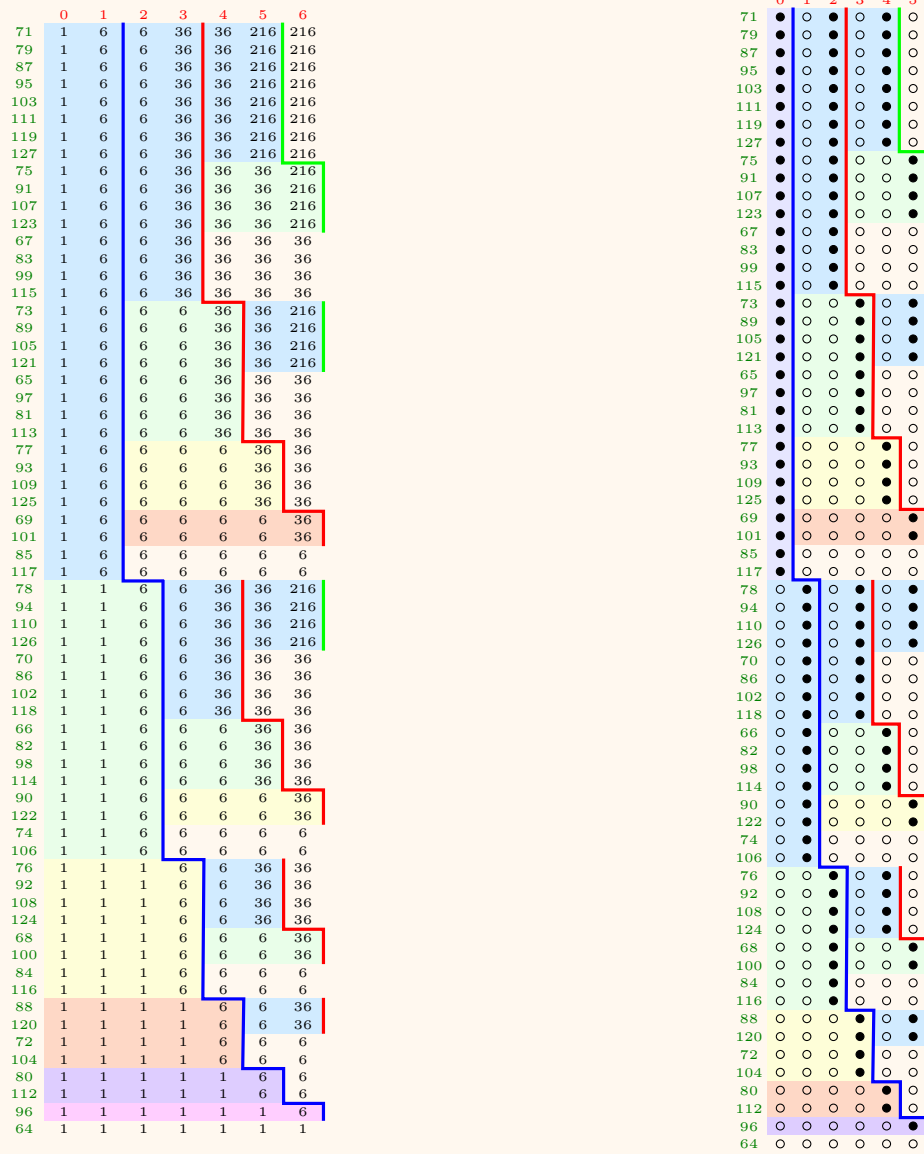
0	1	2	3	4	5	6
64	32	16	8	4	2	1
65	196	98	49	148	74	37
66	33	100	50	25	76	38
67	202	101	304	152	76	38
68	34	17	52	26	13	40
69	208	104	52	26	13	40
70	35	106	53	160	80	40
71	214	107	322	161	484	242
72	36	18	9	28	14	7
73	220	110	55	166	83	250
74	37	112	56	28	14	7
75	226	113	340	170	85	256
76	38	19	58	29	88	44
77	232	116	58	29	88	44
78	39	118	59	178	89	268
79	238	119	358	179	538	269
80	40	20	10	5	16	8
81	244	122	61	184	92	46
82	41	124	62	31	94	47
83	250	125	376	188	94	47
84	42	21	64	32	16	8
85	256	128	64	32	16	8
86	43	130	65	196	98	49
87	262	131	394	197	592	296
88	44	22	11	34	17	52
89	268	134	67	202	101	304
90	45	136	68	34	17	52
91	274	137	412	206	103	310
92	46	23	70	35	106	53
93	280	140	70	35	106	53
94	47	142	71	214	107	322
95	286	143	430	215	646	323
96	48	24	12	6	3	10
97	292	146	73	220	110	55
98	49	148	74	37	112	56
99	298	149	448	224	112	56
100	50	25	76	38	19	58
101	304	152	76	38	19	58
102	51	154	77	232	116	58
103	310	155	466	233	700	350
104	52	26	13	40	20	10
105	316	158	79	238	119	358
106	53	160	80	40	20	10
107	322	161	484	242	121	364
108	54	27	82	41	124	62
109	328	164	82	41	124	62
110	55	166	83	250	125	376
111	334	167	502	251	754	377
112	56	28	14	7	22	11
113	340	170	85	256	128	64
114	57	172	86	43	130	65
115	346	173	520	260	130	65
116	58	29	88	44	22	11
117	352	176	88	44	22	11
118	59	178	89	268	134	67
119	358	179	538	269	808	404
120	60	30	15	46	23	70
121	364	182	91	274	137	412
122	61	184	92	46	23	70
123	370	185	556	278	139	418
124	62	31	94	47	142	71
125	376	188	94	47	142	71
126	63	190	95	286	143	430
127	382	191	574	287	862	431
128	64	32	16	8	4	2

0	1	2	3	4	5	6
1	1	1	1	1	1	1
1	6	6	6	36	36	36
1	1	6	6	6	36	36
1	6	6	36	36	36	36
1	1	1	6	6	6	36
1	6	6	6	6	6	36
1	1	6	6	36	36	36
1	6	6	36	36	216	216
1	1	1	1	6	6	6
1	6	6	6	36	36	216
1	1	6	6	6	6	6
1	6	6	36	36	36	216
1	1	1	6	6	36	36
1	6	6	6	6	36	36
1	1	6	6	6	36	36
1	6	6	36	36	216	216
1	1	1	1	1	6	6
1	6	6	6	36	36	36
1	1	6	6	6	36	36
1	6	6	6	6	36	36
1	1	6	6	36	36	216
1	6	6	36	36	216	216
1	1	1	1	1	1	6
1	6	6	6	36	36	36
1	1	6	6	6	36	36
1	6	6	36	36	36	36
1	1	1	6	6	6	6
1	6	6	6	6	36	36
1	1	6	6	36	36	216
1	6	6	36	36	216	216
1	1	1	1	1	1	6
1	6	6	6	36	36	36
1	1	6	6	6	36	36
1	6	6	36	36	36	36
1	1	1	6	6	6	6
1	6	6	6	6	36	36
1	1	6	6	36	36	216
1	6	6	36	36	216	216
1	1	1	1	1	1	6
1	6	6	6	36	36	36
1	1	6	6	6	36	36
1	6	6	36	36	36	36
1	1	1	6	6	6	6
1	6	6	6	6	36	36
1	1	6	6	36	36	216
1	6	6	36	36	216	216
1	1	1	1	1	1	1

(a) Height Cascade Showing Sink Values (Highlighted Cells) and Pan Flute Template (Outlined by Thicker Borders)

(b) Offset Cascade Showing Downward Flute Tesilation

Figure 9: Quilt Pattern in Offset, and Binary Encoded, Cascade

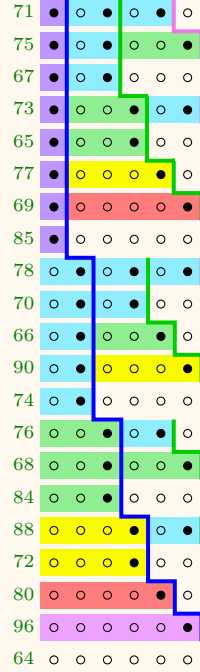
**Sorted Offset Cascade (Colour Coded by Parity)**

Rectangular regions are colour coded according to a horizontal run of constant offsets. The run ends at the first occurrence of an increase in offset. Rectangles of the same width have the same colour. Seed value offsets are shown but not colour coded. Neither are rectangles truncated at the cascade's right end.

Generations are delineated by the the outline of a staircase shape border. The first generation has a trailing blue border. The second generation trailing border is red. The third is green.

**Binary Encoding of Offset Cascade** The lower offset in a colour block is encoded as ○, while the higher offset is encoded as ●. (Refer to Section 9.10)

Figure 10: A Mosaic (All Unique Ones-Phobic Binary Paths From a Cascade)



### Sorted Ones-Phobic Binary Mosaic (Colour Coded By Offset)

A mosaic is a subset of a cascade. Specifically, a mosaic only contains all cascade paths having a unique ones-phobic encoding. In a quilt, rectangular regions are colour coded according to a horizontal run of equal offsets. Alternatively, the same effective colour coding can be assigned according to the highest  $m$  parity (at the rectangles leading edge). Instead of the entire quilt rectangle, the focus becomes on a single row of the rectangle, more specifically, a [parity slide](#) of a ones-phobic path. In the diagram of Table 10 these ones-phobic binary parity slide are illustrated with o's and a trailing • (to distinguish them from ordinary binaries). Since they are reminiscent of (LEGO™ bricks) with a  $1 \times n$  stud arrangement. A parity slide of length  $n$  is referred to as an [n brick](#).

Seed value offsets are shown in green text. Bricks truncated at the mosaic's right end are not color coded. A fall is encoded as o, while a rise is encoded as •. (Refer to Section 9.10)

Figure 11: Cascade of Heights With it's Corresponding Cascade of Offsets and Cascade of  $m$  Parities

Columns with green numbers are seed values. The number of moves to a column are shown by column headings in red.

Within a cascade, each column has elements outlined by successive column-wise flutes. Within a column **0** of **1**, flute lengths are drawn to be as small as possible so that tessellation is still shown. Flute lengths are the same in each column. Flute lengths double in length with each next move. Flute borders are color coded by column to better distinguish flutes of a given size. Those cascades containing manifestations of offsets or standard parity, tessellate end to end downwards.

Colour coded flute boundaries are used to more readily compare cascades, namely those of: height, offset, and  $m$  parity. In the  $m$  parity cascade, a broken line of two dashes, inwardly facing, divide each flute in half. These *half flutes* are used as a reference guide when comparing the contents among the flute boundaries between height cascades and offset cascades are compared with the flutes in the  $m$  Parity Cascade. In a *half flute*, only the shaded cells tessellate from one *half flute* to the next. (The unshaded cells are *traversals*.)

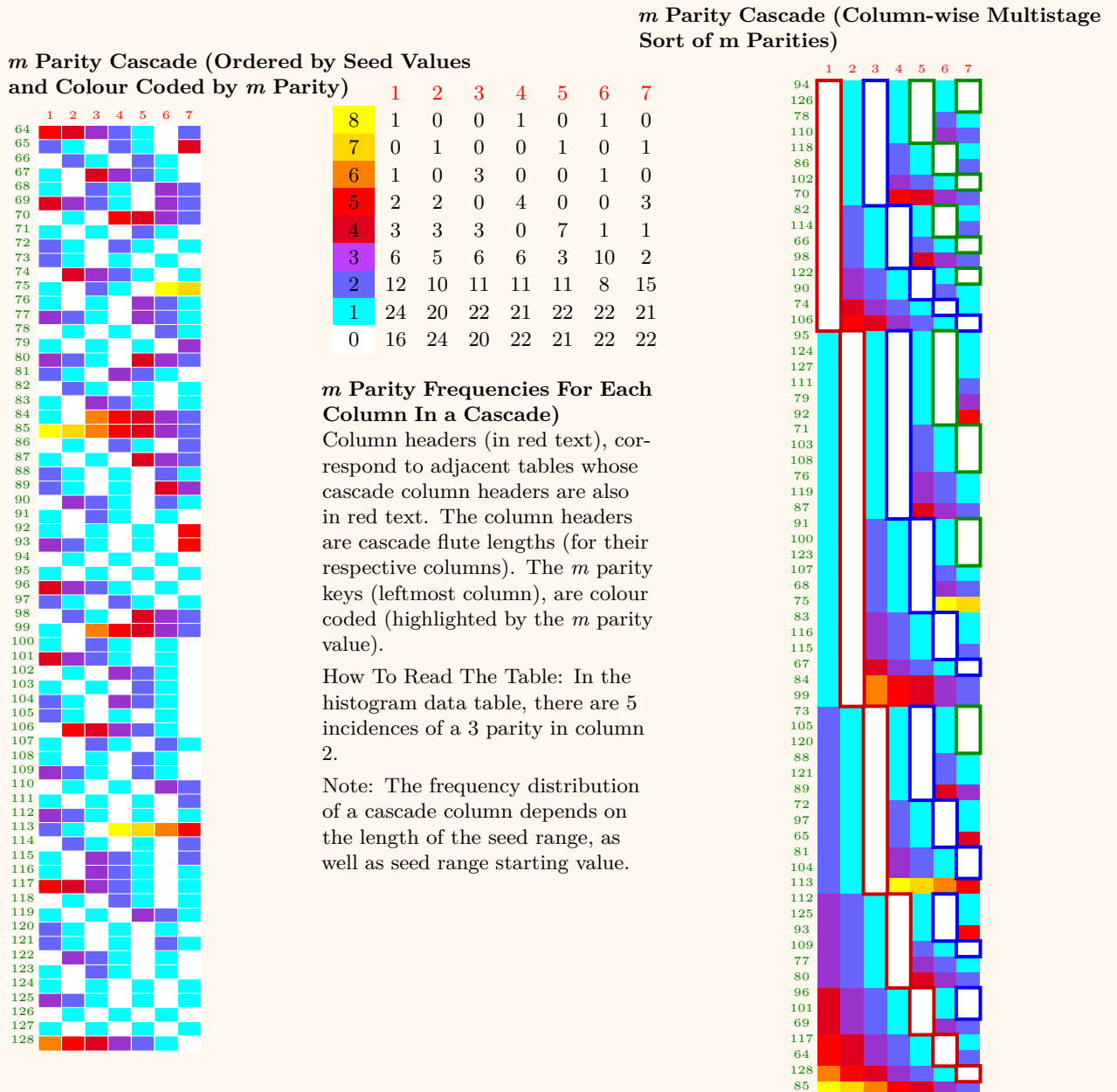
Cascade of Heights				
0	1	2	3	4
1	4	2	1	4
2	1	4	2	1
3	10	5	16	8
4	2	1	4	2
5	16	8	4	2
6	3	10	5	16
7	22	11	34	17
8	4	2	1	4
9	28	14	7	22
10	5	16	8	4
11	34	17	52	26
12	6	3	10	5
13	40	20	10	5
14	7	22	11	34
15	46	23	70	35
16	8	4	2	1
17	52	26	13	40
18	9	28	14	7
19	58	29	88	44
20	10	5	16	8
21	64	32	16	8
22	11	34	17	52
23	70	35	106	53
24	12	6	3	10
25	76	38	19	58
26	13	40	20	10
27	82	41	124	62
28	14	7	22	11
29	88	44	22	11
30	15	46	23	70
31	94	47	142	71
32	16	8	4	2
33	100	50	25	76
34	17	52	26	13
35	106	53	160	80
36	18	9	28	14
37	112	56	28	14
38	19	58	29	88
39	118	59	178	89
40	20	10	5	16
41	124	62	31	94
42	21	64	32	16
43	130	65	196	98
44	22	11	34	17
45	136	68	34	17
46	23	70	35	106
47	142	71	214	107
48	24	12	6	3
49	148	74	37	112
50	25	76	38	19
51	154	77	232	116
52	26	13	40	20
53	160	80	40	20
54	27	82	41	124
55	166	83	250	125
56	28	14	7	22
57	172	86	43	130
58	29	88	44	22
59	178	89	268	134
60	30	15	46	23
61	184	92	46	23
62	31	94	47	142
63	190	95	286	143
64	32	16	8	4

Cascade of Offsets				
0	1	2	3	4
1	1	6	6	36
2	1	1	6	6
3	1	6	6	36
4	1	1	1	6
5	1	6	6	6
6	1	1	6	36
7	1	6	6	36
8	1	1	1	6
9	1	6	6	36
10	1	1	6	6
11	1	6	6	36
12	1	1	1	6
13	1	6	6	6
14	1	1	6	36
15	1	6	6	36
16	1	1	1	1
17	1	6	6	36
18	1	1	6	6
19	1	6	6	36
20	1	1	1	6
21	1	6	6	6
22	1	1	6	36
23	1	6	6	36
24	1	1	1	6
25	1	6	6	36
26	1	1	6	6
27	1	6	6	36
28	1	1	1	6
29	1	6	6	6
30	1	1	6	36
31	1	6	6	36
32	1	1	1	1
33	1	6	6	36
34	1	1	6	6
35	1	6	6	36
36	1	1	1	6
37	1	6	6	6
38	1	1	6	36
39	1	6	6	36
40	1	1	1	6
41	1	6	6	36
42	1	1	6	6
43	1	6	6	36
44	1	1	1	6
45	1	6	6	6
46	1	1	6	36
47	1	6	6	36
48	1	1	1	1
49	1	6	6	36
50	1	1	6	6
51	1	6	6	36
52	1	1	1	6
53	1	6	6	6
54	1	1	6	36
55	1	6	6	36
56	1	1	1	6
57	1	6	6	36
58	1	1	6	6
59	1	6	6	36
60	1	1	1	6
61	1	6	6	6
62	1	1	6	36
63	1	6	6	36
64	1	1	1	1

Cascade of $m$ Parities				
0	1	2	3	4
1	0	2	1	0
2	1	0	2	1
3	0	1	0	4
4	2	1	0	2
5	0	4	3	2
6	1	0	1	0
7	0	1	0	1
8	3	2	1	0
9	0	2	1	0
10	1	0	4	3
11	0	1	0	2
12	2	1	0	1
13	0	3	2	1
14	1	0	1	0
15	0	1	0	1
16	4	3	2	1
17	0	2	1	0
18	1	0	2	1
19	0	1	0	3
20	2	1	0	4
21	0	6	5	4
22	1	0	1	0
23	0	1	0	1
24	3	2	1	0
25	0	2	1	0
26	1	0	3	2
27	0	1	0	2
28	2	1	0	1
29	0	3	2	1
30	1	0	1	0
31	0	1	0	1
32	5	4	3	2
33	0	2	1	0
34	1	0	2	1
35	0	1	0	5
36	2	1	0	2
37	0	4	3	2
38	1	0	1	0
39	0	1	0	1
40	3	2	1	0
41	0	2	1	0
42	1	0	6	5
43	0	1	0	2
44	2	1	0	1
45	0	3	2	1
46	1	0	1	0
47	0	1	0	1
48	4	3	2	1
49	0	2	1	0
50	1	0	2	1
51	0	1	0	3
52	2	1	0	3
53	0	5	4	3
54	1	0	1	0
55	0	1	0	1
56	3	2	1	0
57	0	2	1	0
58	1	0	3	2
59	0	1	0	2
60	2	1	0	1
61	0	3	2	1
62	1	0	1	0
63	0	1	0	1
64	6	5	4	3

Figure 12: Sorted and Unsorted  $m$  Parity Cascade, After a MultiStage Sort a Quilt Pattern is Revealed)

Seed values are shown in green text. Columns are labeled by successive powers of 2 (as in Figure 8), and are shown in red text.



## 11 Demonstrating the Existence of a Tier

Here, a structure called a tier is defined. This is further illustrated by building a tier. Then ruler harmonics are demonstrated, in the column(s) immediately following the tier.

**Definition 11.1** (*phase shift, and phase shift congruent path*). A [phase shift](#) refers to a move at the  $y$ th columnar position, from the top of a [flute](#). A path traversing flutes (along a cascade row), has a phase shift at each move; since each move is in a flute of it's own. Only along a cascade column, an  $n$ th move is [phase shift congruent](#) to another  $n$ th move (from another path), if both moves have the same phase shift with respect to the flute they are in. A path is phase shift congruent to another path when *each*  $n$ th move has an identical phase shift that corresponds to each phase shift of an  $n$ th move in that other path.

**Definition 11.2** (*Tier and Tier Paths*). A [tier](#), consists of successive *phase shift congruent* paths, selected from a cascade. Since a *tier* contains paths, equidistant to its neighbouring paths (same  $\Delta r$ ), all of the same fixed length (by design). Moreover, this coherent region of each tier path has the same number of identically ordered rises and falls (illustrated in Figure 14). Each path encodes the same ones-phobic binary of length  $n$  (shown in the coherence region of Figure 8).

A collection of tier paths will always have a ruler harmonic region. Since by design, each move (by column), in the tier path collection will have a unique offset ( $6^n$ ), associated with it. The coherent region contains columns of *CS* (clamped *rhs<sup>n</sup>*). When advancing to the column of the next move,  $n$ , the sequence order, incrementally decreases by 1 until successive sequences become un-clamped to form the ruler harmonic region. This is elaborated in Section 7.4.1. More on regions is covered in Section 12

### 11.1 Building a Tier

From a cascade consisting of  $2^n$  paths of length  $n$ , paths are selected to form an arithmetic progression from seed values whose common difference is  $\Delta r = 2^k$ , such that  $k < n$ . Begin at the longest flute of the Kuvysti (pan flute, illustrated in 8a). Then  $k$  the length of the longest flute is also the span (or range) of the tier that is built from the selected paths. The span (or range) of the tier is the number of paths in a tier. The reason for choosing paths, in this manner, is to ensure there are congruent phase shifts for every move of the paths selected.

That is: A flute of the column for the  $k$ th move aligns with the top of two consecutive shorter flutes for the column holding the  $(k - 1)$ th move. A pair of shorter flutes align with a flute in the next move which is twice as long. This alignment occurs recursively from the column of the first move to a given later column. Select a path for a tier (a row of a cascade), by choosing a phase shift from the column with the longest flute which will be the last move of a tier path.

To illustrate: Begin by choosing a row from the cascade. Represent this as *path a* in Figure: 13.

Note the location of a row (or path, relative to the position within the flutes it traverses). Proceed to the next path, namely *path b* in Figure 13, which is phase shift congruent to *path a*; Note that the same path positions, relative to flutes are being traversed, for all the other labelled paths. That is *path a* through *path d* are phase shift congruent for only three moves (columns  $2^n$  to  $2^{n+2}$  in Figure 13).

### 11.1.1 Why Uniform Range Spacing, Within a Tier?

Then Section 11.1 shows why uniform range spacing of  $2^k$  ( $k \in \mathbb{N}_1$ ) is used between paths in a tier. Where  $2^k$  is the length of longest flute in the cascade from which the tier is assembled.

Conversely, this ensures that each column-wise selected *offset*, in column-wise preceding, or subsequent, flutes has the same phase shift, and as a consequence the same offset. This is illustrated by comparing Figure 8b and Figure 7. Also note how this follows from: 41, 8.3 and including 8.1. Offsets in a path also determine the composition of a path's ones-phobic binary, refer to Section 9.8.

So this in turn makes the tier *columns* of the type *CS*, which are followed in later moves (in the ruler harmonic region) by  $rhs^n$  columns. (Refer to equation 41 which in turn is used by Remark 7.1).

**Lemma 11.1.** *Flute tessellation can extend indefinitely within for a seed range within  $\mathbb{N}_1$ . By extension, flute location congruent paths, for a fixed path length, occur an infinite number of times.*

*Proof.* Since, during tessellation, flutes from different columns align at regular intervals. That is, because two smaller flutes of the previous leftward column align with a flute twice as large from the adjacent column holding the next move. To find successive *location congruent paths* refer to the Section *Building a Tier* 11.1

Then the seed range distance, to the next location congruent path, is the length of the longest flute, containing the last path move. The reason: Let  $\Delta r_1$  be the seed range between the move, at top of the longest flute, and the last move of the first path.

Let  $\Delta r_2$  be the seed range between the move located top of the longest flute, and the last move of the second path.

Let  $\mathcal{F}$  be the length of the flute holding the last move.

Let the distance between paths be  $\mathcal{DP}$  then,

$$\mathcal{DP} = (\mathcal{F} - \Delta r_1) + \Delta r_2$$

Using flute location congruence, then:

$$\Delta r_1 = \Delta r_2$$

Then,

$$\begin{aligned}\mathcal{DP} &= (\mathcal{F} - \Delta r_1) + \Delta r_1 \\ &= \mathcal{F}\end{aligned}$$

That is, location congruent paths occur with a regular constant interval. This being the seed range of the longest flute in the tier. Refer to Figure 13

There is a bijection between a tier path and the longest flute that has only one tier path traversing it. The number of these flutes; being in length of a power of two, that fit into a cascade of length of a larger power of two; is finite. Therefore, the location congruent paths, *within a cascade*, are finite in number. Conversely, for an infinite seed range  $\mathbb{N}_1$ , there are an infinite number of location congruent paths, occurring at a regular (constant), interval.  $\square$

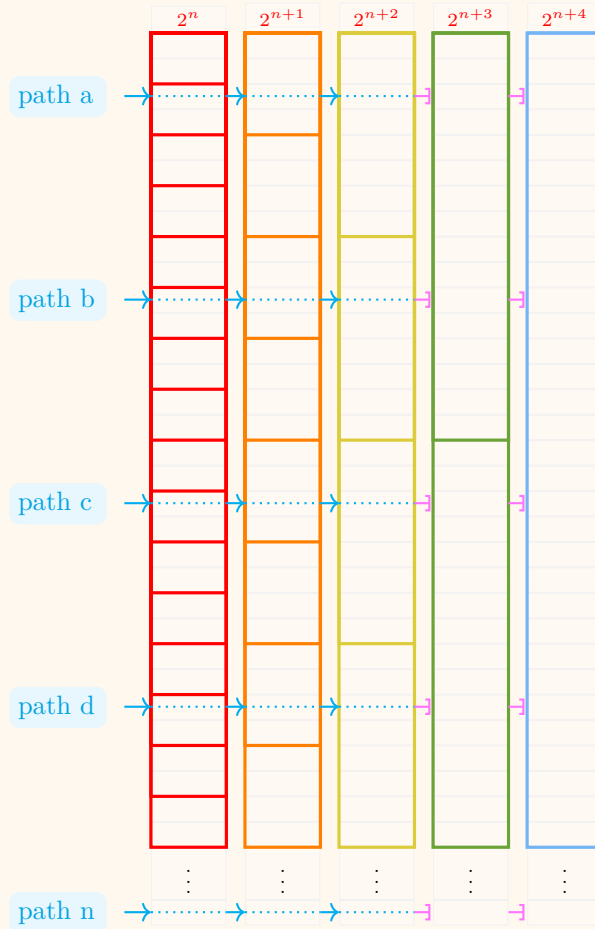
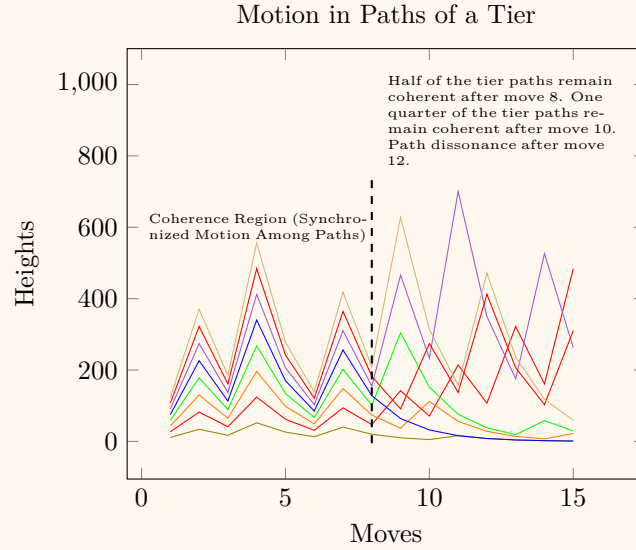


Figure 13: Paths Forming a Tier Within and Outside a Cascade  
 Flute tessellation is shown as successive non-overlapping rectangles.  
 Each of the tier paths traverses a flute at the same phase shift,  
 illustrating phase shift congruence.

Figure 14: Synchronization Transitioning to Dissonance



## 11.2 Properties of Paths in a Tier

Properties of phase shift congruent paths, continued: A graph is employed to better observe how path heights among paths in a tier behave. Refer to Figure 14. The actual data set of heights is shown in Table 5.

Although the data is discrete (not continuous), connecting lines were drawn between successive heights of the same path. This was done to emphasize cadence, (from move to move along paths). These line graphs undulate in cadence, within the coherence region. However, outside the coherence region, this synchronization is gradually lost. Here, lines do not move in the same general direction, some even intersect each other. Upon leaving the coherence region, it is also evident, that after each second move, half of the paths move out of cadence. After a certain number of moves, one of the paths, of the last remaining synchronous pair, moves out of cadence.

Even though the lines (of Figure 14), are an introduced artifact, they illustrate coherence (where lines are equidistant to other lines, at a given move), and dissonance (lines are not equidistant at a given move). (The hailstone analogy was discussed in: [Hay84] however, refer to this first [Del21].)

### 11.2.1 A Reason for Move Synchronization, in a Tier

1. Successive heights down a given column increase by a *power of six constant*.  
The reason: For a given move, the flute location congruent paths all have the

same offset (Section 8.1). This is illustrated in the figure of corresponding offsets, (in the coherent region of Figure 7). Here, in the coherent region, each column has the same offset.

2. Extend the tier paths rightward by one move. (called a *move extension*) Look at the squashed arrow heads in Figure 13. Note that there is more than one arrow pointing to locations within a particular flute. (Refer to: 13) Only one of these locations, per flute, can be used to form the next tier path. That is, these locations all need to have the same, *unique*, phase shift to satisfy phase shift congruency. This means only one path traversal per flute.

This is why, that for each *move extension*, the number of previously available synchronized paths drop by one half (as shown in Figure 13). Rightward successive flutes double in size. Only one relative location, within a flute, can be used for phase shift congruency. Since phase shift congruency is defined between successive downward (or upward), flutes. Thus, phase shift congruency ensures uniform range spacing, between path heights, downwards (or upwards), along a column. (From flute location congruency, recall 8.1.)

### 11.3 More Properties of Ruler Harmonic Sequences

**Lemma 11.2.** *You can not insert a second path into a single flute. This would make the tier path lengths greater than the columnar range between tier paths. In other words:  $m$  parity insertion (not appending), breaks a ruler harmonics sequence (of  $m$  parities). An insertion would be out of place (like a musical note, out of key, in a rising ostinato).*

*Proof.* In a *harmonic rule*, of a given order, there are rises and falls of peaks surrounded by sub peaks. Inserting a new height breaks this pattern. Regard the arbitrary insertion point. In Figure 15, no matter where inserted, the inserted  $m$  parity will always be located in between: an  $m$  parity minima and a non  $m$  parity minima. However, every non minimal  $m$  parity is bounded before and after by adjacent minima  $m$  parities. Also, no two minima  $m$  parities are adjacent to each other (by the definition of harmonic rule 4.2). Consequently, you can not insert an  $m$  parity into a ruler harmonic sequence. Moreover, this means that you can not have two flute location congruent paths in one flute. Since tiers have a ruler harmonic region and including a path would necessitate the insertion an  $m$  parity into a ruler harmonic sequence.  $\square$

**Lemma 11.3.** *Can a ruler harmonic series contain infinity as an element? No.*

*Proof.* In a finite range you have finite heights. For infinity to exist as an element, in a harmonic series, it would ultimately have to be part of an infinite range. Since the magnitude of a binary with an infinite number of trailing zeros is infinite. The sub-harmonics of  $n$  where  $n \in \mathbb{N}_1$ , are:

$$(n - 1), (n - 2), \dots, 2, 1.$$

Shown below are the sub harmonics of 4, namely: 3, 2, 1. Suppose the [standard sequence](#) has the property  $\infty \in S$ . Then  $S$  can be denoted as:

$$0, 1, 0, 2, 0, 1, 0, 3, 0, 1, 0, 2, 0, 1, 0, 4, 0, 1, 0, 2, 0, \dots, 0, \infty, 0, \dots, 0, (\infty - 1), 0, \dots, 0, (\infty - 2), 0, \dots$$

The sub-harmonics of  $\infty$  are:  $(\infty - 1)$ ,  $(\infty - 2)$ ,  $\dots$ ,  $(\infty - \infty)$ . The frequency of  $(\infty - 1)$  would be double that of  $\infty$ . The frequency of  $(\infty - 2)$  would be quadruple that of  $\infty$ , and so on. However, the values of  $\infty - n$  where  $n$  is a positive finite integer are:

$$\begin{aligned}(\infty - 1) &= \infty \\(\infty - 2) &= \infty \\&\vdots \\(\infty - \infty) &= 0\end{aligned}$$

Instead of being successively doubled, the successive sub-harmonic frequencies of  $\infty$  would become the same as  $\infty$ . Then the frequency distribution of the ruler harmonic series would be broken, by the definition of ruler harmonic series (Section 4.2 and Section 4.2.1. Therefore, a ruler harmonic sequence can not contain infinity as an element.  $\square$

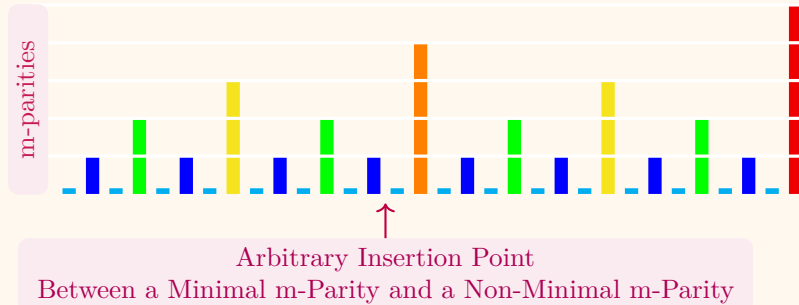
## 12 Regions: Coherent, Ruler Harmonic, and Dissonant

The paths in a tier can usually be subdivided into three regions: the coherent region (the region of synchronized motion among paths); the ruler harmonic region; the dissonant region. The regions can vary in their width (the number of columns they contain). (Each column delineates a particular move, at a particular distance from the seed value.) In tier paths, the coherent region is followed by the ruler harmonic region which in turn is followed by the dissonant region. Occasionally tier paths are missing the coherence region but there is always have a ruler harmonic region. In this case, an initial height with a small enough  $m$  parity (relative to the offset) would cancel sequence clamping (Section 7.4.1).

In the definitions that follow: The *first column* of a region is the column closest to the seed column. Where as, the *last column* of a region is the column farthest from the seed column.

**Definition 12.1** (coherence attenuation). The *coherence attenuation* is the first column of the ruler harmonic zone. Refer to Figure 6 and Figure 14.

Figure 15: A Single Insertion Breaks Ruler Harmonics



At the column of moves called the coherence attenuation the number of m parities that are the same drops to one half. The sequence in this column transitions from being clamped to being unclamped.

Here, at the coherence attenuation, the  $m$  parity minimum determines the width of ruler harmonic region. Why? The  $DV2$  operator removes a trailing zero, from the sequence held in the succeeding column of the next move. After a given number of moves the minimum mparity is reduced to zero. Then  $TP1$  operator is applied to obtain the next move. In the next move, by ruler harmonics, the number of minimal parities will be halved. The remaining,  $m > 0$ , m parities still have the  $DV2$  operator applied to them. (These operators are defined in Section 2.1 for equation 1. Consequently, the number of synchronized paths halves with each move.

Note: column-wise the offsets are the identical to each other, in the ruler harmonic region, and in the coherent region. The offsets are clamped in the coherent region and unclamped in the ruler harmonic region. (Refer to Section 7.4.1.)

As shown by 14 and 7. The coherence attenuation is defined to be the first column of the ruler harmonic region. This is where the number of synchronized paths drops by half. The halving of the number of paths that are synchronous continues, column by column, until the dissonant zone is reached.

**Definition 12.2** (Coherence Frontier). The *coherence frontier* is defined to be the last column of the ruler harmonic region. In Figure 6 it is the column that exhibits  $rhs^0$ . In Figure 7 it is the last column where all the offsets are the same (columnwise).

*Remark 12.1.* (A Coherence Attenuation is  $rhs^n$  for  $n \in \mathbb{N}_0$ ) A coherence attenuation column will either be  $rhs^0$ , or  $rhs^n$  which will ultimately be successively followed by a later  $rhs^0$  column. The coherence attenuation can be the same as the coherence frontier. Otherwise the tier column will begin with  $rhs^n$ ; because this column is built by successive additions of  $6^n$ . (This is from Section 8 in particular equation 41.) Eventually, after a finite number of moves the column becomes unclamped (should it be clamped to begin with). Then as  $rhs^n$ , upon repeated application of  $DV2$ , it's order is reduced to  $rhs^0$  (by Section 7.4.1).

Table 5: Tier Path Heights

Seed	Coherent Region							Dissonant (Non-Coherent) Region						
11	34	17	52	26	13	40	20	10	5	16	8	4	2	1
27	82	41	124	62	31	94	47	142	71	214	107	322	161	484
43	130	65	196	98	49	148	74	37	112	56	28	14	7	22
59	178	89	268	134	67	202	101	304	152	76	38	19	58	29
75	226	113	340	170	85	256	128	64	32	16	8	4	2	1
91	274	137	412	206	103	310	155	466	233	700	350	175	526	263
107	322	161	484	242	121	364	182	91	274	137	412	206	103	310
123	370	185	556	278	139	418	209	628	314	157	472	236	118	59

Table 6: Corresponding m-parities of Tier Paths

Seed	Coherent Region							Ruler Harmonics Region	Dissonant (Non-Coherent) Region						
11	1	0	2	1	0	3	2		1	0	4	3	2	1	0
27	1	0	2	1	0	1	0		1	0	1	0	1	0	2
43	1	0	2	1	0	2	1		0	4	3	2	1	0	1
59	1	0	2	1	0	1	0		4	3	2	1	0	1	0
75	1	0	2	1	0	8	7		6	5	4	3	2	1	0
91	1	0	2	1	0	1	0		1	0	2	1	0	1	0
107	1	0	2	1	0	2	1		0	1	0	2	1	0	1
123	1	0	2	1	0	1	0		2	1	0	3	2	1	0

coherence attenuation column      coherence frontier column

Table 7: Corresponding Cascade Offsets of Tier Paths

Seed	Coherent Region							Ruler Harmonics Region	Dissonant (Non-Coherent) Region						
11	6	6	36	36	36	216	216		216	216	1296	1296	1296	1296	1296
27	6	6	36	36	36	216	216		1296	1296	7776	7776	46656	46656	27993
43	6	6	36	36	36	216	216		216	1296	1296	1296	1296	1296	7776
59	6	6	36	36	36	216	216		1296	1296	1296	1296	1296	7776	7776
75	6	6	36	36	36	216	216		216	216	216	216	216	216	216
91	6	6	36	36	36	216	216		1296	1296	7776	7776	7776	46656	46656
107	6	6	36	36	36	216	216		216	1296	1296	7776	7776	7776	46656
123	6	6	36	36	36	216	216		1296	1296	1296	7776	7776	7776	7776

Seed	Coherent Region						Ruler Harmonic Region	Dissonant (Non-Coherent) Region							
33	●	○	●	○	○	○	●	○	○	○	●	○	○	○	○
27	●	○	●	○	○	○	●	○	●	○	●	○	○	○	●
43	●	○	●	○	○	○	●	○	○	○	○	○	○	○	●
59	●	○	●	○	○	○	●	○	○	○	○	○	○	○	○
75	●	○	●	○	○	○	●	○	○	○	○	○	○	○	○
93	●	○	●	○	○	○	●	○	○	○	○	○	○	○	○
327	●	○	●	○	○	○	●	○	○	○	○	○	○	○	○
323	●	○	●	○	○	○	●	○	○	○	○	○	○	○	○

An extension of Figure 8 is Figure 18; it uses more paths, and highlighted cells instead of o's and ●'s. The extension shows powers of two staircases bounding generations which make the quilt formation more recognizable.

Table 8: Ones-Phobic Encoding of Tier Paths

Seed	Coherent Region						Ruler Harmonics Region	Dissonant (Non-Coherent) Region							
11	34	17	52	26	13	40	20	10	5	16	8	4	2	1	
43	130	65	196	98	49	148	74	37	112	56	28	14	7	22	
75	226	113	340	170	85	256	128	64	32	16	8	4	2	1	
107	322	161	484	242	121	364	182	91	274	137	412	206	103	310	
27	82	41	124	62	31	94	47	142	71	214	107	322	161	484	
59	178	89	268	134	67	202	101	304	152	76	38	19	58	29	
91	274	137	412	206	103	310	155	466	233	700	350	175	526	263	
123	370	185	556	278	139	418	209	628	314	157	472	236	118	59	
16	48	24	72	36	18	54	27	27	<b>Tier offsets (not to be confused with cascade offsets)</b> The rows of the upper array (forming the <b>dissonance start</b> column, blue border), are the height differences between corresponding successive tier paths (magenta border). The results in the lower array, also with blue and magenta borders, are called tier <b>offsets</b> .						
16	48	24	72	36	18	54	27	27							
16	48	24	72	36	18	54	27	27							
16	48	24	72	36	18	54	27	162							
16	48	24	72	36	18	54	27	162							
16	48	24	72	36	18	54	27	162							

What is being shown: the point where one tier is split into two tiers; and how this is accomplished by grouping offsets together.

Continuing from the point where the **dissonance start** column is used as the sort key in a multi-stage sort. (The rows prior to the dissonance start have already been sorted. For comparison, the unsorted rows are shown in Table 5.) Before sorting the dissonance start column holds disparate tier offsets. After sorting, at the dissonance start column partitions into two sub-columns which each attain offset homogeneity. In the dissonance start column, the falling moves are outlined by a cyan border; the rising moves are outlined by a magenta border. Similarly, the tier offsets from falling moves are outlined by a cyan border; the tier offsets from rising moves are outlined by a magenta border. (Compare with cascade **offsets**, illustrated in Table 7.)

Table 9: Tier Path Heights Followed By Tier Offsets

## 13 The Tiara

### 13.1 Extra Structures Arising From (Formed by), Tiers

A tier can be extended to form a structure called a tiara. The tiara uses a building block called a bean. Beans are described in Appendix D more particularly in Section D.2.2. Multi-stage sorting the rows of a cascade forms a structure called a quilt (refer to Section 14). Moreover, sorting tier rows reveals a structure called a powers of two staircase (refer to Section 15.1).

**Definition 13.1** (tiara). Figure 16 is used to define a tiara.

### 13.2 Extending The Ruler Harmonic Sequence Definition to Include Tiaras

#### 13.2.1 Tiara Points of Equal Length Create a Tier

The tiara is a (column-wise or path-wise), extension of the ruler harmonic region of a tier. An example is illustrated in Figure 16. Each individual path, highlighted in plum is called a tiara point. Collectively the tiara points (including the points of zero length) are referred to as a tiara. The longest point is called a crown point. Beans, placed end to end, are the building blocks of a tiara point. A bean constitutes a single unit built from the ones-phobic ordered pair of  $\bullet \circ$ . (Beans are used in enumeration, refer to Appendix D.2.2). When building a tier extension, these properties are utilized:

- Tiara points are selected from regularly spaced seed values ( $\Delta r$  is a singular constant).
- As such, each tiara point will be of equal length (now measured in terms of the number of beans).
- Then the tiara points, having an equal number of beans, can be represented by an identical ones-phobic binary.
- The moves leading up to the tiara points are from a tier, which can also be represented by their own identical ones-phobic binary.

Consequently, the paths ending with tiara points of identical length can be collected together to form a [tier](#). Since for all selected paths  $\Delta r$  is a singular constant, and every path is represented by the same ones-phobic binary. (Defining properties are given in Section 11.2.) Then, the column with the following move after the tiara points is a coherence attenuation column (or a coherence frontier column). Then the  $m$  parities of this column follow a ruler harmonic sequence. (This explains the ruler harmonic distribution regarding the number of 0's following a tiara point, outlined by borders in Figure 16.)

### 13.2.2 Illustrating What Follows Tiara Points

A coherence attenuation column can also explain tiara point lengths. (Here, lengths are expressed in terms of the number of subsequent beans). The ones-phobic binary manifestation this column will have rows of ●'s alternating with ○'s (refer to the binary odometer of Section 4.2.1 based on the defining Section 4.2). The rows beginning with a ○ in the coherence attenuation column have a tiara point length of zero (in terms of bean length). The rows with ●'s in the coherence attenuation column have at least one zero following (by the property of ones-phobic binaries). Then every subsequent second, of these rows, has a tiara point bean length of at least one.

### 13.2.3 Extending a Tier by One Bean

The focus is now restricted to the rows that have one or more beans (shown highlighted in plum, in Figure 16). Since this is every second path, the seed range ( $\Delta r$ ), increases by two, (but stays constant). The moves leading up to the first bean are from a coherent region. Extending the tier to the first bean, will still maintain the coherence property. (The ones-phobic binary manifestation of these path remains identical, refer to Section 13.2.1) Hence the path extensions remain as a tier.

### 13.2.4 Building a Ruler Harmonic Sequence From Beans

Now the focus is on the first column following the first bean (of the tiara extended by one bean). It is a coherence attenuation column (because it follows a coherence region). As such, the  $m$  parties alternate, more specifically the ones-phobic binaries alternate, along the column between ● and ○ (as described by Section 13.2.3). In this column, every move that is a ● has a following ○ (in the next path move). Then (every fourth), path of the tiara extends by one bean.

The above argument, of this section, can be applied recursively, where every  $2^n$  th path of the tiara extends by one bean. However, this process describes the formation of a ruler harmonic sequence (refer to: the binary odometer of Section 4.2.1 and to the fundamental definition of Section 4.2). Consequently, the definition of a ruler harmonics sequence, of order zero, can be extended to apply to the distribution of tiara point lengths.

**Corollary 13.1.** *A tiara's point (exclusively consisting of beans, as a contiguous sequence), can only be finite in size.*

*Proof.* The point lengths of a tiara follow a ruler harmonic sequence ( $rhs^0$ ), refer to Section 13.2.4. A tier, and it's subsequent tiara, can be extended by extending the seed range. Although a crown point can be arbitrarily large, for a finite seed range. (This maximum for a crown point can get larger for a larger seed range. However, the seed range can only increase indefinitely by extending to higher, but not lower, integers.) Note the tiara lengths are a section or segment of ruler harmonics for integers (compare Figure 7 with Figure 16). This means the length for a

crown point(s) length(s) depend on the starting point and end point of the segment chosen for  $\triangle r$ .

Suppose the crown point is infinitely long and the seed range, containing it, is infinite. The unique values of the bean lengths are listed in decreasing order and evaluated.

frequency	1	2	3	4	5	...	$\infty$	$\infty$
bean length	$\infty$	$(\infty - 1)$	$(\infty - 2)$	$(\infty - 3)$	$(\infty - 4)$	...	$(\infty - (\infty - 1))$	$(\infty - \infty)$
evaluation	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	...	1	0

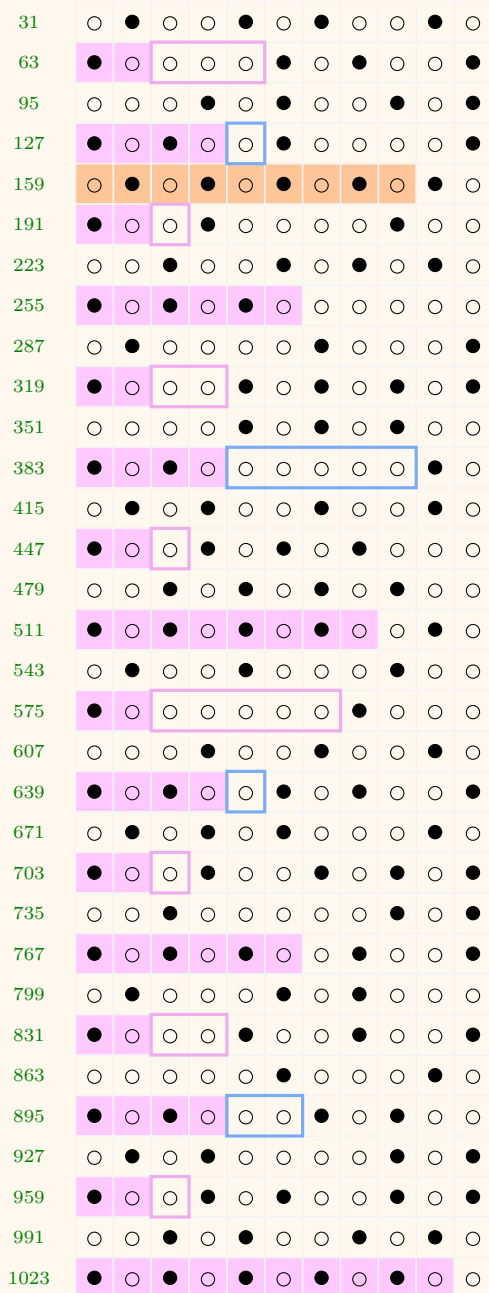
The finite bean lengths have infinite frequencies. This is allowable for an infinitely long sequence.

However, all the bean lengths occurring with a finite frequency have a length of infinity. That is, all of these bean lengths evaluate to the same length. Then by Section 5 the sequence of bean lengths, of successive infinities, can be classified as *CS*, of a finite length.

Lemma 11.3, shows that a ruler harmonic sequence cannot contain  $\infty$  as an element. Moreover, a *CS* sequence can not be matched anywhere into  $rhs^n$ , in whole or in part. Summarized by equation 4.

As such, a tiara point of infinite length can not be part of a ruler harmonic sequence; however, a tiara consists of tiara points whose lengths constitute a ruler harmonic sequence. A contradiction. Therefore it is impossible to have tiara points of infinite length, for a tiara with a finite seed range.

□



### Key to Tiara:

Seed values are in green. A tiara of beans (a  $\bullet \circ$  ones-phobic sequence regarded as a single unit), have their cells highlighted in plum. An individual column of a tiara is called a tiara point. Extra ones-phobic zeros following a tiara point are enclosed with a thicker border. If the tiara point is one bean in length the  $\circ$ 's are bordered in plum. If the tiara point is two beans in length the  $\circ$ 's are bordered in sky blue.

There is a coherence region of nine moves preceding the tiara (not shown). The coherence region consists of identical ones-phobic binaries. Incidentally, at the seed value of 159, the ones-phobic binary, of the coherence region, appears (highlighted in apricot). (Note. The  $\circ \bullet$  ordering is the reverse ordering of a bean.)

Similarly, the lengths of the tiara points follow a ruler harmonic distribution (shown in Section 13.2). Note that, a tiara point length of one bean occurs twice as often as a tiara point length of two beans. In general, a tiara point length of  $n$  occurs twice as often as a tiara point length of  $(n+1)$ .

Figure 16: A Tiara (Highlighted Tier After Sequences)

## 14 The Quilt

A quilt is described by stating its constituents. At the quilt's fundamental level are parity slides. These are the building blocks of a quilt rectangle. Quilt rectangles in turn are the building blocks of stacks. A generation consists of one or more stacks. A cascade is delineated by generations. Next, parity slides are used to reveal that quilt rectangles are actually tiers, which shows how a recursively smaller stack can be generated behind a quilt rectangle. Finally, this recursive property is used to reveal that all cascades contain all possible ones-phobic paths (for a given path length of  $n$  and cascade seed range of  $2^n$ ).

The reason for the name *quilt* is best illustrated by in the highlighted rectangles in Figure 12 and Figure 9.

### 14.1 Components of the Quiltwork Structure

**Definition 14.1** (*parity slide*). This refers to a string (sequence), of consecutive zeroes that can constitute a *parity slide*. A single trailing 1 (analogous to the least significant bit in a binary), if it exists, is considered part of the parity slide. A leading 1 that may bound the parity slide is not considered part of the parity slide. Without a 1, the trailing end of the main sequence can bound the parity slide. Either or both ends of the main sequence can bound the parity slide. Parity slides are defined for ones-phobic binaries of a fixed length  $n$ . Parity slides are numbered, as in the  $n^{th}$  generation parity slide, with respect its position among other sup-parities. A parity slide that is the entire ones-phobic binary is a 1st generation parity slide.

**Definition 14.2** (*generation*). The first *parity slide* is obtained by starting from the first path move (after the seed value). The second *parity slide* (if it exists), follows the first *parity slide*. Similarly, the third parity slide follows the second, etc. Collectively, all of the *first* parity slides, in a cascade, are referred to as the first *generation*. All the second parity slides are referred to as the second *generation*. Similarly, the third parity slides form the third *generation*, etc.

Generations become more obvious when grouped together after a multistage sort. Figure 9 is used to illustrate generations. Each generation is delineated by a thicker coloured line: blue between the first and second generations; red between the second and third generations; and green following the third generation. In the left diagram of Figure 9, the sorted parity slides get longer by one move, while proceeding down the cascade.

Observe individually each of the thicker lines that delineate the parity slide borders of the sub-figures in Figure 9. They are like steps on a staircase whose depths shorten by a half (as they proceed down the cascade). In a *cascade*, this is called a *powers of two staircase*. The staircase lines demarcate parity slides into a stack of quilt rectangles. The quilt rectangles are colour coded by the level of  $m$  parity of the initial move. Blue for 1 parity, green for 2 parity, yellow for 3 parity,

orange for 4 parity, violet for 5 parity, and white for zeroes not right bounded by a 1 before the right end of the cascade is reached.

## 14.2 Quilt Formation (By Sorting an $m$ Parity Cascade)

A quilt is the result of a multistage sort on an  $m$  parity cascade, or on an offset cascade, or on a ones-phobic cascade (illustrated in Figure 12 and Figure 9). Each column is indexed as a sort key, for a given stage in a multistage sort. After each sort stage, the cascade rows are rearranged according to the  $m$  parities of the column being sorted, or the offset values, or the ones-phobic values.

In a sorted cascade, the sub-column of identical moves ( $m$  parities, offsets, or bits), does not effectively change after a later sort stage. Why? Pertaining to a sub-column of identical moves, rearranging elements within that sub-column does not alter its structure, as part of a [quilt rectangle](#) (described in Section 14.1). (Remember each sort stage re-arranges entire rows in the cascade.) This is evident by comparing the seed value columns between the sorted and unsorted cascade in Figure 12 and Figure 8.

**Definition 14.3** ([Quilt rectangles](#)). Quilt rectangles are the result of a multistage sort on a cascade. Quilt rectangles are distinguished by color coding as in Figure 9.

The interest in sorting is to group parity slides, identical in length, according to their generation. Moreover, rows of identical parity slides need to be grouped contiguously to be considered a [quilt rectangle](#). A sub-sequence of descending  $m$  parities to a 0 parity, constitutes a parity slide.<sup>5</sup> A sub-sequence of identical offsets appended with an offset of higher value, constitutes a parity slide (shown in Section 9.9 and Section 9.10). Parity slides, by definition 14.1 are sub-divisions of a ones-phobic binary. (How it is encoded, is shown in Section 9.8).

### 14.2.1 Stacks (of Quilt Rectangles)

**Definition 14.4** ([stack](#)). Notice how the *quilt rectangles* are stacked on top of each other. The next quilt rectangle underneath is half the height of the one above it and is one cell wider. Here a *stack* refers to a collection of quilt rectangles stacked on top of each other.

In a [stack](#): Frequency of occurrence determines the depth dimension of the quilt rectangles. (More rows having the same *parity slide* at the same generation, increases the depth of the rectangle.

The  $m$  parity of the leading column of a quilt rectangle determines the width of the quilt rectangle; while the ruler harmonic frequency distribution (of Section 4.2), of that  $m$  parity determines the depth of the quilt rectangle.

<sup>5</sup>Since,  $m$  parities where  $m > 0$  are falling moves, are signified by a ones-phobic binary zeroes, and  $m = 0$ , being a rising move, is signified by a ones-phobic binary one.

While the width is determined by the  $m$  parity, of the leading column of a quilt rectangle, which decreases by 1 from  $m$  to 0 parity. (The leading column of the quilt rectangle consists of  $m$  parity values while the trailing column consists of 0 parity values;  $m > 0$  applies the *DIV2* condition of equation 1.)

The first generation contains only one stack. Its quilt rectangles are the largest. Smaller stacks are spawned from a single quilt rectangle (from the first generation). These are second generation stacks. Recursively increasingly smaller stacks are spawned from previous generation rectangles. This is observed in Figure 9 and Figure 12.

### 14.3 The Other Offset: Tier Offsets

There are two types of *offsets*. To distinguish between them, the first will now be called a *cascade offset*. This offset is generated from phase congruent locations within a cascade flute column. The second type of offset, called a *tier offset* is generated from successive height differences at the same move in an array of tier paths.

Compare the first type of offset (from Table 7), with the second type (from Table 9).

Recall (from Definition 11.2 that the tier height differences between successive paths, at a given move, are identical (in the coherence region).

The focus here continues after a multistage sort on the  $m$  parity manifestation of a cascade. The resulting multistage sort forms quilt rectangles consisting of identical sub-rows of descending  $m$  parities. That is, a *parity slide*. Their  $m$  parities successively descend, by 1 with each move, until a 0 parity is reached. The 0 parity column is the last, if not sometimes the only, column in the quilt rectangle. Consequently (by Section 3), the rectangle consists of rows of identical parity slides (from ones-phobic paths). The rises and falls of each path will be in sync. (Descending  $m$  parities signify a falling move(s). Since,  $m$  parities where  $m > 0$  are associated with an even height, implying *DV2*. A the terminating 0 parity signifies a rising move, since its associated height is odd, which implies *TP1*.)

The sub-row of zero's are falling moves. Then successively, these offsets are divided by two. Do not confuse this tier offset with a cascade offset. The terminating 1 will have an increase in its offset by a multiple of three. Offsets are rates of change (The 1 in  $3x + 1$ , is constant so  $\Delta 1 = 0$ . This was first developed in Section 8 in particular the right hand side of equations 40 and 39.) That is:

$$3^a \Delta^a r \quad \text{constant range, rising move}$$

$$\frac{1}{2^b} \Delta^b r \quad \text{constant range, falling move}$$

Now to focus on the first column of the quilt rectangle. Based on the  $m$  parity sorting selects every  $n$ th path from the unsorted cascade. That is, every  $n$  parity has a range  $\Delta r = 2^{n+1}$  towards the next seed of the same  $n$  parity, (from Definition 4.1, illustrated in Figure 7 and alternatively in Figure 28). The reason this happens is that the seed column is  $rhs^0$ . The selected paths are still equidistant but farther apart. The instances of the skipped offsets have to be added in, to adjust the tier offset value from the cascade offset value. That is:

$$\mathcal{O}_{tier} = \frac{\Delta r_{tier}}{\Delta r_{cascade}} \cdot \mathcal{O}_{cascade} \quad (54)$$

**Lemma 14.1.** *All cascade offsets, and tier offsets, even in the dissonant region, contain a factor of  $3^a$  in their offsets.*

*Proof.* To begin refer to Section 8.1.6, (the left hand side of equation 32). That is:

$$\frac{3^a}{2^b} (\Delta r^{a+b}) \quad (55)$$

The  $3^a$  numerator ensures that a tier offset will always have a factor of three. The  $2^b$  denominator cancels out since the  $\Delta r$  term has a factor of 2. Why? The first flute has a length of 2. It's offset elements are the tuple containing 1 and 6. The first stage in a multistage sort selects exclusively all paths beginning with an offset of 1, then exclusively all paths beginning with an offset of 6. This doubled the distance between sorted paths (within paths beginning with an offset of 1, and among paths beginning with an offset of 6). Consequently, the denominator in the term 55 cancels out, leaving an integer with a factor of  $3^n$ , for the tier offset (illustrated in Table 7). All odd factors do not change the order of any ruler harmonic series (by Lemma 7.2). The offset being a common difference in an arithmetic progression of heights from a quilt rectangle, results in a ruler harmonic series (by Remark 7.1).  $\square$

## 14.4 How a Quilt Rectangle Leads To a Next Generation Stack

To begin with, the first generation stack is obtained from the move **0** column, whose indices are  $\mathbb{N}_j$ , ( $j > 0$ ); the range is  $2^k$ . Then the  $m$  parity type is  $rhs^0$  (refer to Section 4.2). Within a cascade, manifested as  $m$  parities, a column-wise multi-stage sort, using  $m$  parities as the sort key, causes the formation of a first generation stack. In later stages of the multi-stage sort, each of the quilt rectangles can recursively spawn a stack that shadows it. How?

A quilt rectangle (by Definition 14.3) consists of a stack of column-wise identical parity slides (Definition 14.1).

A quilt rectangle (being a stack of [parities slides](#)) is a special instance of a tier. The moves are not only column-wise synchronized in the coherent region.

They are also column-wise synchronised in a way that all moves are falling, except for the moves of the last column that all rise. A quilt rectangle is the coherence region of a tier (Section 11.2).

Quilt rectangles have tier offsets (refer to Section 14.3). These offsets form column-wise arithmetic progressions. However, each has its own common difference. These arithmetic progressions are the indices of  $cs^k$ , if these columns exist, or after several moves  $rhs^k$  and ultimately after more moves  $rhs^0$  (Why? Remark 9.2, Remark 7.2 and Remark 7.1).

As shown by equation 41, cascade offsets are all  $6^n$ ,  $n \in \mathbb{N}_0$ . While tier offsets are arithmetic progressions, that are different in that their common differences can include a factor of a multiple of three. The common difference is reduced to a multiple of three by successive operations of  $DV2$  after successive fall moves from a quilt rectangle's [parity slide](#) (an example is shown in Table 9).

Then each quilt rectangle has a coherence frontier (a  $rhs^0$  column) by Remark 12.1). Another approach of reasoning is developed by using placements and landings in Section 9.

Remark 9.1. The coherence frontier (of sequence type  $rhs^0$  and a columnar range of  $2^k$  [a quilt rectangle's range] ) will exist due to the following constructs:

- i. Remark [Generating, Constant or Ruler Harmonic, Sequences From Powers of Six] 7.1
- ii. Remark [Division of Ruler Harmonic Sequences by  $2^n$ ] 7.2 similar to Remark 9.2
- iii. Remark [ $m$  Parity Enumeration With Regards to Sub-Sequence Span and Translation] 9.1 Of particular relevance is Theorem 9.3.
- iv. Remark [ $m$  Parity Properties Involved When Changing The Common Difference of an Arithmetic Progression] 9.2

Summarizing with a key point: The coherence frontier column of a quilt rectangle can be manifested as an  $m$  parity subsequence. Let it be called  $W$ . Why  $W \lesssim S$  is developed in Section 7.

## 14.5 A Quilt Margin and Fill Sufficiency

We need to show that any stack's quilt rectangles successively increase from a width of one move to a width that extends to, or exceeds, the trailing boundary of its cascade. To accomplish this, the concept of a stack enclosing rectangle (a quilt margin) is introduced. The foundational rationale used by this section is in Section 14.4

**Definition 14.5** (*quilt margin and margin's compliment*). The dimensions of a [cascade](#) are given by the row range of  $2^n$  and the column range of  $n$ . A *quilt*

*margin* is contained within a cascade. A *quilt margin* is a rectangular enclosure with dimensions  $2^k$  (rows) by  $k + 1$  (columns) where  $k \leq n$  with  $n \in \mathbb{N}_2$ , and  $k \in \mathbb{N}_0$ . (Note also, Section 5.3 with Remark 9.1). The *quilt margin*, shadows a quilt rectangle.

That is, a *quilt margin* is allocated in the region that uses the same paths as the quilt rectangle. However, the *quilt margin* consists of all moves, and only those moves that, follow its quilt rectangle (within the confines of the cascade). The *quilt margin* will exist provided that the trailing border of the quilt rectangle being shadowed does not reach the trailing border of the cascade. The trailing The row-wise dimension (length) of the *quilt margin* is the same as the longest  $m$  parity slide following the quilt rectangle being shadowed.

Note the distance between the leading column of the first generation and the and the trailing column of the quilt in the generation being shadowed. This distance (called the *margin's compliment*). Quilt margins are illustrated in Figure 17.


**Definition 14.6** (*fill sufficient*). A quilt margin with boundaries that encompass a shadowing stack that has its trailing boundary (of the last move) reach, or surpass, the trailing boundary of the cascade is *fill sufficient*. In other words, any stack in a fill sufficient quilt margin will contain all parity slides ranging, by succession, from 1 to a length  $k + 1$ , where the width (row range is  $2^k$ .

The longest parity slides are included even if they are truncated by the trailing boundary of the cascade. (Because we are only concerned with the parts of paths forming a ones-phobic binary *within* a cascade.)

**Corollary 14.2.** *Every quilt margin's trailing boundary extends, at least, to the trailing boundary of its cascade. (Here, this is referred to as: trailing boundary attained).*

*Proof.*

By definition, the dimensions of a *cascade* are:  $2^n$  by  $n$ . Which is the row range by column range (or seed value range by path length). The span of a coherence frontier column, of a quilt rectangle, is  $2^k$ ,  $k, n \in \mathbb{N}_1$ ,  $k \leq n$ . In a cascade the change in dimensions among quilt margins can be can observed in two ways, vertically and horizontally.

To illustrate (Figure 17), the successive top quilt margins are bounded by these rectangles: .

### The vertical progression of quilt margins along a cascade

We now focus on the stack of the first generation. The quilt rectangle the first generation stack follows is the seed column. The quilt margin of the seed column is the entire cascade. The top quilt rectangle of this stack has the narrowest and longest dimensions ( $2^{n-1}$  by 1). Its margin compliment dimensions are  $2^{n-1}$  by  $n$

(Section 14.4). The columnar dimension of the margin compliment is the same as columnar dimension of the cascade. As a consequence, the quilt margin is fill sufficient.

Focusing now on the next lower quilt rectangle of the first generation stack, the quilt rectangle's height reduces by half, and the width increases by 1. Likewise, the shadowing quilt margin row dimension reduces by half; the column dimension decreases by 1. (Section 14.4). The decrease in the quilt margin's row dimension is compensated for by a similar increase in the the width of the quilt rectangle. The sum of margin's compliment and the quilt margin's width ensures fill sufficiency.

Similarly, the widths of succeeding first generation quilt rectangles increase by 1, and the corresponding lengths decrease by half. The sums of their corresponding margin compliment with the quilt margin width ensures fill sufficiency.

Consequently, all of the quilt margins of the first generation are fill sufficient.

### **The dimensional change in quilt margins during their horizontal progression along a cascade**

Refer to Figure 17. We now focus on the top quilt rectangle, of each generation, of a cascade. The lengths of a quilt rectangle, of each successive generation, reduces by one half. (by Section 5.3, Section 14.2.1), also Section 11.2).

Consequently by definition, the corresponding quilt margins lengths (of these quilt rectangles), each successively reduces by half. In the first generation the smallest width of a quilt rectangle is 1 (the margin compliment), its length would be half of the cascade length  $\frac{2^n}{2}$ . The quilt dimension would be  $2^{n-1}$  by  $n$  (making it *fill sufficient*). In the second generation the smallest width of the quilt rectangle would be 2 (ones-phobic property), its length would be one quarter of the cascade length  $\frac{2^n}{4}$ . The quilt dimension would be  $2^{n-2}$  by  $n - 1$ . The margin compliment would be  $1 + 2$ . Adding the margin compliment to the margin width yields  $(1 + 2) + (n - 1)$  (making it *fill sufficient*). Similarly for the third generation, adding the margin compliment to the margin width yields  $(1 + 2 + 2) + (n - 2)$  (making it *fill sufficient*). This result would repeat for the  $a$ th generation,  $(1 + 2a) + (n - a)$  (making it *fill sufficient*).

To summarize: After establishing a quilt rectangle in the first generation is quilt sufficient, the top quilt rectangle in the next rectangle is shorter by one half, making its longest slide parity shorter by 1. To compensate the margin compliment is increased by 2, the sum of the parity slide length and the margin compliment reaches or exceeds the width of the cascade, making the next quilt rectangle *fill sufficient*.

This process repeats with the top rectangle of the following generation. Again, the current quilt rectangle length is halved making the longest parity slide reduce by 1. Again, the margin compliment increases by 2. This causes the sum of the margin compliment and the longest parity slide length reach or exceed the cascade width, ensuring fill sufficiency. This process repeats for succeeding quilt rectangles.

## Horizontal and Vertical Progression

We have established fill sufficiency for the first generation stack and the top row of quilt rectangles in a cascade.

Now we begin with the second (from the top) quilt rectangle of the first generation. This quilt rectangle is fill sufficient. The top rectangle in the second generation stack shadowing is then fill sufficient by the same reasons given in Section 14.5. The remaining quilt rectangles of this stack are also fill sufficient by the same reasons given in Section 14.5.

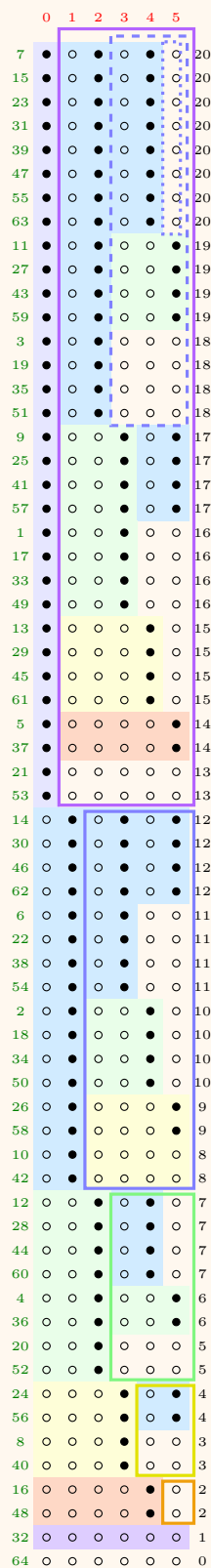
To summarize: Going either horizontally or vertically: The margin compliment increases by the width of the introduced quilt rectangle which is more than the reduction in width of the new quilt margin. Since the quilt margin's width is determined by its height, refer to Remark 9.1; and since the corresponding height of the given quilt margin is tied to its quilt rectangle, which reduces by no more than half, refer to Section 14.4.)

This summary, when applied to any other quilt rectangle of the cascade, demonstrates fill sufficiency, by reiterating the combined arguments of Section 14.5 and Section 14.5.  $\square$

**Corollary 14.3.** *Given that all quilt rectangles are fill sufficient. This establishes that a cascade contains all possible ones-phobic binaries that fit into the width of that cascade.*

*Proof.* Every quilt rectangle in the first generation is fill sufficient. That is, the parity slides of each of those quilt rectangles will have all possible parity slides in the second generation. Similarly, due to recursive reiteration by applying fill sufficiency, all possible parity slides for the next generation are generated. This approach is repeated until the last generation, of the cascade, is reached.  $\square$

Figure 17: Quilt Margins



**KEY:** The seed value column is in green. The decimal equivalent of the ones-phobic binary is the right most column in black. (A numeral system using Fibonacci numbers as a basis is described in Appendix A.) Quilt margins are delineated by rectangles. Proceeding vertically, solid rectangles are used. Proceeding horizontally, dashed rectangles are used.

## 15 Enumeration of a Mosaic

**Definition 15.1** (mosaic). A mosaic contains only all unique parity slide permutations of the ones-phobic binaries of a cascade. The dimensions of a cascade are  $2^n$  by  $n$ ; the dimensions of a mosaic are  $F_{n+2}$  by  $n$  (where  $F_x$  is the  $x$ th Fibonacci number).

With the arguments developed in Section 14.4, we can now also demonstrate, by an illustrative aid, the sorted quilt structure of a cascade as a result of a process that generates all possible ones-phobic binaries, of a given length  $n$  (the width of the cascade). This is done by a bijective mapping of a grove of enumerative trees to a mosaic.

The enumeration of Appendix C also maps to the mosaic. Each node value maps to a parity slide length. Each branch from the root node to a leaf maps to a ones-phobic path. Consequently, the mosaic holds all possible unique ones-phobic binaries, which are also found in the cascade, by a bijective mapping.

Compare the powers of two sequence (OEIS: A000079) to the Fibonacci sequence (OEIS: A000045)<sup>6</sup>. We note that powers of two sequence is component-wise dominant to The Fibonacci sequence.

Compare Figure 9 with Figure 10. More specifically, compare the step depth of corresponding staircases (within the corresponding stacks within corresponding generations) each figure. Here is where component-wise dominance manifests itself. Visually notice the bijective mapping of staircases and their steps between the corresponding Figure 9 and Figure 10.

Also note:

A parity slide is color coded by its bit length (the  $m$  parity of the move represented by the leading bit in the parity slide). A last parity slide that overflows is not color coded since its bit length is not specified. The figures only show the truncated bit length to the trailing border. (Using only ones-phobic binaries of a given size  $n$  (the width dimension of the cascade and its mosaic).

---

<sup>6</sup>The On-Line Encyclopedia of Integer Sequences, 1964 N.J.A. Sloane

## 15.1 Sorting the Ruler Harmonic Region to Form a Powers of Two Staircase

In Figure 9 and Figure 10, the trailing border of a stack is outlined by thicker line forming a powers of two staircase.

In Figure 12 and Figure 18, a powers of two staircase is demarcated by outlining a sequence of cells. These cells highlight a column wise, and a diagonally, *connected* staircase pattern that is continuous. Which are the trailing border cells included in a *stack*. These *staircase cells* form an intermediate region that falls in-between the coherent region (of alternating single column stripes), and the dissonant region (where there is no apparent cellular pattern for the given seed range).

This tier uses a greater seed range than the tier in Table 9. Seeds from a different tier are used. This tier was sorted to reveal *powers of two staircase patterns*. (Refer to Section 14 which shows quilt formation.) The blue bordered cell blocks recursively shrink by a half going rightwards and downwards. Ones-phobic encoding: ■ = a rise, ■ = a fall.

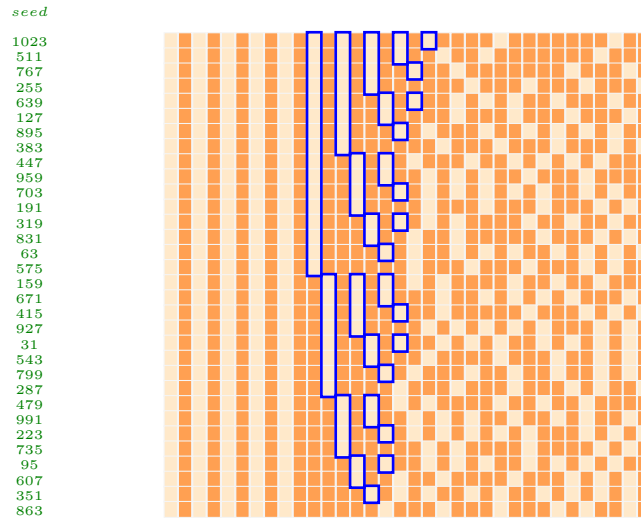


Figure 18: Region Transition: From Coherent to Quilt to Dissonant.

Remark 9.3 and For Remark 12.1 are conclusions, each acting as containers that spawn multiple references.

## 16 Bringing Together Remarks

Remark 9.3 concludes that all offsets of a flute tessellate column-wise with successive flutes. Phase shift congruent offsets selected from column-wise successive flutes from a cascade a column are used to form a tier column.

Remark 12.1 concludes that successive tier column elements are essentially arithmetic progressions. These may start as a  $cs^n$  which after a few moves genetates a  $rhs^n$  column. After a few more moves a  $rhs^0$  column is generated. (For some tiers the initial column is already  $rhs^0$ .) Therefore, within a tier, after a given number of moves, a coherence frontier is generated. In turn the coherence frontier generates a ruler harmonic region. Consequently, a ruler harmonic region will always exist.

### 16.1 Longer and Longer Paths

**Lemma 16.1.** *Suppose there exists an infinitely long path, without any move containing 1 as a height. (The seed range containing such a path, is  $\mathbb{N}_2$ .) This path could optionally, although not necessarily, contain a loop (other than the  $[4 \rightarrow 2 \rightarrow 1 \rightarrow \dots]$  loop). A loop unrolled contains successive subsequences of finite length that repeat indefinitely. Consequently, such a loop also would generate an infinitely long path.*

*Proof.* Suppose a tier containing infinitely long paths can be created. (All tier paths are of the same length; also, each tier path would have the same ones-phobic binary representation. Refer to the definition in Section 11.2.) Suppose the first tier path has a seed at some finite value. The seed range to (or length of the longest flute in the column of the last move), would be the seed range to the next path in the tier (based on Section 11.1). Then there are at least two flutes in the tier.

Aside: Recall that a tier can contain an infinite number of tier paths all equal in length (refer to 11.1). All of these paths will have an identical ones-phobic binary encoding, for tier paths in their coherence region. Other tier paths begin only with a ruler harmonic region. Within a given tier, begin with a finite length tier path and a finite  $\Delta r$  (the seed range between tier paths, which is also the length of the longest flute, residing in the column of the last move). Then  $\Delta r$  doubles for each added move that forms the selected paths for a tier (this is by design, illustrated in Section 11.1).

Imagine that the path length  $n$  of the tier gets longer such that  $n \rightarrow \infty$ . Then  $\Delta r \rightarrow \infty$ , (by Section 11.1).

(Imagine that each flute length, being the longest in it's path is analogous to a bus with an infinite number of passengers, and a tier is an infinite number of those buses arriving at the Hilbert hotel.)

Then, in effect, reverse a pairing function so that each of the flutes, with their elements, is concatenated. The Zermelo–Fraenkel set theory, with the axiom of countable choice, proves that the union of countably many countable sets is countable, by using a pairing function.<sup>7</sup> Then the cardinality of the tier seed range is  $\aleph_0$ .

Regard the tier with all paths being infinitely long. (The ones-phobic binary for each of these paths is identical.) At any move, there is no divergence among tier paths, since they are tier paths (by definition 11.2. For an infinitely long coherent region, the tier paths have no ruler harmonic region (with it's coherence periphery and coherence frontier). Then an  $rhs^0$  tier column does not exist. However, all tier paths have a ruler harmonic region. (Delve into 16 along with it's associated concluding remarks and the sections they refer to.)  $\Rightarrow\Leftarrow$

#### 16.1.1 Exception: Singular Paths

Suppose there exist infinitely long paths (with or without loops), that are singular (that is, not part of any tier).

Note that, in a cascade which can include all moves, each move is part of a column-wise arithmetic progression (being either  $cs^n$  or  $rhs^n$ ). However, arithmetic progressions are used to build a tier. Namely, from moves having identical phase shifts from successive flutes (arithmetic progressions), of a cascade column. Moreover, these paths have a seed range  $\Delta r \in \mathbb{N}1$ , even if the seed range is very long. Then, by the definition of a tier, the paths are part of a tier. (Refer to Section 11.2).  $\Rightarrow\Leftarrow$

#### 16.1.2 Conclusion

Using the terminology of the iterative form of 1, that is,  $hs_x$  is the path height at move  $x$ .

Then (excluding the  $[4 \rightarrow 2 \rightarrow 1 \dots]$  loop subsequence):

$$\forall hs_x \text{ such that } x \in \mathbb{N}_1, \nexists hs_\infty. \quad (56)$$

Also:

$$\forall hs_x \exists hs_x = 1 \text{ such that } x \in \mathbb{N}_1. \quad (57)$$

---

<sup>7</sup>[https://en.m.wikipedia.org/wiki/Zermelo–Fraenkel\\_set\\_theory](https://en.m.wikipedia.org/wiki/Zermelo–Fraenkel_set_theory)

To paraphrase: There does not exist an infinitely long hailstone path, or a hailstone path with a loop (other than the  $[4 \rightarrow 2 \rightarrow 1 \rightarrow \cdots]$  loop).

□

## A How and Why Fibonacci Numbers Are Used as a Basis in a Numeral System

Why?

By Definition 15.1 mosaics are subsets of cascades. A sorted mosaic is made up of all possible unique ones-phobic binaries of a given length (by Corollary 14.3). Consequently, these ones-phobic binaries range from **unsaturated** to fully **saturated**. We begin by assigning zero to the unsaturated ones-phobic binary and the value of the length of a mosaic, minus one, to the fully saturated ones-phobic binary. Refer to Figure 19. The **canonical bean form** occurs (at the rows with the associated decimal values of): **1, 2, 4, 7, 12, 20, 33**. When excluding the columns preceding these canonical bean form, their rows successively form the bottoms of larger mosaics. The rectangles (plumb coloured), in Figure 20, delineate nested mosaics of successively larger sizes. These mosaics list all possible ones-phobic binaries. The lengths of these mosaics are Fibonacci numbers, as demonstrated in Appendix C. This can be shown by manual enumeration of the first few smallest mosaics (starting from a mosaic length of 2, 3, 4,  $\dots$ ,  $n$ ). Fibonacci numbers can be used as a basis for a numeral system to enumerate a mosaic of any length  $n$ . How? To answer this, we need to implement succession.


## B Succession in One's Phobic Binaries

Succession proceeds from a fully **unsaturated** ones-phobic binary to fully **saturated** ones-phobic binary (described in the side note of Figure 19). Comparing succession in ones-phobic binaries with succession in ordinary binaries:

**Note 1:** Where **step 1** is analogous to binary addition by 1 to a 0 bit, the least significant bit position of the augend.

**Note 2:** Also **step 2** is analogous to binary addition by 1 to a 1 bit. When the augend of the current place value has a 1. Adding 1 to that place value will set it to 0 and a 1 will be carried to next significant place value. Moreover, if all remaining (less significant), place values are 1s, then these are all set to zero. For example, in binary:  $0111 + 0001 = 1000$ .

Taken to be axiomatic: Succession in  $\mathbb{B}_n$  (ordinary binaries of a fixed length  $n$ ), will ultimately obtain every possible binary digit arrangement for that length. This is conceptualized as the odometer effect, mentioned in Section 4.2.1.



By extension **Note 1** and **Note 2** the binary succession algorithm can be adapted to tethered ones-phobic binaries. Instead of succession by binary addition by 1, succession is performed on ones-phobic binaries by . The  $? \dots ?$  notation is explained in Section 3.1.

$$? \dots ? 0 + 1 = ? \dots ? 1 \quad \text{corresponds to} \quad ? \dots ? \bigcirc + ? \dots ? \text{blue} = ? \dots ? \text{blue} \quad (58)$$

$$? \dots ? 01 + 1 = ? \dots ? 10 \quad \text{corresponds to} \quad ? \dots ? \text{blue} + ? \dots ? \text{blue} = ? \dots ? \text{blue} \bigcirc \quad (59)$$

$$? \dots ? 10 + 1 = ? \dots ? 11 \quad \text{no correspondence} \quad ? \dots ? \text{blue} \bigcirc + ? \dots ? \text{blue} = ? \dots ? \text{blue} \bigcirc \bigcirc \quad (60)$$

$$? \dots ? 11 + 1 = ? \dots ? 100 \quad \text{no correspondence} \quad ? \dots ? \text{blue} \bigcirc \bigcirc + ? \dots ? \text{blue} = ? \dots ? \text{blue} \text{blue} \quad (61)$$

**Caution:** When regarding  $\mathbb{B}_n$  with its succession operator and  $\mathbb{F}_{n,2}$  with its succession operator as groups, there does not exist a complete isomorphism. This is shown as *no correspondence* in equations 60 and 61. However, the *clear and carry* operation (of **Note 2**) combined with the *increment to the least significant clear place values* operation (of **Note 1**) causes its own version of an *odometer effect* to be observed in tethered ones-phobic binaries (utilizing s and  $\bigcirc$ s in place of binary 1s and 0s). That is, successively adding a  using the appropriate operations of **step 1** or

**step 2** causes the resulting dot product of  $\mathbb{F}_{n,2} \cdot \mathbb{P}_{n-1}$  to increase by one. This is what implements an exhaustive approach that is complete (in the sense that all possible ones-phobic binaries are generated). Why?

Whenever the least two significant ones-phobic binary place values are both clear (set to zero) adding a ■ increases  $\mathbb{F}_{n,2} \cdot \mathbb{P}_{n-1}$  by 1. This being **step 1** is demonstrated by equation 58.

Whenever the leading half of ■ is advanced to the next available more significant place value while clearing the remaining ■ **saturation**, the  $\mathbb{F}_{n,2} \cdot \mathbb{P}_{n-1}$  increases by 1. Instances of this are shown in Figure 19 at the decimal values of **2, 3, 5, 8, 13, 21, 34**. **Saturation** behind the place value being carried into occurs at odd significant place value positions (originating from right to left) at decimal values **1, 4, 12, 33** and even place value positions at decimal values **2, 7, 20**. The generalization of this (**step 2**) that is, succession is implemented by adding in the next Fibonacci number at the next significant place value that is available and clearing to zero the **saturation** of less significant ■s. That is all less significant odd Fibonacci numbers, or otherwise all less significant even Fibonacci numbers. The derivation follows:

Note: Starting with a given Fibonacci number, we need to develop an expression that includes a generalization of every second Fibonacci number (all less than the given Fibonacci number summation) plus 1. By implementing an iterative proof, we begin with the definition of a Fibonacci number. The last term of any given line is expanded into the last two terms in the next line ( $F_n = F_{n-1} + F_{n-2}$ ). We continue with a finite number of expansions towards a minimal finite value (of  $F_3$  or  $F_2$ ). Note.<sup>8</sup>

The following identity validates the clear and carry succession procedure described in [Note 1](#) and [Note 2](#), for a given length of a ones-phobic binary.

$$\begin{aligned}
F_{n+1} &= F_n + F_{n-1} && \text{by definition of a Fibonacci number} \\
&= F_n + F_{n-2} + F_{n-3} \\
&= F_n + F_{n-2} + F_{n-4} + F_{n-5} \\
&= F_n + F_{n-2} + F_{n-4} + F_{n-6} + F_{n-7} \\
&\vdots \\
&= \begin{cases} F_n + F_{n-2} + F_{n-4} + F_{n-6} + \cdots + F_3 & \text{if } n \text{ is odd} \\ F_n + F_{n-2} + F_{n-4} + F_{n-6} + \cdots + F_2 & \text{if } n \text{ is even} \end{cases} \\
&= 1 + \sum_{\substack{i \geq 2 \\ i \equiv n \\ i \in \mathbb{F}_{n,2}}} F_i
\end{aligned}$$

This identity can be adapted to determine the length of a **mosaic** given its width  $n$ , when it is applied to a fully saturated ones-phobic binary.

$$\text{mosaic length} = 1 + \sum_{\substack{i \geq 2 \\ i \equiv n \\ i \in \mathbb{F}_{n,2}}} F_i = F_{n+1} \tag{62}$$

<sup>8</sup>The result of this derivation as an identity from equation 62 led me to speculate that the identity might be listed. A check, found similar identities in [Wel97] and [And71].

Figure 19: Succession in a Ones-Phobic Numeral System

**KEY:** In the top row  $F_n$ , designates the  $n^{th}$  Fibonacci number. The second row (from the top), shows the Fibonacci number's corresponding value. Succeeding rows are ones-phobic binaries. (The notation is described in Section D.2.2.) To the right of a ones-phobic binary is its tethered version. To the left (in green) is the associated decimal value of the ones-phobic binary. The decimal is obtained as follows: Let,  $\bullet = 1$  and  $\circ = 0$  (to disambiguate from the balance calculations of Section D.2.2). Next, assign

a basis of consecutive Fibonacci numbers:  $\mathbb{F}_{n,2} = F_n, \dots, F_4, F_3, F_2$ . The dot product ( $\mathbb{F}_{n,2} \cdot \mathbb{P}_{n-1}$ ) of any ones-phobic binary (from third row to bottom), with the basis of Fibonacci values, yields the associated decimal value. A ones-phobic cascade can be sorted using a multi-stage sort. Instead, the same can be accomplished using a single sort on the decimal binary associated with a ones-phobic binary. Here is an example calculation to obtain the decimal value of a ones-phobic binary:

$$\begin{aligned} \mathbb{P}_n \cdot \mathbb{F}_{n,2} &= [\circ, \circ, \bullet, \circ, \bullet, \circ, \circ, \bullet] \cdot [F_8, F_7, F_6, F_5, F_4, F_3, F_2] \\ &= [0(34) + 0(21) + 1(13) + 0(8) + 1(5) + 0(3) + 0(2) + 1(1)] \\ &= 19 \end{aligned}$$

	$F_9$	$F_8$	$F_7$	$F_6$	$F_5$	$F_4$	$F_3$	$F_2$
	34	21	13	8	5	3	2	1
0	○	○	○	○	○	○	○	○
1	○	○	○	○	○	○	○	●
2	○	○	○	○	○	○	○	○
3	○	○	○	○	○	●	○	○
4	○	○	○	○	○	○	○	○
5	○	○	○	○	○	○	○	○
6	○	○	○	○	○	○	○	○
7	○	○	○	○	○	○	○	○
8	○	○	○	○	○	○	○	○
9	○	○	○	○	○	○	○	○
10	○	○	○	○	○	○	○	○
11	○	○	○	○	○	○	○	○
12	○	○	○	○	○	○	○	○
13	○	○	○	○	○	○	○	○
14	○	○	○	○	○	○	○	○
15	○	○	○	○	○	○	○	○
16	○	○	○	○	○	○	○	○
17	○	○	○	○	○	○	○	○
18	○	○	○	○	○	○	○	○
19	○	○	○	○	○	○	○	○
20	○	○	○	○	○	○	○	○
21	○	○	○	○	○	○	○	○
22	○	○	○	○	○	○	○	○
23	○	○	○	○	○	○	○	○
24	○	○	○	○	○	○	○	○
25	○	○	○	○	○	○	○	○
26	○	○	○	○	○	○	○	○
27	○	○	○	○	○	○	○	○
28	○	○	○	○	○	○	○	○
29	○	○	○	○	○	○	○	○
30	○	○	○	○	○	○	○	○
31	○	○	○	○	○	○	○	○
32	○	○	○	○	○	○	○	○
33	○	○	○	○	○	○	○	○
34	○	○	○	○	○	○	○	○

○○○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

○○○○○○○

### A One's Phobic Binary Succession Algorithm

Succession can be more readily observed with tethered ones-phobic binaries. Exclusively using  $\bullet$ 's and  $\circ$ 's ensures that the ones-phobic property is not violated. No possible arrangement of  $\bullet$ 's and  $\circ$ 's will involve two  $\bullet$ 's next to each other. (Successive  $\bullet$ 's consist of  $\bullet\circ$  pairs so that no arrangement among them will allow successive  $\bullet$ 's to arise, and no  $\bullet$ 's outside of a  $\bullet\circ$  are utilised.)

In the tethered ones-phobic binary array, an extra column of zeros is appended to become the rightmost column. This column when not part of a  $\bullet\circ$  is distinguished by (red)  $\circ$ 's. This column is simply a place holder for the right half of a bean  $\bullet\circ$  (since a bean uses the space of two place holders, holding  $\bullet\circ$ ). As a consequence this newly appended place holder does not contribute, at any time, to the dot product sum. (As a matter of preferred consistency you could extend the basis by assigning  $F_0 = 0$  to this newly appended placeholder.) The steps in the succession algorithm are:

- step 0** Begin with an arbitrary sequence of bits, all set to 0.
- step 1** Find the first available pair of least significant zero bits. Set these zero bits to  $\bullet\circ$ . Availability means, a place value pair holding  $\circ\circ$ .
- step 2** Advance to the next significant bit with a new  $\bullet\circ$  and check for availability. If the new trailing half of  $\bullet\circ$  overlaps with the leading half of a previous  $\bullet\circ$  and if what follows this previous  $\bullet\circ$  is a contiguous sequence of  $\bullet\circ$ 's (having **saturation**), then set to zero all place values following the new bean.
- step 3** Repeat **step 1** and **step 2**, stop when a predetermined length of contiguous  $\bullet\circ$ 's is reached.

Figure 20: Succession of Ones-Phobic Binaries in Nested Mosaics

**KEY:** In the top row  $F_n$ , designates the  $n^{th}$  Fibonacci number. The second row (from the top), shows the Fibonacci number's corresponding value. Succeeding rows are ones-phobic binaries. (The notation is described in Section D.2.2.) To the right of a ones-phobic binary is its tethered version. To the left (in green) is the associated decimal value of the ones-phobic binary. The decimal is obtained as follows: Let,  $\bullet = 1$  and  $\circ = 0$  (to disambiguate from the balance calculations of Section D.2.2). Next, assign

a basis of consecutive Fibonacci numbers:  $\mathbb{F}_{n,2} = F_n, \dots, F_4, F_3, F_2$ . The dot product ( $\mathbb{F}_{n,2} \cdot \mathbb{P}_{n-1}$ ) of any ones-phobic binary (from third row to bottom), with the basis of Fibonacci values, yields the associated decimal value. A ones-phobic cascade can be sorted using a multi-stage sort. Instead, the same can be accomplished using a single sort on the decimal binary associated with its ones-phobic binary.

	$F_9$	$F_8$	$F_7$	$F_6$	$F_5$	$F_4$	$F_3$	$F_2$
	34	21	13	8	5	3	2	1
0	○	○	○	○	○	○	○	○
1	○	○	○	○	○	○	○	●
2	○	○	○	○	○	○	●	○
3	○	○	○	○	○	●	○	○
4	○	○	○	○	○	●	○	●
5	○	○	○	○	●	○	○	○
6	○	○	○	○	●	○	○	●
7	○	○	○	○	●	○	●	○
8	○	○	○	●	○	○	○	○
9	○	○	○	●	○	○	○	●
10	○	○	○	●	○	○	●	○
11	○	○	○	●	○	●	○	○
12	○	○	○	●	○	●	○	●
13	○	○	●	○	○	○	○	○
14	○	○	●	○	○	○	○	●
15	○	○	●	○	○	○	●	○
16	○	○	●	○	○	●	○	○
17	○	○	●	○	○	●	○	●
18	○	○	●	○	●	○	○	○
19	○	○	●	○	●	○	○	●
20	○	○	●	○	●	○	●	○
21	○	●	○	○	○	○	○	○
22	○	●	○	○	○	○	○	●
23	○	●	○	○	○	○	●	○
24	○	●	○	○	○	●	○	○
25	○	●	○	○	○	●	○	●
26	○	●	○	○	●	○	○	○
27	○	●	○	○	●	○	○	●
28	○	●	○	○	●	○	●	○
29	○	●	○	●	○	○	○	○
30	○	●	○	●	○	○	○	●
31	○	●	○	●	○	○	●	○
32	○	●	○	●	○	●	○	○
33	○	●	○	●	○	●	○	●
34	●	○	○	○	○	○	○	○

### Nested Mosaics

By definition a sorted mosaic is an array of unique ones-phobic binaries of all possible values ranging from completely unsaturated (all un-tethered zeros) to fully saturated (all beans in canonical form). In canonical form, a leading un-tethered zero is included for paths of an odd length. The ones-phobic binaries of the mosaic are sorted according to their decimal value which is converted from its ones-phobic binary by using the Fibonacci basis  $\mathbb{F}$  (refer to Figure 19). In the accompanying diagram, smaller mosaics are recursively nested inside the largest mosaic. (The mosaic's perimeters are delineated by plum rectangles.) Then, with the exception of the first two nested mosaics (whose lengths) are one, every last ones-phobic binary of each mosaic is fully saturated, or in canonical form, with respect to beans. The last ones-phobic binary in a sorted mosaic yields the largest decimal value. That plus 1 is the length of the mosaic. That is, equation 62 yields the length of the mosaic, when using the last path of the sorted mosaic. Why?

The number of beans of a fully saturated ones-phobic binary extends across the full width of the mosaic (when  $n$  is even). With the canonical form (when  $n$  is odd, an un-tethered zero leads the beans). Succession following the last ones-phobic binary, of a mosaic, clears all of the beans while carrying the leading part of a bean to the next significant place value for the next Fibonacci number used as a basis (refer to Note 2). By itself the fully saturated ones-phobic binary determines the length of a mosaic as:  $\mathbb{F}_{n+1}$  where  $n$  is the mosaic's path length.

Ones-phobic binaries can be directly sorted using a multi-stage sort. A base ten equivalent of a ones-phobic binary can be used as a sort key for a simpler sort. Assigning a Fibonacci based numeral system to ones-phobic binaries, of a given length, allows the associated base ten equivalent to be used as a sort key. As opposed to the base two numeral system, the Fibonacci numeral system leaves no gaps between successive ones-phobic binaries (of a cascade). A smaller input range allows for faster sorting.

## C Why there is a Fibonacci Staircase in A Mosaic

We demonstrate that the number of paths in successively longer mosaics correspond to successive Fibonacci numbers. This is achieved by using enumerating graphs.

**Definition C.1** (*enumerative grove*). The number of ones-phobic binaries from a [mosaic](#) can be enumerated by using a disconnected directed graph (of trees). Every vertex in this graph has a node associated with it. Each node contains one value that is the length of a parity slide. Let the root node of each tree represent the first (unique), parity slide of a ones-phobic binary. Then all ones-phobic binaries beginning with the same parity slide can be represented as [branches](#) on such a tree. (All other possible initial parity slides, of a given length from a ones-phobic binary, can be associated with its own tree whose root node value corresponds to that parity slide.) A [branch](#) whose associated node values (from the root vertex to a leaf vertex), correspond to the parity slides of a ones-phobic binary from its mosaic. This is what makes up a *enumerative grove*.

**Definition C.2** (*enumerative grove magnitude*). To list all possible ones-phobic binaries of a mosaic, the permutations of parity slides in each ones-phobic binary of a mosaic, an [enumerative grove](#) can be used. Let the length of these ones-phobic binaries of  $n$ , be the *enumerative grove magnitude*.

**Definition C.3** (*branch to root sum*). In an enumerative grove, all the vertices of a tree have an associated node. A node holds a value which is the length of a parity slide. A tree branch will reference the associated nodes of the vertices in a path proceeding from a end vertex to the root. The sum of the referenced nodes is called the branch to root sum.

**Definition C.4** (*enumeratively complete*). A tree of an enumeration grove is said to be *enumeratively complete* if the *branch to root sum*, for every end vertex node, sums up to either  $n$  or  $n - 1$ , for an enumerative grove of magnitude  $n$ . Note: the smallest node value that can be introduced is 2. Why? A branch to root sum of  $n - 2$  would invalidate the *enumeratively complete* property since a node value of 2 could still be added to form a sum of  $n$ . Moreover, a parity slide consisting of a single ones bit, can only appear at the beginning of a ones-phobic binary (By the property of a [ones phobic binary](#).) This is illustrated by the 1x1 bricks in the first column of a mosaic, refer to Figure 10.

### C.1 Building an Enumerative Grove

Once the root node is set, continue with the next level (node progeny or node generation), only if the ones-phobic binary length of  $n$  is greater than one. Let a cluster refer to only those vertices, of a given level, that share a common parent. To build a [cluster](#) under the root node, add nodes starting from 2 to  $n - 1$ . The value of successive nodes is incremented by one. Below each node of this cluster introduce its own child cluster. Continuing breadth first, nodes are added to the cluster. Then the next cluster at the same level is built, provided the cluster had a parent node to build upon. Clusters, like nodes, at a given level, are also built

breadth first. This process continues recursively when building the clusters to the next lower level. The decision of whether or not to add a node is determined by the branch path sum of the node values to the root node value. This sum includes the node value being introduced. This sum has to be equal to  $n$  (the magnitude of the enumerative grove), or  $n - 1$ . The process stops when no new nodes can be introduced.

This process is repeated for each successive root node, from 1 to  $n$ . The trees of resulting enumerative grove are referred to as being enumeratively complete

## C.2 Properties of an Enumerative Grove

### C.2.1 Enumerative Groves are self similar

The number of leaf nodes determine the width of the mosaic. (The length of a ones-phobic binary in the mosaic is  $n$ . This is the direct path from root to a leaf.) Also, let  $n$  be the magnitude of the enumerative grove. Each enumerative grove of magnitude  $n$ , where  $n > 1$ , contains a subgraph that is isomorphic to the enumerative grove of magnitude  $n - 1$ . Compare Figure 21 and Figure 22 with Figure 23.

That is, for the time being, ignore the numerical labels in the nodes of an enumerative grove. Note that when the first tree of Figure 23 is excluded the remaining trees are isomorphic to the tree group of Figure 22, which has a magnitude of 7. When the trees formed by root node 1 and 2 are excluded the remaining trees are isomorphic to Figure 21, which has a magnitude of 6. This process continues recursively, yielding successively smaller enumeration grove magnitudes, until the smallest enumerative grove of a single node is reached, which has a magnitude of 1.

Now to pay attention to the numerical labels of the nodes. The magnitude of the root node determines the size of the tree formed by it. The lowest root node value of 1 forms the largest tree in the enumerative tree group. As the value of the root node increases the corresponding tree gets smaller and smaller. At the root node of  $n$ , the tree consists only of the node itself.

In general, to obtain a smaller magnitude for an enumerative grove (for a magnitude of  $k$ ). Successively select the smallest  $k$  trees from the larger enumerative tree group. Subtract 1 from the value of each root node selected. This adjusts the branch to root sum to  $k$ . Since a root node value adjustment affects branch sums for the entire tree. Note, the selected trees are isomorphic to the enumerative grove of magnitude  $k$ . All that remains is to relabel the root nodes (by subtracting 1 from each selected root node value), the result will be the enumerative grove of magnitude  $k$ .

Another way to obtain an enumerative grove of the next smaller magnitude is to subtract the last (or largest), node value from each cluster at each level. These would be the red nodes in figures: 21, 22, and 23.

Removing the last node of a cluster, at a given level, sets the branch sum of the remaining nodes to  $n - 1$  or  $n - 2$ . Consequently, the enumerative grove magnitude is reduced by one.

Why? Since the value of these red nodes is only one more than the highest value of a cluster sibling, when there is more than one node in a cluster. This would reduce the branch to root sum by one. (Shown in the figures of: 21, 22, and 23). Should the red node be the only node in a cluster, its value would be 2. this would reduce the branch to root sum by two. The enumeration grove magnitude is determined by the higher of the two branch to root sums (by definition). This reduces the enumerative grove magnitude by one.

### C.2.2 Fibonacci Property Associated With An Enumerative Grove

Again for the time being, ignore the values of the nodes.

Refer to Figure 24. Here each tree (from an enumerative group), is partitioned into two sub-trees. The left sub-tree's root is attached as the first node of the first level. (This is always 2 in the labeled graphs, which was the root of the second next tree following the tree that formed from partitions.) The right sub-tree's root is attached as root in the tree formed from partitions. The right sub-tree makes up everything excluded by the first sub-tree. It is from the next tree following the tree being partitioned. This partitioning process repeats for each successively smaller tree in the enumeration grove, until the smallest tree is reached (which is a single node).

In Figure 24 the sub-trees are delineated by a border of a given colour. The corresponding colours are again delineating the next two trees having the next successively higher root values. (Root value succession is by 1.)

Within an enumerative grove, the partition of a tree shows that the sum of nodes of the two immediately smaller trees add up to the sum of the nodes of the current tree. Consequently, the number of nodes in each successive pair, of the next two smaller trees, of an enumerative grove, of a given magnitude, corresponds to the Fibonacci sequence (refer to Figure 24).

### C.3 Building the Next Larger Enumerative Grove

Begin with an enumerative group of magnitude  $n$ . Let each tree be partitioned as shown in Figure 24 (as described in section C.2.2).

This time, attention is paid to the values of the nodes.

Set aside copies of the first two trees with root values of one and two.

These are used as sub-tree partitions that form the next larger tree. The tree copy with a root value of one, has its root attached as the root of the new larger tree. For every node at the first level just below the root, the values are increased by one. This is to adjust each branch to root sum in the new tree, so that it corresponds to the magnitude of the new enumeration tree group, which is larger by one with respect to the old tree group.

The tree copy with a root value of two is attached as the smallest value node in the first level, which is always two. This tree copy is attached as is. Since, the root node value is one. This increases the branch to root sums by one. The number of nodes in both sub-trees add up to make the total number of nodes for the new tree being built. The sub-trees are based on enumeratively complete trees, of the previous enumerative grove of magnitude  $n$ . Then the sub-trees with their branch to root sums are now all increased by one, and are derived from enumeratively complete trees; now combine to form an enumeratively complete tree for an enumerative grove of the next larger magnitude.

Copy the former enumerative group. In the copy, increase all of the root node values by one and include them to the tree just formed by sub-trees. The result is an enumerative grove with a magnitude of  $n + 1$ .

*Remark C.1 (Results).* Therefore, in an enumerative grove the order of a tree is the sum of the two orders of last two smaller trees (of decreasing succession, starting from the third smallest tree). That is, the order of each successively larger tree (in an enumerative grove), is given by the Fibonacci sequence.

Refer to Figure 24, note that not just the order of a tree is given by the Fibonacci sequence, but also the number of branches in that tree. Since a tree is composed of the two sub-trees of the last two smaller trees (of decreasing succession, excluding the two smallest trees, of order one). The sum of the number of branches in each sub-tree forming the next larger tree, is also the number of branches in the next larger tree. This is due to the way the two subtrees are joined, (described in Section C.3). Therefore, the number of branches in each successively larger tree of an enumerative grove follows the Fibonacci sequence. The number of branches in a given tree represent the number of ones-phobic binaries in a mosaic. The root node value, of a given tree, is the first parity slide of a ones-phobic binary. The depth of a step in the first generation staircase is due to the number of branches (ones-phobic binaries with the same initial parity slide). Since the number of branches per tree follow the Fibonacci sequence, the depth of each step in the first generation staircase also follows the Fibonacci sequence. Thus, the name Fibonacci staircase. The argument just used can be applied recursively, with the parent node of a cluster, to show that later generation staircases, of a mosaic are also Fibonacci staircases.

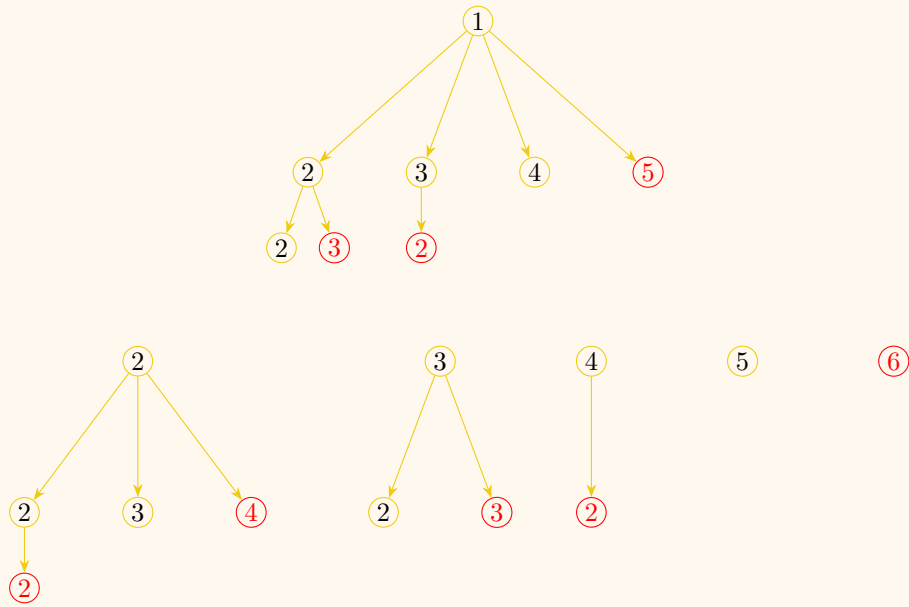


Figure 21: An Enumerative Grove of Magnitude 5 (Magnitude 6 With Red Vertices)

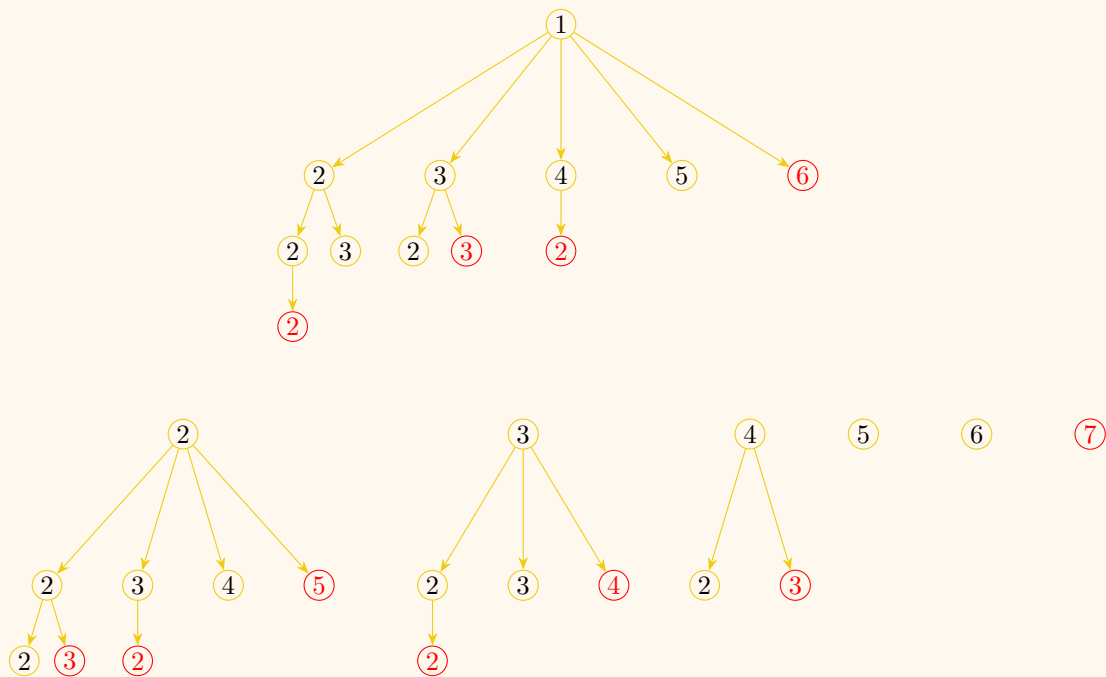


Figure 22: An Enumerative Grove of Magnitude 6 (Magnitude 7 With Red Vertices)

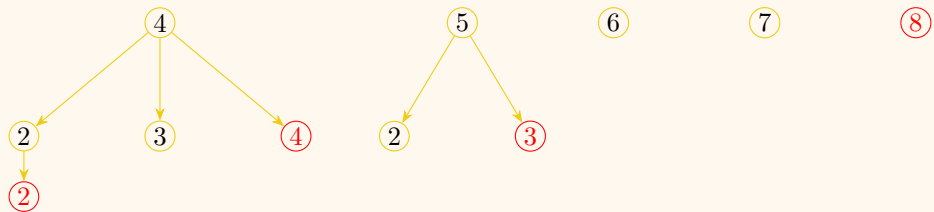
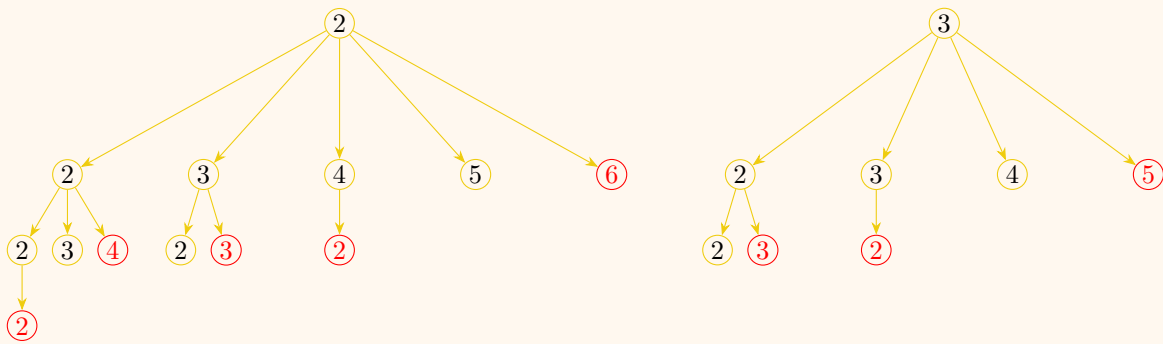
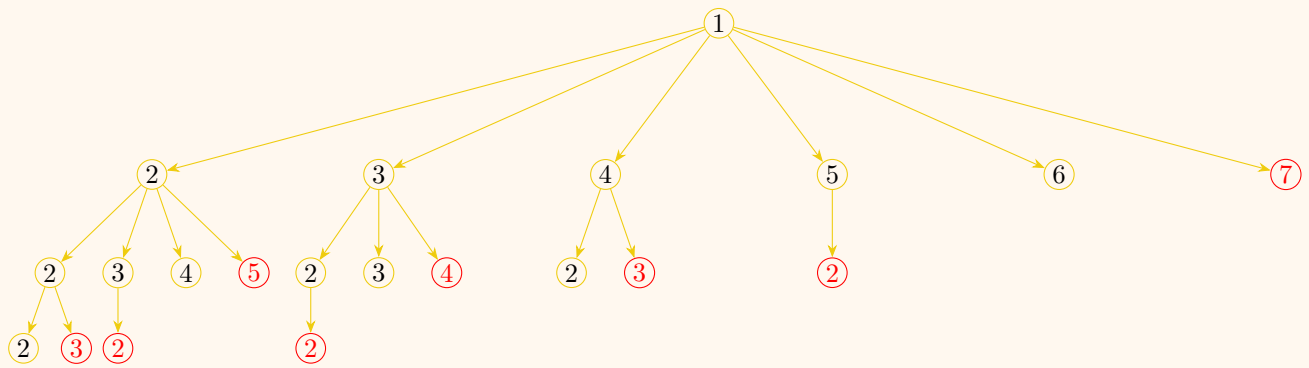


Figure 23: An Enumerative Grove of Magnitude 7 (Magnitude 8 With Red Vertices)

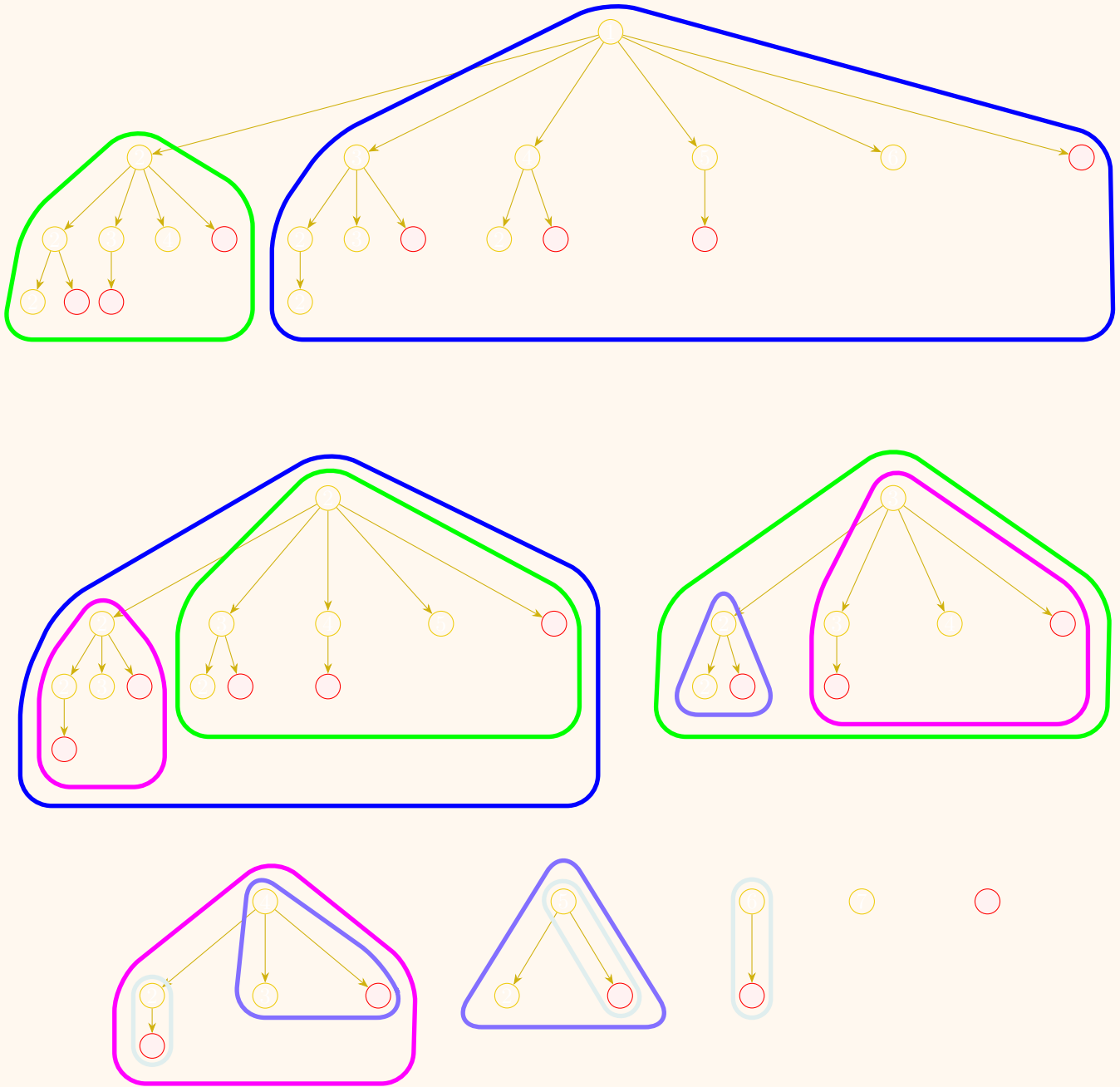


Figure 24: An Enumerative Grove of Magnitude 7 (Magnitude 8 With Red Vertices)

## D A Bounding Estimate *Using Pseudo Gravity*

When set as a universally quantified problem, the Collatz conjecture is undecidable and in the arithmetical hierarchy it is  $\Pi_2^0$ -complete.[Sim07]

Inspired by an Erdős style of approach (by using estimation), a model of a imaginary solar system is created; *escape velocities* are used as an analogy for closed paths. The model is applied to obtain an insight into the decidability of the Collatz conjecture.

### D.1 Modelling an Imaginary Solar System

Imagine a weather system, involving rising and falling imaginary hailstones on another imaginary planet. Let the motion of these hailstones be governed by two weights: one (a positive integer), for the rise factor and the other (a negative integer), for the fall factor. After a given number of moves, of the pseudo hailstone, the weighted rises and falls are summed to determine the [balance](#). If the balance is positive the imaginary hailstone has reached a pseudo escape velocity. Otherwise, the hailstone has landed (it has reached a height of 1). Altering the weights associated with rises and falls sets the pseudo escape velocity. Different pseudo planets would have different pseudo escape velocities.

### D.2 The Effect of Rises and Falls on The Balance

Imagine a height at an some previous location. There has to be twice as many falls as rises for the current height to subside to the level below that of an initial reference height (usually the seed). This is due to the fact that a rise increases a height by factor of three, plus one. Where as, two falls lower the height by a factor of 4. Since, it takes two falls to negate the effect of a single rise, a transform can be created utilizing this fact.

#### D.2.1 A Transform to Encode Balance

When a rise occurs, normalize the move by adding a nominal value of 1 to the balance. When a fall occurs, subtract by one half. To make calculations more amenable for humans, encode as integer values. To do this, double the previous encodings. A rise becomes addition by 2, a fall becomes subtraction by 1. The accuracy of this encoding is conservative (however the trailing 4, 2, 1 path heights eventually sets the balance to  $\leq 0$ , detailed in Section [D.3](#).

#### D.2.2 Tethering Rises to Falls In Ones-Phobic Binaries


An alternate notation is introduced as a visual aid. This is done to more readily recognize changes in the position of rises and falls for a path, especially while comparing paths in a [cascade](#). (Essentially, looking for patterns in an array).



Mono-space fonts improve the readability, yet glyphs improve readability even more. An alternate notation more readily distinguishes a ones-phobic binary (from an ordinary binary). Also, in manual balance calculations, glyphs are easier to distinguish and count. Refer to Figure 18. Here, the glyphs are differently shaded cells.


For example, in a path of seven moves: (rise, fall, rise, fall, fall, fall, fall).


Its ones-phobic binary  $\mathbb{P}_7 \in 1010000$  is alternatively represented as:

●○○○○○.

As a second visual aid, for enumeration, tether each ●○ pair as a single unit (represented as ). This glyph was derived by blurring or blending a ●○ combination). Any extra remaining ○s are regarded as separate units. Then a path of tethered units along with un-tethered ○s can be more easily evaluated.

Now, a rise is represented by ●, which implies addition by 2. Whereas, a fall is represented by ○, which implies subtraction by 1. Since,  = ●○, implies addition by 2 followed by subtraction by 1. Then,  implies addition by 1.

All paths here are ones-phobic (as shown in figure 2), each ● has a following ○, unless the ● occurs last, in a path. Then, the only place a ● could be shown in a path would be at the very end, (any other(s) are being bound inside any  (s)).

The balance at a given location can be obtained by subtracting the number of ○s from the number of s.

### D.2.3 Balance Encoding To Find A Sink Location

Demonstrating how to find a sink location with balance encoding. To show relationships: A path of heights is followed below by corresponding offsets. This in turn is followed by a corresponding ones-phobic binary. Below this are the balance encodings. At the bottom are the corresponding running cumulative subtotals.

Figure 25: Using Balance to Find A Sink Location.

Move	Seed	1	2	3	4	5	6
Heights	67	202	101	304	152	76	38
Offsets	1	6	6	36	36	36	36
Ones Phobic Binary	●	○	●	○	○	○	○
Balance Encoding	2	-1	2	-1	-1	-1	-1
Running Subtotals	2	1	3	2	1	0	-1

By using balance encoding, the sink value can be found without *TP1* or division *DB2*. A running subtotal (bottom shaded row), is calculated using the balance encoded values (upper shaded row). The sink is located where the running subtotal first changes from positive to negative (last column, at move 6). Note: Height 38 is lower than the seed of 67)

#### D.2.4 A Shorter Method to Locate a Sink in a Path

1. **A balance increase**, is caused by alternating ● with ○. The surplus increases by 1 for each ■.

The sequence,

●○●○●○●○

re-represented as,

■■■■ yields a balance of 4.

2. **A local balance reduction**, occurs with each ●○○○ string. It takes three or more successive falls to create a local balance reduction.

That is,

●○●○●○●○○○...

re-represented as,

■■■■○○○... reduces the local balance to 1

3. **Constant Balance**. One or more occurrences of the string ●○○ causes the balance to remain constant. The sub-sequence ●○○ is called a balance maintaining sequence. The balance will be remain the same even if the balance maintaining sequence occurs repeatedly.

If we begin with,

●○ the balance is 1.

Repeatedly append the constant series to form,

●○●○○●○○●○○...●○○

re-represented as,

■■○●○○■■○...■■○ the balance remains as 1.

Using rules i) to iii) to more readily facilitate manual calculations. For example, re-represent the ones-phobic row in figure 25:

■■○○○

The leading ■ (that is ●○), repeats twice yielding a preliminary balance of 2 (as in column 2 of figure 25). Three following ○s reduce the balance to -1.

### D.3 Falls and Rises and Ratios

A description of the contents of the columns of Table 10:

The first column, called the *common seed* list, contains seed values that generate a path containing the intersection of two seed lists: list A and list B. Where, list A contains the seeds of the path length maxima that are larger than any in a previous path. List B contains path height maxima that are larger than any in a previous path. (Both paths are generated from successively increasing seed values.)

The *seed to sink ratio* is the ratio of *rises/falls* in a path as counted from the seed to the sink. The *path ratio* is the ratio of *rises/falls* in a full path (seed to the move whose height is 1).

The following items apply to seed to sink ratios:

- The ratio values for a list of consecutive seed values oscillate.
- The ratio values, for list A, oscillate in a downward trend, and so do the ratios for list B.
- The ratio values, for the common seed monotonically decrease.

The first column labeled *common seed* is also referred to as OEIS entry A276665.

The second column contains the falls of a path (generated by the seed of the first column).

The third column contains the rises of a path (generated by the seed of the first column).

The fourth column contains the *path length* to a move where the balance is zero or negative (for the path generated by the seed of the first column).

The ratio is  $\frac{falls}{rises}$

common seed	full path falls	full path rises	seed to sink falls	seed to sink rises	seed to sink ratio	full path ratio
3	5	2	4	2	2.0000	2.50000
7	11	5	7	4	1.7500	2.20000
27	70	41	59	37	1.5946	1.70732
703	108	62	81	51	1.5882	1.74194
26623	194	113	65	41	1.5854	1.71681

Table 10: Seed Values That Generate a Path Containing Both: a Height Maxima and a Path Length Maxima

During a scan of a seed range of 10 billion, the above table holds the only known *common seed* values. Conjecture: These are the only values of their kind.

From Table 10, the encoding transform can add a ratio for a path rise and

subtract 1 for a path fall. The lowest found ratio that yields a 0 balance is  $\frac{65}{41}$ . Other ratios are lower but they yield a positive balance.

#### D.4 Modelling A Pseudo Solar System (Analogy)

Imagine that pseudo hailstones on pseudo gas giants never escape their pseudo gravitational pull. Conversely, on lighter pseudo moons pseudo hailstones always escape.

These pseudo gravity extrema form a bounding region for an earth like pseudo gravity. Acting on pseudo hailstones. The pseudo gravity of the pseudo planets is modelled by varying the *falls/rises* ratios. For pseudo earth gravity, choosing a ratio to obtain a zero balance (rounded to the nearest integer), depended on: the longest path length in the seed range being used and on the most common path length. The ratio chosen is the rise to fall ratio of 65/41. It is based on the number of moves from seed to sink at the seed 26623. This is the seed value for the last known common seed from Table 10. This ratio consistently gives a balance of zero (rounded). The other ratios are: 1 (Jupiter), 1.5 (Saturn), 2.0 (Mars), 3.5 (Mercury), 9.5 (Ceres). Due to the limitations of computing Inf is set to 10,000.

Intrepreting Table 11: For Pseudo Jupiter, all balances are negative (all hailstone paths return to the surface). The opposite occurs for Pseudo Ceres, all balances are Inf (all hailstone paths escape their pseudo gravity). For Pseudo Ceres all balances are Inf. For Pseudo Earth, all balances are zero (it lies in between the two balance extreems). By using Table 11 to estimate the behaviour hailstone paths, the model indicates that the behaviour equation 1 falls in between the extrema of Pseudo Jupiter and Pseudo Ceres. That is, no definite conclusion (one way or the other, but in between), on how hailstone paths end.

#### D.5 Longer Paths Approach a Ruler Harmonic Frequency Distribution

Observation shows that as paths get longer: the frequency distribution for the path's  $m$  parities approaches the frequency distribution of a ruler harmonic sequence. This is especially true for paths with the next record length maxima. However, cascades contain paths of all parity slide distributions (allowable in a ones-phobic binary). Still, in paths approaching  $m$  parity frequency distributions to those in ruler harmonic sequence: The following calculations show that the balance tends to negative values as higher orders of  $m$  parities are included.

For the following calculations: Let  $u$  be the balance adjustment for a rising move. Let  $d$  be the balance adjustment for a falling move.  $N$  represents the path length, it is normalized to 1 (to compare ratios, instead of arbitrarily large values).

$$u = 65/41$$

$$d = -1$$

$$\frac{(u+d)N}{2} = 0.292682926829$$

$$\frac{(u+d)N}{2} + \frac{(u+2d)N}{4} = 0.189024390243$$

$$\frac{(u+d)N}{2} + \frac{(u+2d)N}{4} + \frac{(u+3d)N}{8} = 0.012195121951$$

$$\frac{(u+d)N}{2} + \frac{(u+2d)N}{4} + \frac{(u+3d)N}{8} + \frac{(u+4d)N}{16} = -0.138719512195$$

$$\frac{(u+d)N}{2} + \frac{(u+2d)N}{4} + \frac{(u+3d)N}{8} + \frac{(u+4d)N}{16} + \frac{(u+5d)N}{32} = -0.245426829268$$

seed	Jupiter	Saturn	Earth	Mars	Mercury	Ceres
3	-4	-1	0	0	Inf	Inf
5	-1	0	0	0	2	Inf
7	-9	-1	0	4	Inf	Inf
9	-1	0	0	0	Inf	Inf
11	-6	-3	0	1	Inf	Inf
13	-1	0	0	0	6	Inf
15	-9	-1	0	1	Inf	Inf
17	-1	0	0	0	Inf	Inf
19	-4	-1	0	0	Inf	Inf
21	-1	0	0	0	2	Inf
23	-6	0	0	1	Inf	Inf
25	-1	0	0	0	Inf	Inf
27	-94	-91	0	Inf	Inf	Inf
29	-1	0	0	0	Inf	Inf
31	-89	-6	0	Inf	Inf	Inf
33	-1	0	0	0	Inf	Inf
35	-4	-1	0	0	Inf	Inf
37	-1	0	0	0	2	Inf
39	-11	0	0	20	Inf	Inf
41	-1	0	0	0	Inf	Inf
43	-6	-3	0	4	Inf	Inf
45	-1	0	0	0	Inf	Inf
47	-86	-8	0	Inf	Inf	Inf
49	-1	0	0	0	Inf	Inf
51	-4	-1	0	0	Inf	Inf
53	-1	0	0	0	2	Inf
55	-6	0	0	Inf	Inf	Inf
57	-1	0	0	0	Inf	Inf
59	-9	-6	0	1	Inf	Inf
61	-1	0	0	0	Inf	Inf
63	-86	-73	0	Inf	Inf	Inf
65	-1	0	0	0	Inf	Inf

Table 11: Pseudo Escape Velocities

Program listings to generate Table 11 follow:

Listing 1: An Algorithm Generating a Hailstone Path Where Rises and Falls are Counted for A Balance

```
1 using Printf
2 function hs(seed,ratio)
3     stone = seed
4     move = 0
5     balance = 0.0
6     rise = 0
7     fall = 0
8     found_sinkloc = false
9     found_landing = false
10    landing = 0
11    sinkloc = 0
12
13    while true
14        move +=1
15        if mod(stone,2) == 0
16            stone >>=1
17            balance -=1.0 #41
18            fall +=1
19        elseif mod(stone,2) == 1
20            stone = stone+stone+stone+1
21            balance +=ratio #65
22            rise +=1
23        end
24        if stone <= seed && !found_sinkloc
25            sinkloc = move
26            found_sinkloc = true
27        end
28        if balance <= 0.0 && !found_landing
29            landing = move
30            found_landing = true
31        end
32        if found_sinkloc && found_landing
33            return (seed, balance, sinkloc, landing)
34        end
35        if move >= 10_000
36            if !found_sinkloc
37                return (seed, balance, Inf, landing)
38            elseif !found_landing
39                return (seed, balance, sinkloc, Inf)
40            end
41        end
42    end
43 end
```

Listing 2: Table Generator Showing The Effect of Ratios for Pseudo Planets

```

1
2 using Printf
3 #= Note some seed values are more sensitive to changes in pseudo gravity than
   others. Compare seeds 21 to 27 =#
4
5 @printf("(Relative Heights)  Analogy Uses Various Celestial Masses\
   n")
6 @printf("seed  Jupiter Saturn Earth  Mars  Mercury  Ceres \n")
7 for x in 3:2:1024
8     ra = hs(x,1.0)           #Jupiter (analogy)
9     a = ra[4]-ra[3]
10    rb = hs(x,1.5)           #Saturn (analogy)
11    b = rb[4]-rb[3]
12    rc = hs(x,65.0/41.0)     #Earth (analogy)
13    c = rc[4]-rc[3]
14    rd = hs(x,2.0)           #Mars (analogy)
15    d = rd[4]-rd[3]
16    re = hs(x,3.5)           #Mercury (analogy)
17    e = re[4]-re[3]
18    rf = hs(x,9.5)           #Ceres (analogy)
19    f = rf[4]-rf[3]
20    @printf(
21    " -4d  -4d      -4d  -3d      -4d  -4d
   -4d%n",x,ra[4],rb[4],rc[4],rd[4],re[4],rf[4])end

```

## E The Binary Approach

### E.1 Establishing Ruler Harmonics for Offsets of Six

There are regular patterns in the binary of offsets of six. Successive binaries are aligned as a column. Columns within the main column are colour coded. The least significant column (tangerine bits), is always zero. The next significant column (red bits) follows a repeating pattern, whose binary values are: 3, 2, 1, 0. Note how this corresponds to Figure 28. (Same continuous pattern, except when grouped by clusters of four, the cluster origin is at 2 instead of 3.) The pattern in blue bits is similar to the pattern in the previous column. Except that the digits are in reverse order. Also, an extra digit is repeated highlighted in (green). That is:

0, 1, 2, 3, **3**   0, 1, 2, **2**, 3   0, 1, **1**, 2, 3   0, **0**, 1, 2, 3

In the most significant column, the binary value of the bit pair increase by 1 after five repetitions (similar to an odometer).

The cyclic pattern of the red column determines the frequency and occurrence of the most common parity (the smallest parity). (Observe from Figure 26 that the most frequently occurring patterns are here.) Two columns of bits are colour coded with a single colour. The bit pair of the same colour acts like a gate if there is a 1 the parity is determined at that point. If the bit pair is 00 the parity is determined by more significant bits (proceeding leftward). Moving on, refer to the next significant column (black). Form a subset of Figure 26 by using only **00** are highlighted in light blue.) Note how the blue column determines the parity lengths in this subset (since the red and yellow columns only contain zeroes). Here also, a shorter parity occurs twice as often as it's next larger parity. Note to summarize, these results agree with Figure 7.

Figure 26: Multiples of 6 with their binary representation

6	00	00	11	0
12	00	01	10	0
18	00	10	01	0
24	00	11	00	0
30	00	11	11	0
36	01	00	10	0
42	01	01	01	0
48	01	10	00	0
54	01	10	11	0
60	01	11	10	0
66	10	00	01	0
72	10	01	00	0
78	10	01	11	0
84	10	10	10	0
90	10	11	01	0
96	11	00	00	0
102	11	00	11	0
108	11	01	10	0
114	11	10	01	0
120	11	11	00	0
126	11	11	11	0
132	00	00	10	0
138	00	01	01	0
144	00	10	00	0
150	00	10	11	0
156	00	11	10	0
162	01	00	01	0
168	01	01	00	0
174	01	01	11	0
180	01	10	10	0
186	01	11	01	0
192	10	00	00	0
198	10	00	11	0
204	10	01	10	0
210	10	10	01	0
216	10	11	00	0
222	10	11	11	0
228	11	00	10	0
234	11	01	01	0
240	11	10	00	0
246	11	10	11	0
252	11	11	10	0
258	00	00	01	0

## E.2 Distribution of $L$ parity in positive integers.

**Definition E.1** ( $L$  parity). This term denotes the short form for *look ahead parity*. The precursor odd height that, as input to TP1, generates the succeeding height having an  $m$  parity. This succeeding  $m$  parity is relabelled as an  $L$  parity of the preceding odd height (which actually refers to the number of trailing binary zeroes in the succeeding even height).

## E.3 Relating L parity to Offsets Whose base is Six

Showing frequency distribution of L parities from a sequential odd integer sequence. (Refer to Figure 30). Note the middle column, these are all offsets of 6. Also note, compare this parity distribution with that indicated by the blue scale marks of Figure 7. Offset shifting one ruler and superimposing it over the second ruler will reveal an exact overlap. In other words one  $m$  parity sequence can be found located from an index offset of the other  $m$  parity sequence. Both are infinite sequences. Both of these sequences are defined to have *second order subharmonics*. Refer to Figure 29 specifically to the columns with numerals in black. The first of these columns (numerals in black), shows ruler harmonics. The second such column shows second order ruler subharmonics. Following columns (again in black numerals), shows *higher order ruler subharmonics*.

### E.3.1 Frequency of $L$ parity for a given $m$ in $B_n$

Since  $L$  parity is based on  $m$  parity the same previous argument (section 4.1), can be used to show that there are  $2^k$  times as many values having  $L$  parity as there are having  $(L+1)$  parity.

### E.3.2 Leading and Trailing Parity

Defining: *leading parity* as the number of leading consecutive zeroes in a binary value. While, *trailing parity* is the number of trailing consecutive zeroes in a binary value. Recall from 4.1 that  $B_n$  contains all of the consecutive integers from  $00 \dots 00$  to  $11 \dots 11$  (all of  $n$  bits). The set  $B_n$  has the property of mirror image symmetry. That is:

**Theorem E.3.3.** *Each element  $x \in B_n$  has a mirror image, (sometimes itself as a mirror image). Implying that the frequency distributions are the same for leading parity and trailing parity.*

*Proof.* Choose an arbitrary  $x \in B_n$ . Let it's mirror image be called  $'x$ . If  $x$  is  $0 \dots 0$  or  $1 \dots 1$  then  $x$  is a mirror image of itself. If  $x$  lies in between  $0 \dots 0$  and  $1 \dots 1$ . Then reversing the bits would form a binary of length  $n$  that lies in between  $0 \dots 0$  and  $1 \dots 1$ . Thus,  $'x \in B_n$ .

Since,  $\forall x \in B_n, \exists 'x \in B_n$ ; each *leading parity* has it's *trailing parity* counter part. Thus, the frequency distributions are the same for *leading parity* and *trailing parity*.  $\square$

Figure 27: An Icicle Sector, Abridged From Figure 28.

index	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
(mod 2)		0		0		0		0		0		0		0		0
(mod 4)				0				0				0				0
(mod 8)								0								0
(mod 16)																0
(mod 32)																0
(mod 64)																0
mparity	0	1	0	2	0	1	0	3	0	1	0	2	0	1	0	6

The *icicle sector* is a section of an array shown in Figure 28. It is delineated by the length of the pivot row (the landing bordered in red). The icicle sector includes the pivot row landing along with all the landings above that are confined within the span of the pivot row landing. Icicles are defined as the part of a column that holds the placement(s) that are zero (highlighted in light blue, here and in Figure 28). Icicles extend downwards but stop before the pivot row. (Only the highlighted light blue cells have placements that are zero in value.) Landings in rows, (mod 2) through (mod 8), are bordered in cyan. The traversal, is a type of icicle distinguished as the remaining column of zero valued placements that reaches, or optionally passes, the landing of the pivot row. A traversal forms the right boundary of an icicle sector. Note: landings are delineated in a way that the last placement of a landing has a zero value. Observe that an icicle (or traversal) is at the right boundary of a landing for each of the rows that the icicle (or traversal) intersect.

To better elucidate the concepts of icicles and traversals, the non-zero placements (shown in Figure 28) are excluded to reduce visual obfuscation.

index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
(mod 2)	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
(mod 4)	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0
(mod 8)	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0
(mod 16)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0
(mod 32)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	0
(mod 64)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
<i>m</i> parity	0	1	0	2	0	1	0	3	0	1	0	2	0	1	0	4	0	1	0	2	0	1	0	3	0	1	0	2	0	1	0	5

index	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
(mod 2)	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
(mod 4)	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0
(mod 8)	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0
(mod 16)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0
(mod 32)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	0
(mod 64)	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	0
<i>m</i> parity	0	1	0	2	0	1	0	3	0	1	0	2	0	1	0	4	0	1	0	2	0	1	0	3	0	1	0	2	0	1	0	6

Figure 28: Showing How Modulo Arithmetic Can Generate Ruler Harmonics

Modulo  $2^n$  ( $n \in \mathbb{N}_1$ ) arithmetic is applied to the values of the main index (red row), which forms a least residue system of  $\{1, 2, \dots, 2^{n-1}, 0\}$ . The elements of this least residue system when contained within a landing (and ordered, as illustrated above) are called placements. Landings are shown with cyan borders (horizontal rectangles). Each landing holds a least residue system (ordered placements), for each row labeled as (mod  $2^n$ ). These placements, formed by modulo  $2^n$  arithmetic, tessellate row-wise from one landing to the next. By design,  $n$  increases by 1, for each following row. Each landing is as long as exactly two landings as those in the previous row, above it. Since:

$$2 \cdot 2^n = 2^{n+1}$$

The number of the rows of (mod  $2^n$ ), and columns, can be indefinitely extended (by symmetry and the properties of modulo arithmetic). The table columns are also formed by breaking down the *mp\_modular* function (in appendix G), into intermediate results (like those used in debugging a function). The function sums up the number of zero placements of a column to yield the *m* parity of its corresponding index (the functions input).

For this table, the columnar sum of zero placements is highlighted in the corresponding cell of the *m* parity row. This is expressed as:

$$\text{counter :} \quad k(x, pt) = \begin{cases} 1 & \text{if } x \bmod pt = 0, \\ 0 & \text{otherwise.} \end{cases}$$

$$\text{m parity :} \quad mp(x) = \sum_{y=1}^{k(x, 2^y) \neq 1} k(x, 2^y)$$

$$\text{where :} \quad x, y, pt \in \mathbb{N}_1$$

This definition is not as straight forward as counting the number of 0's (from the binary version of an index). However, this definition, using modulo arithmetic, is used to show how subsets of the sequence (the index row), generates sequences of *m* parities. The index subsets are selected from the index of every successive  $k$ 'th value, where  $k$  is a constant increment, such that  $k \in \mathbb{N}_1$ . Note: An increment of  $k = 1$ , returns the index. Look at Figure 27 for a key to this diagram.

Figure 29: Ruler Harmonics in Powers of Six With Landings and Placements

$6^0x$		placement	$6^1x$		placement	$6^2x$		placement	$6^3x$		placement	$6^4x$		placement
$m$	parity	(mod 4)	$m$	parity	$\frac{6^1x \pmod{8}}{2}$	$m$	parity	$\frac{6^2x \pmod{16}}{4}$	$m$	parity	$\frac{6^3x \pmod{32}}{8}$	$m$	parity	$\frac{6^4x \pmod{64}}{16}$
1	0	1	6	1	3	36	2	1	216	3	3	1296	4	1
2	1	2	12	2	2	72	3	2	432	4	2	2592	5	2
3	0	3	18	1	1	108	2	3	648	3	1	3888	4	3
4	2	0	24	3	0	144	4	0	864	5	0	5184	6	0
5	0	1	30	1	3	180	2	1	1080	3	3	6480	4	1
6	1	2	36	2	2	216	3	2	1296	4	2	7776	5	2
7	0	3	42	1	1	252	2	3	1512	3	1	9072	4	3
8	3	0	48	4	0	288	5	0	1728	6	0	10368	7	0
9	0	1	54	1	3	324	2	1	1944	3	3	11664	4	1
10	1	2	60	2	2	360	3	2	2160	4	2	12960	5	2
11	0	3	66	1	1	396	2	3	2376	3	1	14256	4	3
12	2	0	72	3	0	432	4	0	2592	5	0	15552	6	0
13	0	1	78	1	3	468	2	1	2808	3	3	16848	4	1
14	1	2	84	2	2	504	3	2	3024	4	2	18144	5	2
15	0	3	90	1	1	540	2	3	3240	3	1	19440	4	3
16	4	0	96	5	0	576	6	0	3456	7	0	20736	8	0
17	0	1	102	1	3	612	2	1	3672	3	3	22032	4	1
18	1	2	108	2	2	648	3	2	3888	4	2	23328	5	2
19	0	3	114	1	1	684	2	3	4104	3	1	24624	4	3
20	2	0	120	3	0	720	4	0	4320	5	0	25920	6	0
21	0	1	126	1	3	756	2	1	4536	3	3	27216	4	1
22	1	2	132	2	2	792	3	2	4752	4	2	28512	5	2
23	0	3	138	1	1	828	2	3	4968	3	1	29808	4	3
24	3	0	144	4	0	864	5	0	5184	6	0	31104	7	0
25	0	1	150	1	3	900	2	1	5400	3	3	32400	4	1
26	1	2	156	2	2	936	3	2	5616	4	2	33696	5	2
27	0	3	162	1	1	972	2	3	5832	3	1	34992	4	3
28	2	0	168	3	0	1008	4	0	6048	5	0	36288	6	0
29	0	1	174	1	3	1044	2	1	6264	3	3	37584	4	1
30	1	2	180	2	2	1080	3	2	6480	4	2	38880	5	2
31	0	3	186	1	1	1116	2	3	6696	3	1	40176	4	3
32	5	0	192	6	0	1152	7	0	6912	8	0	41472	9	0

The blue columns are multiples of powers of six. The black columns list the  $m$  parity of the corresponding integer (adjacent left blue column). A green value is the relative available position (placement) within a landing.

Note: The  $m$  parities increase by one, when going across any row, from one power of six to the next power of six. Local  $m$  parity maxima of a column (*traversals*), are in red numerals.

The placements within four consecutive landings are demarcated by alternate background shading, which also delineates the least residue system (mod 4), whose order alternates between ascending and descending from one green column to the next.

## F Addendum of Observations

To better illustrate  $m$  parity formation, under various criteria, tables are included as examples.

### F.1 Tabulating Results: Multiplying Index by $2^k$ With Varying Origin

Here we discuss  $m$  parity sequences that are generated from arithmetic progressions whose common difference is  $2^n$ .

#### F.1.1 Increment Advance by 2

Starting with an increment of 2 which returns every second element of the index. An origin of 1 causes the increment to jump to indices of: 1, 3, 5,  $\dots$ . The  $m$  parity function on this resulting sub-index returns the top row in the table below. An origin of 2 causes the increment to jump to indices of: 2, 4, 6,  $\dots$ . The  $m$  parities of these indices is shown in the bottom row of the table below.

Here are the tabulated results of the  $m$  parity function, when varying the origin. Only showing (in red), the indices (for the row of  $m$  parities whose origin is 1).

	index, n								
	1	3	5	7	9	11	13	15	
origin	m parities								
1	0	0	0	0	0	0	0	0	...
2	1	2	1	3	1	2	1	4	...

The top row (row 1, as denoted by the origin), is a constant sequence consisting of the constant zero. The generalization of row 1 is proved in Section 7.1.4. The bottom row (row 2), is defined as  $rhs^1$ . The generalization, of row 2, is proved in Lemma 7.1.5 Note the origin determines whether the result of the  $m$  parity function is a constant sequence, or one that follows ruler subharmonics.

#### F.1.2 Increment Advance by 4

Here are the tabulated results of the  $m$  parity function, when varying the origin. Only showing (in red), the indices (for the row of  $m$  parities whose origin is 1).

		index, n							
		1	5	9	13	17	21	25	29
origin		m parities							
1		0	0	0	0	0	0	0	...
2		1	1	1	1	1	1	1	...
3		0	0	0	0	0	0	0	...
4		2	3	2	4	2	3	2	5

The sequences of rows 1 through 3 are constant sequences. Rows 1 and 3 consist of the constant zero. While row 2 consists of the constant 1. Proof of the generalization of constant sequences is given in Section 7.1.4. The sequence in the bottom row, having a origin of 4, is defined as  $rhs^2$ . Proof of the generalization, of this, is given in Lemma 7.1.5.

### F.1.3 Increment Advance by 8

Again, tabulated here are the results of the  $m$  parity function, when varying the offset origin. Only showing (in red), the indices (for the row of  $m$  parities whose origin is 1).

		index, n							
		1	9	17	25	33	41	49	57
origin		m parities							
1		0	0	0	0	0	0	0	...
2		1	1	1	1	1	1	1	...
3		0	0	0	0	0	0	0	...
4		2	2	2	2	2	2	2	...
5		0	0	0	0	0	0	0	...
6		1	1	1	1	1	1	1	...
7		0	0	0	0	0	0	0	...
8		3	4	3	5	3	4	3	6

While rows 2,6 consist of the constant 1; rows 2,3,5 and 7 consist of the constant 0. Row 4 consists of the constant 2. Proof of the generalization of constant sequences (rows), is given in Section 7.1.4.

Row 8, is defined as a ruler harmonic sequence, of the third order. Proof of the generalization, of this, is given in Lemma 7.1.5.

Note in the tables of Section F.1.1 to Section F.1.3, the blue *columns* of  $m$  parities express the initial subsequence of a ruler harmonic sequence, of the 0th order (namely, the standard sequence  $S$ ). While the other columns, as a subsequence, match their maxima to the first same value in a subsequence of  $S$ , (in 5). The remaining values, of the columnar subsequences, are respectively matched backwards towards the origin.

Why the table columns are  $\lesssim \mathbb{S}$ . Although, each column has a different index origin; each successive columnar entry has the index increase by one. This makes each column of the type:

$$\left( mp(n) \right)_{n \in \mathbb{N}_j} \quad (63)$$

Where  $j$  initially is the **index** origin. Pertaining to the columns, each initial value of  $j$  is set to the **index** value of the top column element. While successive values of  $j$ , within its column, have a common difference of 1.

## F.2 The $m$ Parities of an Index Multiplied by a Power of Three

	Index (red row, selecting every $3^n$ th value) $m$ parity (blue row)									
multiplier										
$3^0$	1 0	2 1	3 0	4 2	5 0	6 1	7 0	8 3	9 0	...
$3^1$	3 0	6 1	9 0	12 2	15 0	18 1	21 0	24 3	27 0	...
$3^2$	9 0	18 1	27 0	36 2	45 0	54 1	63 0	72 3	81 0	...

## F.3 Aside: Illustrating the Effect of Sequential Multiplication of a Constant and/or Varying the Origin, on Sequences, by Using Tables

These are illustrated, by example, with the following tables: one for an index common difference of six, the other for an index common difference of 36; both show the types of sequences generated, as the offset from the index origin varies. Refer to Table 12 and Table 13. We can compare Table 12 and Table 13, respectively with the tables in Section F.1.1 and Section F.1.2. The rows of constant sequences of zeroes in the table of Section F.1.1 correspond with the rows of zeros in table 12. Likewise, the rows of constant sequences of zeros along with rows of ones, in Section F.1.2 correspond with rows of zeros and ones in Table 13. The table in Section F.1.3 extends the results of the previous tables of Sections F.1.1 and F.1.2. The blue rows of Table 12 are  $rhs^1$  sequences. The blue rows of Table 13 are  $rhs^2$  sequences.

Figure 30: Ruler harmonics, for m parities and L parities.

(a) m parities		(b) L parities		
Even Seed	m Parity	Odd Seed	TP1 Result	L Parity
2	1	3	<b>10</b>	1
4	2	5	<b>16</b>	4
6	1	7	<b>22</b>	1
8	3	9	<b>28</b>	2
<b>10</b>	1	11	<b>34</b>	1
12	2	13	40	3
14	1	15	46	1
<b>16</b>	4	17	52	2
18	1	19	58	1
20	2	21	64	6
<b>22</b>	1	23	70	1
24	3	25	76	2
26	1	27	82	1
<b>28</b>	2	29	88	3
30	1	31	94	1
32	5	33	100	2
<b>34</b>	1	35	106	1

Note: The *TP1 Result* column, these are all offsets of 6. That is, they are every third entry of the *Even Seed* column (shown in red).

Table 12: Varying the Origin for Index Common Difference of Six.

origin	Index Values (Add origin for indices at a given row.)																	
	0	6	12	18	24	30	36	42	48	54	60	66	72	78	84	90	m parities	
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
2	1	3	1	2	1	5	1	2	1	3	1	2	1	4	1	2		
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
4	2	1	4	1	2	1	3	1	2	1	6	1	2	1	3	1		
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
6	1	2	1	3	1	2	1	4	1	2	1	3	1	2	1	5		
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
8	3	1	2	1	5	1	2	1	3	1	2	1	4	1	2	1		
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
10	1	4	1	2	1	3	1	2	1	6	1	2	1	3	1	2		
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
12	2	1	3	1	2	1	4	1	2	1	3	1	2	1	5	1		
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
14	1	2	1	5	1	2	1	3	1	2	1	4	1	2	1	3		
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
16	4	1	2	1	3	1	2	1	6	1	2	1	3	1	2	1		
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
18	1	3	1	2	1	4	1	2	1	3	1	2	1	5	1	2		
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
20	2	1	5	1	2	1	3	1	2	1	4	1	2	1	3	1		
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
22	1	2	1	3	1	2	1	6	1	2	1	3	1	2	1	4		
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
24	3	1	2	1	4	1	2	1	3	1	2	1	5	1	2	1		
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
26	1	5	1	2	1	3	1	2	1	4	1	2	1	3	1	2		
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
28	2	1	3	1	2	1	6	1	2	1	3	1	2	1	4	1		
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
30	1	2	1	4	1	2	1	3	1	2	1	5	1	2	1	3		
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
32	5	1	2	1	3	1	2	1	4	1	2	1	3	1	2	1		
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
34	1	3	1	2	1	6	1	2	1	3	1	2	1	4	1	2		
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
36	2	1	4	1	2	1	3	1	2	1	5	1	2	1	3	1		
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
38	1	2	1	3	1	2	1	4	1	2	1	3	1	2	1	7		
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
40	3	1	2	1	6	1	2	1	3	1	2	1	4	1	2	1		
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
42	1	4	1	2	1	3	1	2	1	5	1	2	1	3	1	2		
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
44	2	1	3	1	2	1	4	1	2	1	3	1	2	1	7	1		
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
46	1	2	1	6	1	2	1	3	1	2	1	4	1	2	1	3		
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
48	4	1	2	1	3	1	2	1	5	1	2	1	3	1	2	1		

Table 13: Varying the Origin for Index Common Difference of 36

origin	Index Values (Add origin for indices at a given row.)															
	0	36	72	108	144	180	216	252	288	324	360	396	432	468	504	540
	m parities															
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	2	3	2	4	2	3	2	5	2	3	2	4	2	3	2	5
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	2	4	2	3	2	5	2	3	2	4	2	3	2	9	2
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	2	4	2	3	2	6	2	3	2	4	2	3	2	5	2	3
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	4	2	3	2	5	2	3	2	4	2	3	2	6	2	3	2
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	2	3	2	7	2	3	2	4	2	3	2	5	2	3	2	4
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	3	2	5	2	3	2	4	2	3	2	7	2	3	2	4	2
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	2	6	2	3	2	4	2	3	2	5	2	3	2	4	2	3
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	5	2	3	2	4	2	3	2	6	2	3	2	4	2	3	2
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	2	3	2	4	2	3	2	5	2	3	2	4	2	3	2	6
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	3	2	4	2	3	2	8	2	3	2	4	2	3	2	5	2
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
42	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
44	2	4	2	3	2	5	2	3	2	4	2	3	2	9	2	3
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
46	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
48	4	2	3	2	6	2	3	2	4	2	3	2	5	2	3	2

## G Two Algorithms To Evaluate $m$ Parity

We show that both algorithms yield the same result. (The source code uses the julia language. A description follows the source code.)

Listing 3: Algorithm to obtain  $m$  parity (bitwise approach)

```
1 # Input is a positive integer (>0)
2 function mparity_bitwise(height)
3     p = 0
4     while height & 1 == 0
5         height >>= 1
6         p += 1
7     end
8     return p
9 end
```

Listing 4: Algorithm to obtain  $m$  parity (modular arithmetic approach)

```
1 """
2 Input is a positive integer (>0)
3 Returns the m parity of an integer called height.
4 """
5 In[]: mparity_modular(4)
6 Out[]: 2
7 """
8 """
9 function mparity_modular(height)
10     p = 1
11     while mod(height, 2^p) == 0
12         p += 1
13     end
14     return p - 1
15 end
```

### G.1 Description of The Source Code

#### G.1.1 How the `mp_bitwise` Function Yields $m$ Parity.

The first algorithm directly counts the number of trailing zeroes of the input. Here is how:

In the first iteration of the `mparity_bitwise` function, the trailing bit of the input is determined by the result of a Boolean AND, using 1, on the least significant bit of the input. If the result is zero, the  $m$  parity counter is incremented, by 1, and the input is right shifted for the next iteration. If the result is 1, the last of the trailing zeros has been counted, and the function returns the  $m$  parity.

#### G.1.2 How the `mp_modular` Function Yields $m$ parity.

In the first iteration of the `mparity_modular` function, we set  $p = 1$  and if we obtain the result,  $0 \equiv (\text{mod } 2^p)$ , then the  $m$  parity counter is incremented by 1,

and  $p$  is incremented by 1, for the next iteration. If the result is  $\not\equiv (\text{mod } 2^p)$ , then the last of the trailing zeros has been counted, and the function returns the  $m$  parity. Since,  $(\text{mod } 2^n)$  obtains the remainder after division by  $2^p$ . A zero remainder implies divisibility by  $2^p$ , which means  $2^p$  is a factor of the input, this in turn implies that there are at least  $p$  trailing zeroes in the input. The *mparity\_modular* function sums up all of the zero remainders from division by  $2^p$  where  $p \in \mathbb{N}_1$ . This sum is also the highest value of  $p$  of  $2^p$  that divides the input. Therefore, this sum determines the  $m$  parity.

In Figure 28 we employ the *mparity\_modular* function to generate a table. The running subtotals of the function are used to generate the rows of the tables.

Although the two algorithms are different, they both return the same  $m$  parity of an input.

Listing 5: Procedure to establish whether a subsequence is a ruler harmonic sequence of order zero.

```

1
2 """
3 The function to check whether or not a sequence is a ruler harmonic
   sequence of order zero.
4
5 The standard sequence. A finite implementation of the ruler
   harmonic sequence of order zero:
6 S = "0, 1, 0, 2, 0, 1, 0, 3, 0, 1, 0, 2, 0, 1, 0, 4"
7
8 The two sequences, used to establish whether either is a ruler
   harmonic sequences of order zero:
9 T = "0, 1, 0, 3"
10 U = "0, 1, 1, 3"
11
12 '''
13 iso_check(S,T)
14 "A rule harmonic sequence of zero order."
15
16 iso_check(S,U)
17 "Something else."
18 '''
19 """
20
21 function iso_check(STD,SUB)
22 if occursin(SUB,STD)
23     println("A rule harmonic sequence of zero order.")
24 else
25     println("Something else.")
26 end

```

Listing 6: CS and RHS Frequency Distributions of Each Order At a Given Move From a Cascade.

```

1
2 The source code to verify the entries of Table 3 and Table 4.
3
4 using Printf
5
6 """

```

```

7     height_mparity_at_move(seed,path_length)
8
9     Returns the height and its mparity for a given move of a path
10    with a given seed value.
11    '''
12    In[]: height_mparity_at_move(8,2)
13    Out[]: 1
14    '''
15    """
16    function height_mparity_at_move(seed,path_length)
17        height = seed
18        at = 0
19        # hailstone routine
20        while at < path_length
21            if mod(height,2) == 0
22                height >>= 1
23            elseif mod(height,2) == 1
24                height = height+height+height+1
25            end
26            at += 1
27        end
28        return mparity_modular(height), height # described in Listing 4
29    end
30
31
32    """
33    cascade_column(first_seed, seed_range, path_length)
34
35    Returns a tuple consisting of a column from a height
36    cascade and a column from an mparity cascade. The
37    position of the columns returned is set by the path
38    length. The input is the seed value of the initial
39    cascade path, the seed range (the number of paths in
40    the cascade), and the path length.
41    '''
42    In[]: height_mparity_cascades(1, 8, 4)
43
44    Out[]:
45    (Any[0, 1, 4, 2, 2, 0, 1, 0], Any[1, 2, 16, 4, 4, 5, 34, 1])
46    '''
47    """
48    function cascade_column(first_seed, seed_range, path_length)
49        CASCADE_MPARITIES = []
50        CASCADE_HEIGHTS = []
51        for s in range(first_seed,length=seed_range)
52            MH = height_mparity_at_move(s, path_length)
53            push!(CASCADE_MPARITIES, MH[1])
54            push!(CASCADE_HEIGHTS, MH[2])
55        end
56        return CASCADE_MPARITIES, CASCADE_HEIGHTS
57    end
58    """
59    frequency_distribution(move, COL)
60
61    Looks at two consecutive flutes within a column. If the
62    respective phase shifts from the flute pair are equal then
63    these phase shifts are part of an arithmetic progression

```

```

64     whose type is a constant sequence. Otherwise, the phase
65     shifts are part of an arithmetic progression whose type is
66     a ruler harmonic sequence. After the sequence type and
67     order are established the outcome is incremented to the
68     corresponding vector which is then output as a tuple. Each
69     of these vectors is then output as a frequency distribution
70     by rank (or order).
71 '''
72 In []: move = 5
73         HMPC = cascade_column(1, 2^(move+1), move)
74         frequency_distribution(move, HMPC[1])
75
76 Out[]: Rank Order  0   1   2   3   4
77         CS 10    8   1   0   0
78
79         Rank Order  0   1   2   3   4
80         RHS  1   5   6   1   0
81 '''
82 """
83 function frequency_distribution(move, COL)
84     CS = zeros(Int,16) # cs histogram data
85     RHS = zeros(Int,16) # rhs histogram data
86     fl = 2^move
87     for phase_shift in range(1,fl)
88         if COL[phase_shift] == COL[phase_shift+fl]
89             CS[COL[phase_shift]+1] += 1
90         elseif COL[phase_shift] != COL[phase_shift+fl]
91             RHS[min(COL[phase_shift], COL[phase_shift+fl])+1] +=
1
92         end
93     end
94
95     rank_order=[x for x in range(0,move+1)]
96
97     print("Rank Order")
98     for x in rank_order[1:move]
99         @printf("%3d ",x)
100     end
101     println(" ")
102     print(" ^8,"CS")
103     for x in CS[1:move]
104         @printf("%3d ",x)
105     end
106
107     println(" ")
108     println(" ")
109     print("Rank Order")
110     for x in rank_order[1:move]
111         @printf("%3d ",x)
112     end
113     println(" ")
114     print(" ^7,"RHS")
115     for x in RHS[1:move]
116         @printf("%3d ",x)
117     end
118 end

```

## Glossary

**$m$  parity** Where  $m$  is number of consecutive trailing zeroes from the least significant end of the binary representation of an integer ( $\in \mathbb{N}_1$ ), which is called it's  $m$  parity (for multiple parity). Note,  $0$  parity is an integer with no trailing zeros,  $1$  parity is an integer with 1 trailing zero, etc . [1](#)

**$m$  parity clamp** Look at two integers in their binary form. Suppose the pair of integers have different  $m$  parities. Should the addend have a lesser  $m$  parity than the augend; then the sum of both will have the  $m$  parity of the addend. The addend is said to be an  $m$  parity clamp . [10](#)

**DV2** If the last integer in a sequence is even, then divide that integer by two. Then append it to the sequence. [1](#), [54](#)

**TP1** If the last integer in a sequence is odd, then multiply that integer by three and add one. Then append it to the sequence. [1](#), [54](#)

**arithmetic progression** An arithmetic progression, or arithmetic sequence, is a monotonically increasing sequence of positive integers that have the same (or common) difference between successive elements. It's origin and common difference, with regards to ruler harmonics, are in  $\mathbb{N}_1$ . [6](#), [7](#)

**balance** The *Balance* is a measure that compares the number of rises and falls between a given move and the seed value. One weight is associated with a rise. One other weight is associated with a fall. At a given move (with regards to its height manifestation), the balance is a measure of how many falls are still needed to subside below (or reach), the height of an earlier given move (usually the seed height). The move where the balance first becomes negative, or zero, is called the *sink* (usually with reference to the seed). An example is given in Figure [25](#).. [77](#)

**branch** On a disjoint directed graph of trees (an enumerative grove), within any given tree: a *branch* is a walk from a root vertex to a leaf vertex, where each vertex has an associated node that contains a value which is the length of a parity slide. The joined ordered sequence of these parity slides forms a ones phobic binary. Each *branch*, of a tree in an enumerative grove, forms a permutation of parity slides that constitute a distinct ones phobic binary. [70](#)

**canonical bean form** Conceive of a ones-phobic binary consisting of beans and un-tethered zeros. Suppose all of the beans are contiguously grouped together at the end having the lesser significant positions. The contiguous beans immediately could optionally be followed by one and only one un-tethered zero which occupies the least significant position. A ones-phobic binary in this state is said to be in a canonical bean form. [66](#)

**cascade** A cascade is a two dimensional array of "hailstone" paths. All of the paths contain  $n$  moves. These paths are said to be of length  $n$ . The paths in the cascade are initiated by successive seed values. The number of paths is  $2^n$ . Consequently, the dimensions of a cascade are given as:  $2^n$  by  $n$ . In some illustrations of cascades an extra path is included (a path starting with the seed value of  $2^n + 1$ , and when illustrating all possible ones phobic binaries). [14](#), [56](#), [57](#), [77](#)

**cluster** A cluster refers only to those vertices that share a common parent vertex. A cluster may be all or part of a level (vertex generation from a tree's root). [70](#)

**constant sequence** As defined here, constant sequence is a sequence of two or more elements all of which have the same constant. [6](#)

**dissonance start** A collection of tier paths are arranged as an array where the paths are rows and the columns refer to a move at given location for each path. The array is subdivided into three regions. Initially the array is partitioned into a coherent region and a dissonant region. The coherent region is further subdivided by a ruler harmonic region. The coherent region may or may not exist but the ruler harmonic region always exists. The dissonance start is the first column in the dissonance region that immediately follows the ruler harmonic region. [47](#)

**enumerative grove** An enumerative grove is a disjoint graph of trees. Each vertex in this graph has a node associated with it. Such a node contains a value which is the length of a parity slide. A branch from a root node to end node lists one permutation of parity slides forming a ones phobic binary from a [mosaic](#). [70](#)

**flute** Columns of a cascade are delineated into segments. Every seed is contained in a flute of length  $2^0$ . Columns of following moves are delineated by flutes of length  $2^n$ , where  $n$  is the move number. Refer to figure [8](#). [38](#)

**generation** A generation, as in  $n$ th generation, refers to all of the  $n$ th parity slides of a cascade. [52](#)

**index** Within context of this discussion, an index is an arithmetic progression of positive integers greater than zero. The common difference is greater than or equal to one. An index is an input to the  $mp$  function. Having the index as an input, this function returns the number of consecutive trailing zeros. The plural, indices refer to elements of an arithmetic progression. While the singular, index value, can refer to one element. Index can also refer to the entire arithmetic progression. The arithmetic progression could be finite or infinite. [94](#)

**landing** A landing's range is defined by a sub-sequence of positive consecutive integers whose range is a power of two. This range is called indices (in general). The range is between seed values (when referring to a cascade). Otherwise the range is the least residue system for a given  $(\text{mod } n)$ , where  $n$  is the length of the range. [18](#)

**main index** An arithmetic progression whose origin and common difference is 1. When the main index is used as input to the function  $mp$  the [standard sequence](#),  $S$ , is generated. [6](#)

**mosaic** A subset of the ones phobic binary manifestation of a cascade. A mosaic consists of all unique entries of ones phobic binaries from a cascade. [67](#), [70](#), [103](#)

**n brick** A parity slide of length  $n$  is called an  $n$  brick. An  $n$  brick is color coded by length to distinguish it from bricks of other lengths.. [35](#)

**offset** There are two types of offsets: *cascade offsets* and *tier offsets*. In one manifestation of a cascade, a move depicted as an offset. A *cascade offset* is distinguished by it's phase shift within a flute. The running column-wise total of phase congruent offsets having ranging successively from the first flute in a column to a given last flute will yield the height, at the move the last offset resides. Given a collection of tier paths. A *tier offset* is the common difference for tier path heights of a given column that represents a given move for each of those tier paths. [1](#), [14](#), [47](#), [54](#)

**ones phobic binary** A ones-phobic binary is a subset of binary numerals restricted to a length of  $n$  bits, with the condition that no bit set to 1 is adjacent to another bit set to 1. The ones phobic binary is a representation for the rise or fall movements of a "hailstone" path of length  $n$ . In a ones phobic binary a  $\bullet$  represents a rise and  $\circ$  represents a fall. [1](#), [21](#), [27](#), [70](#)

**parity slide** A parity slide is a substring of a binary number. The substring consists of zero, or more, 0's followed by a 1. Any 1 leading these 0's does not count as part of this parity slide. Should the parity slide be at the beginning of a ones phobic binary, the leading boundary of this parity slide is the leading boundary of the ones phobic string. A parity slide need not have a trailing 1 if the end of the parity slide is the end of the ones phobic binary. The parity slide could consist of the entire ones phobic binary. Should a binary number contain more than one parity slide, the parity slide is referred to by its position with respect to other parity slides, as in the  $n$ th generation parity slide. In ones phobic binaries, the first parity slide originates with the first move after the seed value. The second parity slide, if it exists, immediately follows the first, etc. [35](#), [52](#), [54–56](#)

**phase shift** Within a flute, or landing, phase shift is the indexed  $n$ th position from the beginning of the flute. (From the top of the flute, when the flutes are aligned vertically). In a flute it refers to a move (be it height, offset, m parity, or binary encoded rise/fall). In a landing, it refers to an element of an ordered least residue system  $(1, 2, 3, \dots, 0)$ . Here the zero remainder is positioned last.. [14](#), [38](#), [43](#), [104](#)

**phase shift congruent** The paths of a tier having the same [phase shift](#) for each move as the other paths in a tier. Only one path can traverse a flute at any given move. An entire path, or single move, can be phase shift congruent, within a containing flute, to the next phase shift congruent path, or move, that resides in the next adjacent flute, of equal length. [38](#), [39](#)


**placement** The index position of the mparity of: an index, seed, or sub-range. The sub-range is called a landing. [32](#)

**quilt rectangle** A quilt rectangle is the result of a multistage sort on a cascade. The sort comparison can be made with respect to either: an m parity, an offset, or the bit of a ones phobic binary. The quilt rectangle consists of equal length parity slides of the same generation. After a multistage sort, only contiguous rows of parity slides can form a quilt rectangle.. [53](#)

**ruler harmonic** A ruler harmonic sequence applies to a sequence of integers whose values, plotted as a bar chart, can be represented by the lengths of ruler scale marks where the length of mark is proportional to the distance between marks of the same length. A successive distance increase between marks of the same length is a change by a power of two. The scale mark

lengths correspond to the number of trailing zeroes, called the m parity value, of index values. These being in an arithmetic progression with a common difference of 1. The index origin is arbitrary but must be greater than 1. The index sequence length must be at least two. 5, 6, 32

**ruler subharmonic** Ruler Subharmonic, or simply subharmonic, or a ruler subharmonic sequence, is the general term for a ruler harmonic sequence of unspecified order. 5, 7, 32

**saturation** Given two paths of equal length, (where length refers to the number of moves in a path). The path with a greater degree of saturation is determined by the path with the greater number of rises, not counting the last move. A ones phobic binary has a greater degree of saturation if it has a greater ratio of ●s with respect to ○s, provided the conditions for a ones phobic binary hold. The ●○ notation is introduced to disambiguate a ones phobic binary from an ordinary binary. This concept is more readily visualized in tethered ones phobic binaries. Here an ordered pair of ●○ is represented by  which is called a bean. A tethered ones phobic consisting of nothing but consecutive beans with no interleaved un-tethered ○s, except for possibly one un-tethered ○ at one and –only one– end of the tethered ones phobic binary, is said to be fully saturated. At the other extreme, an unsaturated ones phobic binary exclusively consists of un-tethered zeroes. 66–68

**stack** A stack is a contiguous top down arrangement of quilt rectangles. A stack begins with a quilt rectangle one or two moves wide. Should any quilt rectangles follow directly underneath they would be stacked as follows: Successive quilt rectangle(s) located directly underneath each other are one move wider and half as long. 53, 62

**standard sequence** Define

$$S = \{mp(n)\}_{n \in \mathbb{N}_1}$$

to be the standard sequence. 6, 43, 103

**tier** A tier is a collection of paths grouped together to form an array. The elements of paths at a given move form a column in that array. There exists a common difference between heights at a given column (including the seed value column). All such columns constitute the coherence region of a tier and the ruler harmonic region of a tier. For the paths in these region(s), the ones phobic binary manifestations of the paths are identical. This implies equal path lengths. In some instances, only the ruler harmonic region of a tier exists as the entire coherence region. 38, 48, 62

**traversal** A table of landings consisting of placements is formed. Each row is the result of modulo arithmetic acting on  $\mathbb{N}_1$ . The moduli for successive rows ranges from  $2^1, \dots, 2^n$ . A segment of the index ( $\mathbb{N}_1$ ) is delineated by the pivot row, where the entire segment is delineated by a landing. Icicles are defined as the part of a column, holding zero placements; these extend downwards but stop before the pivot row. The traversal is what we call the remaining column, of zero place values, that reaches, or optionally passes, the landing of the pivot row. Note: landings are delineated in a way that the last place value of a landing is zero. 18, 36, 91

## Acronyms

**rhs** A ruler harmonic sequence of order  $n$ . [5](#)

## Bibliography

- [And71] George E. Andrews. *Number Theory*. Dover Publications, Inc., 1971, p. 7. ISBN: ISBN 0-486-68252-8.
- [Hay84] Brian Hayes. “Computer Recreations. On the ups and downs of hailstone numbers”. In: *Scientific American* 210 (Jan. 1984), pp. 10–16. ISSN: 0036-8773.
- [Wel97] David Wells. *The Penguin Dictionary of Curious and Interesting Numbers*. Penguin Books Ltd., 1997, pp. 55–56. ISBN: 978-0-14-192940-8.
- [Sim07] Stuart A. Kurtz; Janos Simon. In: TAMC 2007, held in Shanghai, China. Ed. by J. Y. Cai; S. B. Cooper; H. Zhu. May 2007, pp. 542–553. DOI: [10.1007/978-3-540-72504-6\\_49](https://doi.org/10.1007/978-3-540-72504-6_49).
- [Del21] Jean-Paul Delahaye. “Some Puzzles from the Game of Life Creator John Conway”. In: *Scientific American* (Apr. 2021). ISSN: 0036-8733.