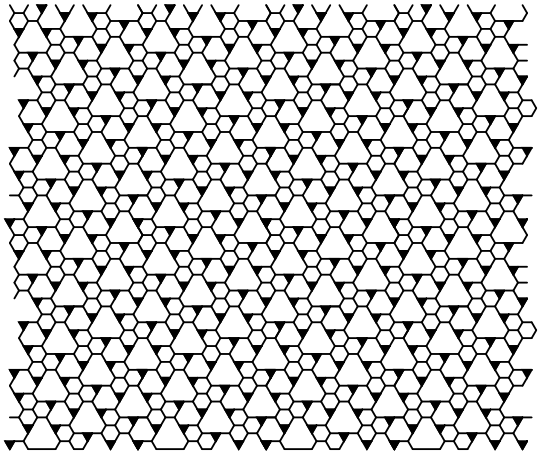


Properties of Bitloops

Malcolm Black

2020

1. Bits

A bit is a number that is a zero or one. It is the most fundamental material presented here.

$$Bit \in \{0, 1\}$$

One signifies value of itself, as well as a difference between numbers. Zero signifies no value of itself, and no difference between numbers.

0 1 0 1 1 1 0 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0

See the individual bits, each a one or zero. See them as a group, and there is continuity and change.

0 0 1 1 0 1 1 0 1 0 1 0 1 0 0 1 0 1 0 0 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1

On the left is mostly change. On the right is mostly continuity.

0 1 1 0 0 0 1 1

The difference between one and zero is one, hence one signifies change. The difference between two ones or two zeros is zero, hence zero signifies continuity.

2. Bytes

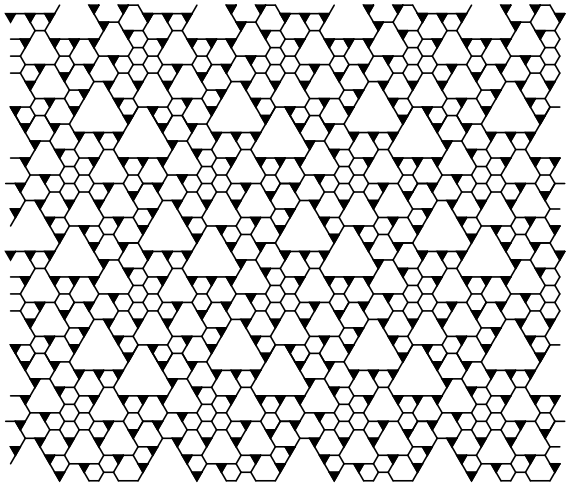
A byte is a group of bits in a row, and has two ends. It may have as few as one bit. Each end is a bit that is adjacent to only one other bit, while the other bits are adjacent to two.

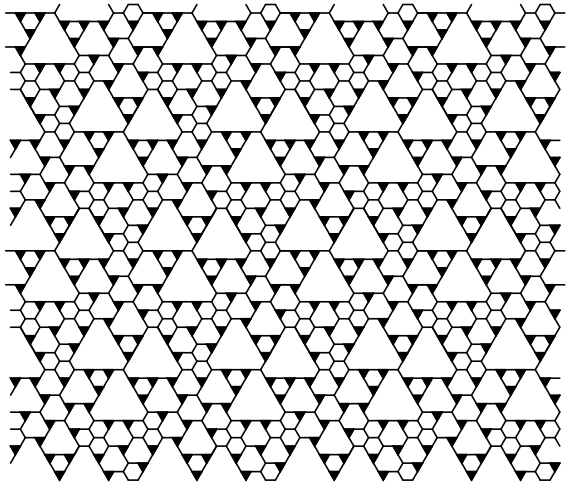
[0 0 0 0 1 1 1 0 1 0 1] [0 1 0 1 0] [1 0]

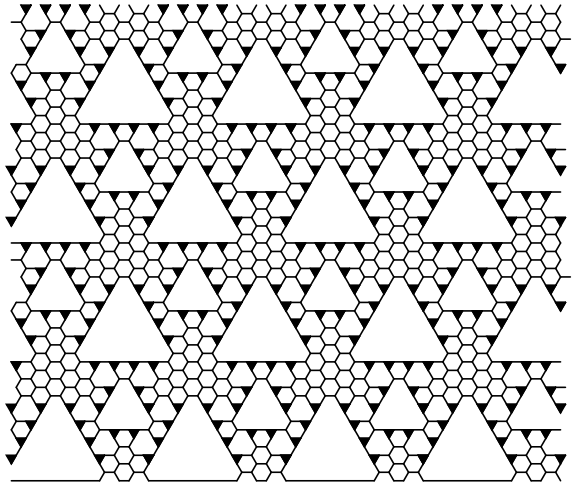
There is not one direction from one end to the other, so there is not a first bit and a last bit. Whether the two ends are distinct depends on the bits.

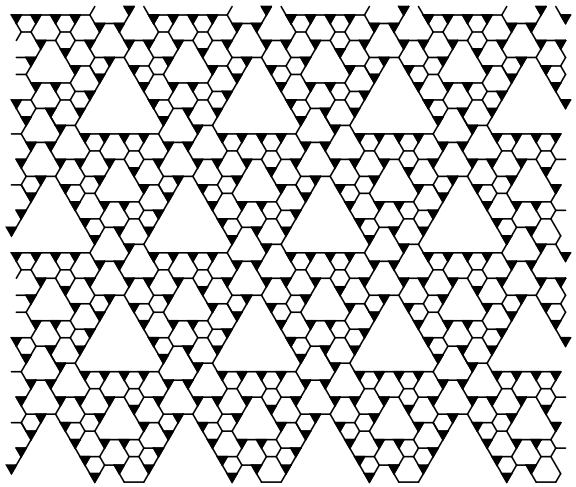
[0 0 0 1 0 1 1 1 0 1 0 0 0] [0 1 0 1 0] [1 1 1 1 0 0 0 0]

Square brackets may indicate bytes if needed, for example, to distinguish from a set. Commas are not used here.



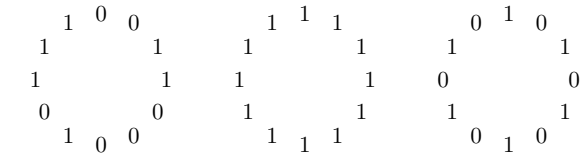






3. Bitloops

A bitloop is a byte without ends.



The entirety of this work is directly consequential to this concept. All the things that follow are original observations about what bitloops are and how they interact.

For the sake of communication, bitloops are often displayed as a straight row of bits, with the implication that the bits on the ends are connected. Consequently, there may be multiple ways to display the same bitloop. For example:

[0 1 0 0 0 1 0] [0 0 1 0 0 1 0] [1 0 0 0 1 0 0] [1 0 0 1 0 0 0]

Each of these signify a distinct byte, and all of them signify the same bitloop. Three more bytes that have not been displayed also signify the same bitloop.

Another useful interpretation of the bitloop is a byte that repeats indefinitely from both ends. The repetitions are like reflections in a mirror, which are only apparent. Notice how in this manner, the end bits in a byte cannot be determined. All that can be determined is the number of consecutive bits in a byte that is not itself repetitive.

. . . 0 1 1 1 0 0 1 1 1 0 0 1 1 1 0 0 1 1 1 0 0 1 1 1 0 0 1 1 1 0 0 . . .

[0 0 1 1 1] [0 1 1 1 0] [1 1 1 0 0] [1 1 0 0 1] [1 0 0 1 1]

All five of these bytes generate the same sequence when repeated end to end.

4. Bitloop Classes

The set of all bytes that may be derived from the same bitloop (as above) is here referred to as a bitloop class. For example, here are eleven bytes derived from a bitloop with eleven bits.

```
      1   0
  0           1
0               0
1               0
  1           0
    0   0
```

```
10100001100  01100101000
01000011001  11001010000
10000110010  10010100001
00001100101  00101000011
00011001010  01010000110
00110010100
```

Bitloop classes are sets, and unlike the bits of a byte or bitloop, all the elements of a set must be distinct. The bitloop below has twelve bits, but only three distinct bytes in its bitloop class.

```
    1   0   1
1           1
0           0
1           1
    1   0   1
```

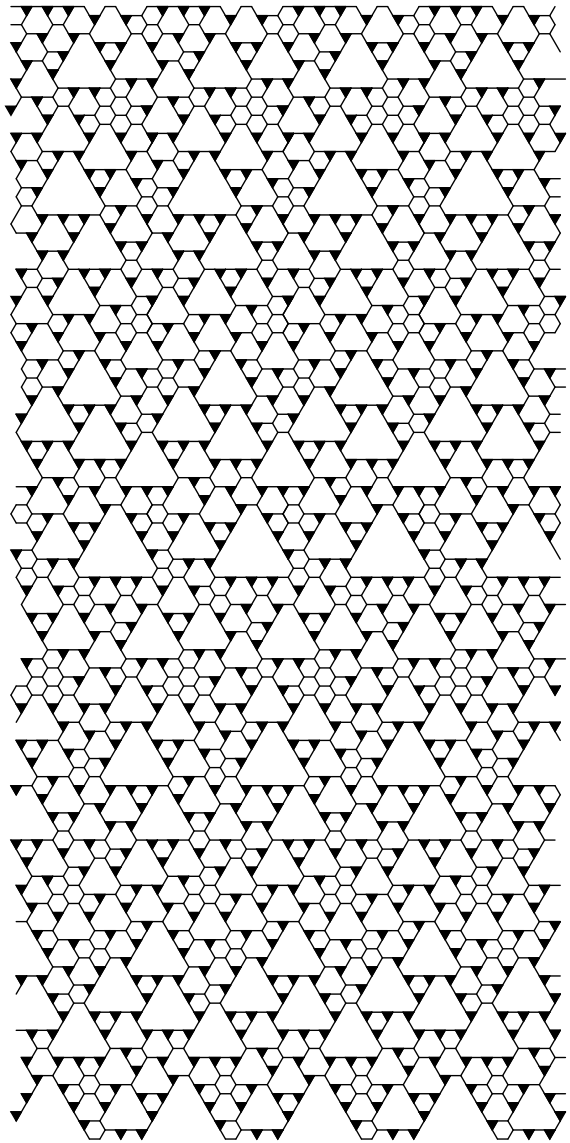
```
110110110110
101101101101
011011011011
```

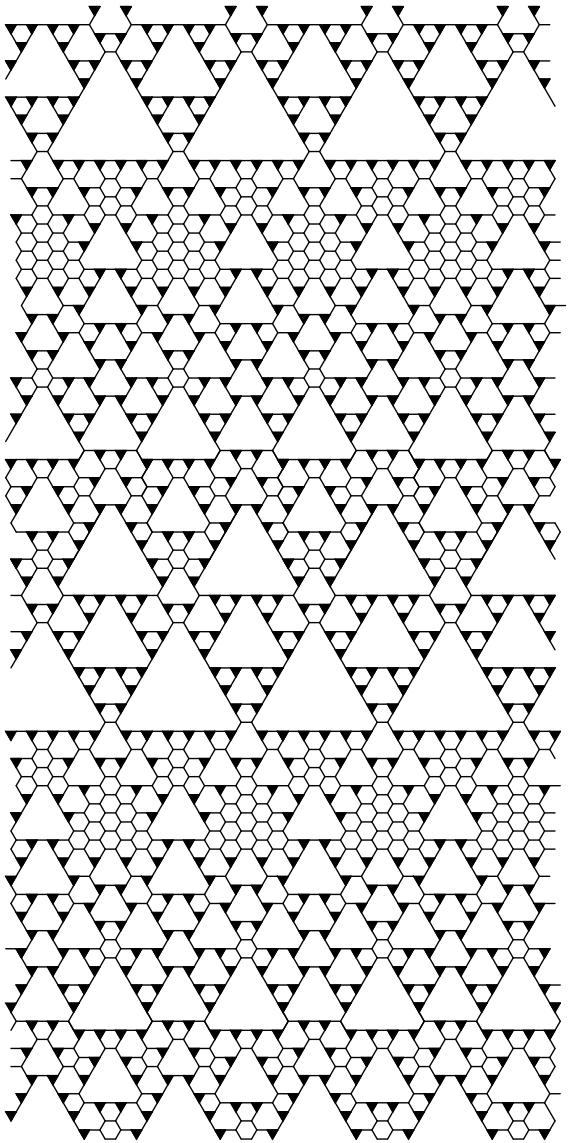
5. n

The letter n refers to the number of bits in a byte or bitloop. For any bitloop with n bits, the number of bytes in its bitloop class is also n, unless it consists of a smaller repeating byte, later referred to as byte B.

6. A Bitloop Class in Music

The seven modes of the diatonic scale may be interpreted as seven bytes in a bitloop class. In each mode, seven notes are played in ascending or descending order, and span a twelve-note octave. Notes are skipped according to the same recursive squence of intervals: skip one, skip one, skip none, skip one, skip one, skip one, skip none.





Allow one to signify skip one, and zero to signify skip none, and the modes may be represented by the bitloop class below. Notice the bits do not signify notes, but intervals between notes.

	1	1	1101110	1110110	1011011
0		0	1011101	1101101	0110111
1	1	1	0111011		

7. L

This letter refers to a kind of set. It is the set of all bytes with n bits, where n is the same value for every byte. Thus for every positive interger there is a corresponding L. The cardinality of L, or the number of bytes it consists of, is always a power of two.

L_n refers to the set for a specified n, and n may be replaced by a number. For example:

$$L_1 = \{[0], [1]\}$$
$$L_2 = \{[00], [11], [01], [10]\}$$

8. Inverses

In our present context, the inverse function regards a byte or bitloop. The function transforms a byte or bitloop so all its zeros become ones, and all its ones become zeros. The function is bijective and involutary. In other words, every byte or bitloop has one inverse, and is the inverse of its inverse. Two bytes or bitloops that are the inverses of each other must have the same number of bits. Here are some inverse pairs:

[1110010]	[0]	[0011]	[000000101000]
[0001101]	[1]	[1100]	[111111010111]

If the bytes of an inverse pair belong to different bitloop classes, the two bitloop classes consist entirely of such inverse pairs. Thus bitloops also form inverse pairs, because the bytes in their bitloop classes all form inverse pairs with one byte from each bitloop.

Here are two bitloop classes, left and right, with ten inverse pairs between the two.

[0000000100]	[1111111011]
[0000001000]	[1111110111]
[0000010000]	[1111101111]
[0000100000]	[1111011111]
[0001000000]	[1110111111]
[0010000000]	[1101111111]
[0100000000]	[1011111111]
[1000000000]	[0111111111]
[0000000001]	[1111111110]
[0000000010]	[1111111101]

9. Half Inverses

If both bytes of an inverse pair belong to the same bitloop class, the corresponding bitloop does not have an inverse. These bitloops and the bytes in their bitloop classes are referred to as half inverse.

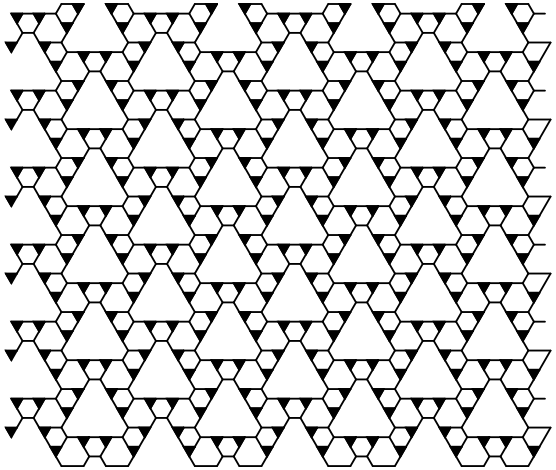
If a half inverse byte or bitloop is split in half, the two halves are an inverse pair. A bitloop class with one half inverse byte consists entirely of half inverse bytes. For example, here are the eight bytes of a single bitloop class, again with inverse pairs to the left and right of each other. The corresponding bitloop does not have an inverse.

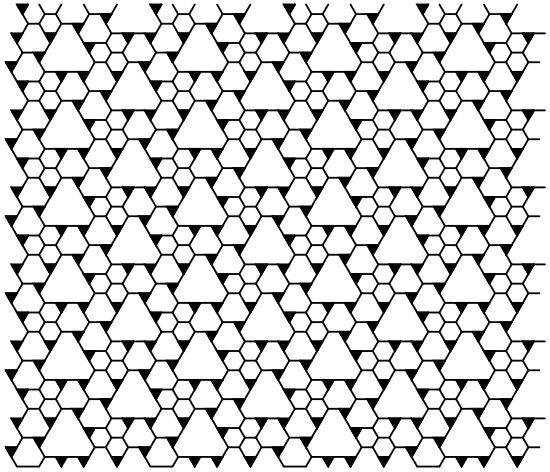
[11110000]	[00001111]
[11100001]	[00011110]
[11000011]	[00111100]
[10000111]	[01111000]

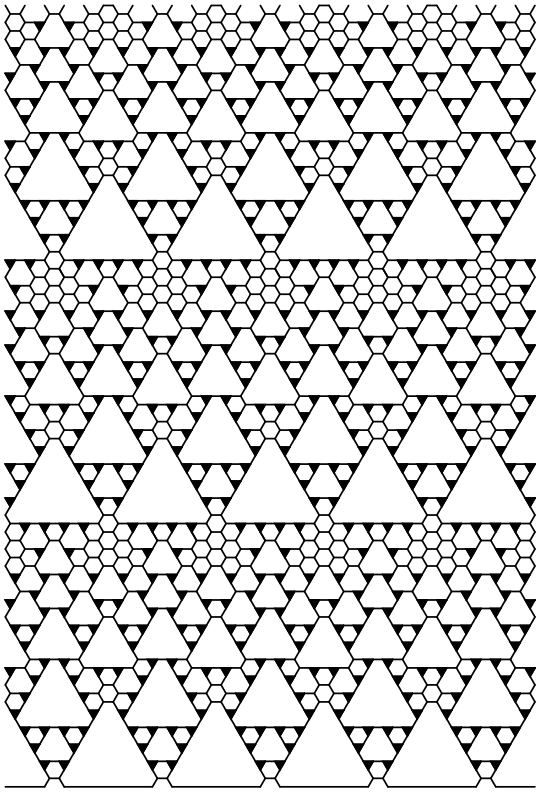
10. Partition by Inverses

Every byte has one inverse, so it may be said that L consists entirely of inverse pairs. This set of pairs is also referred to as a partition of L. Another partition of L with respect to inverses is two subsets that separate every byte from its inverse. For example, L_3 consists of eight bytes:

[000] [111] [001] [010] [100] [011] [101] [110]







Here the inverse pairs are arranged together:

[000] [111] [001] [110] [100] [011] [101] [010]

Here they are kept separate:

[000] [001] [010] [100] [111] [110] [101] [110]

11. Partition by Bitloop Classes

L can be partitioned into bitloop classes, such that every byte in L belongs to one bitloop class. Although bitloops are not elements of L, bitloops can be used to represent its bytes.

Take for example L_5, the set of all bytes with five bits. It consists of 32 bytes, and is partitioned into eight bitloop classes. With the understanding that the cardinality of L_5 is 32, the reader can confirm the bitloops shown represent the entire set.

[00000]	[00101]	[11010]
[11111]	[00011]	[11100]
	[11110]	[00001]

Each bitloop on the left represents a bitloop class consisting of one byte, and the display of that byte is identical. The other bitloops all represent bitloop classes consisting of five bytes each. $2 \times 1 + 6 \times 5 = 32$

In the effort to display only what is necessary to determine a complete set, one bitloop from each inverse pair may also be discarded from the display. Here is L_5 again, with half the bitloops:

[00000] [00101] [00011] [11110]

Here's L_6, represented by bitloops, including only one bitloop from each pair of inverse bitloops. Two of them are half inverse, so the total amount of bitloops signified is $2 + 2 \times 6 = 14$.

[000000]	[010101]	[000111]	[011011]
[000001]	[000011]	[000101]	[001011]

12. B

...0111001110011100111001110011100...

Recall the indefinitely repeating byte that resembles the bitloop. Compare it to a byte that repeats, say, forty times. Consider the result as a byte in its own regard, for it has definite ends. There are other bytes that seem to repeat in it, but their iterations on the ends are not complete. B refers to the byte that repeats in complete iterations. B itself must also consist of a B that occurs once. Every byte, no matter what it is, has its own B, and only one.

[001001001001001001001001]

Displayed is a byte with 24 bits. The B that corresponds is [001]. In a byte with all zeros or all ones, B consists of one bit and occurs n times. In bytes that are not repetitive, B occurs once. The cardinality of any B must be a factor of n.

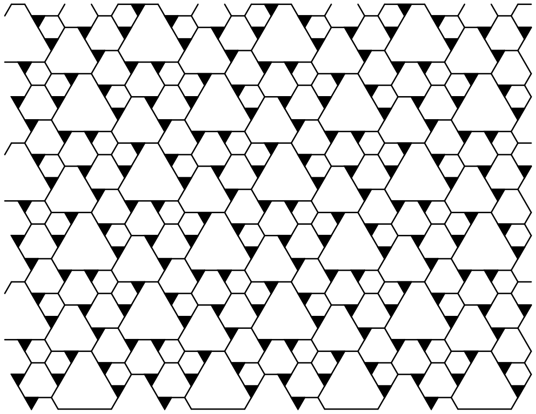
BYTE	[0000]	[010101]	[01100100110011]
B	[0]	[01]	[01100100110011]

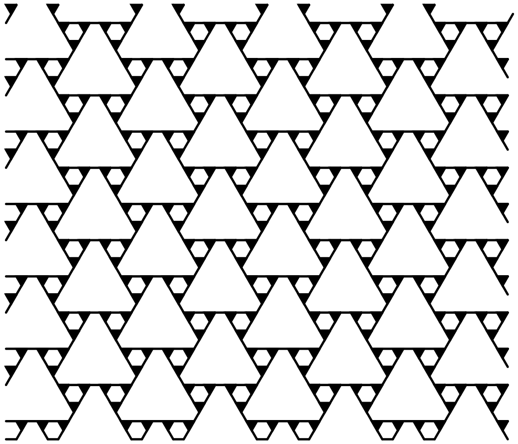
13. Partition by B

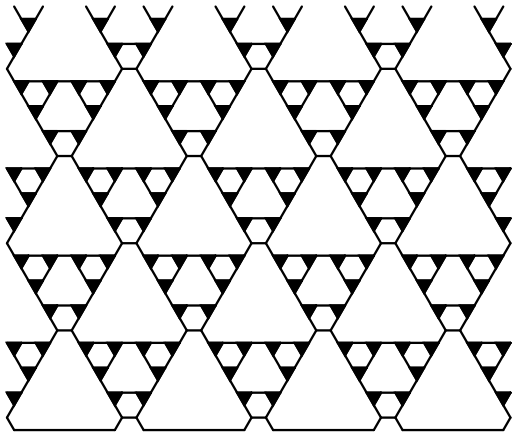
No more than one B may correspond to a given byte, so L_n may be partitioned according to the factors of n. In such a partition there is a subset for every factor of n, and every byte in L belongs to one subset.

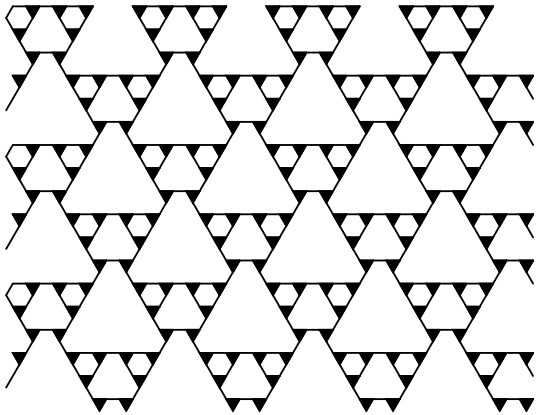
Of the sixteen bytes in L_4, two bytes have a B with one bit, two bytes have a B with two bits, and twelve bytes have a B with four bits. Below the bytes of L_4 are represented by bitloops, and inverses are not included.

BITS IN B	1	2	4
BITLOOPS	[0000]	[0101]	[1100]
			[0111]









Of the 64 bytes in L_6, two bytes have a B with one bit, two bytes have a B with two bits, six bytes have a B with three bits, and 54 bytes have a B with six bits.

BITS IN B	1	2	3	6
BITLOOPS	[000000]	[010101]	[011011]	[000111]
				[000001]
				[000011]
				[000101]
				[001011]

14. *la*, *lb* and *lc*

Another way to partition L_n according to the factors of n is to use the same subset for every factor that is not 1 or n. This leaves three subsets, referred to here as *la*, *lb* and *lc*.

SUBSET	*la*	*lb*	*lc*
FACTOR OF n	1	*other*	n

For any n, *la* always consists of two bytes, and when n is a prime number, *lb* is empty.

15. Proof About Prime Numbers

$$\{n \in \mathbb{P} \mid (2^n - 2)/n\} \subset \mathbb{N}_1$$

la, *lb* and *lc* happen to identify a property of prime numbers: For any prime number n, $2^n - 2$ is divisible by n. The reason is that $2^n - 2$ is the number of bytes in L_n minus those in *la*. This leaves *lb* and *lc*, and there are not any bytes in *lb*, because there are no factors of n besides 1 and n. Finally the number of bytes in *lc* is always divisible by n. The reason has to do with bitloop classes. A byte whose B consists of n bits belongs to a bitloop class having n bytes. So for any n, *lc* can be partitioned into bitloop classes such that every bitloop class consists of n bytes. So the number of bytes in *lc* is divisible by n, and $2^n - 2$ is divisible by n when n is prime.

16. Conjecture About Prime Numbers

$$\{n \in \mathbb{N}_1 \Delta \mathbb{P} \mid (2^n - 2)/n\} \subset \mathbb{R}\Delta\mathbb{N}_1$$

Maybe it is true that if n is not prime, then $2^n - 2$ is not divisible by n. It is at least calculating n for values up to fifty. Other than that, the problem reduces slightly. $2^n - 2$ is the sum of the cardinalities of lb and lc, and it is given from the last proof that lc is divisible by n. What remains to be seen is if lb is divisible by n. Also, the value of n may be generalized to include prime numbers, since when n is prime, lb is empty, and zero is not divisible by n.

$$\{n \in \mathbb{N}_1 \mid |lb|/n\} \subset \mathbb{R}\Delta\mathbb{N}_1$$

17. Links

ABSOLUTE DIFFERENCE	0	0	1	1
ADJACENT BITS	0 0	1 1	0 1	1 0

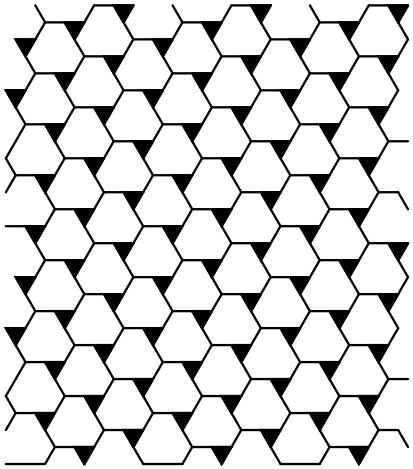
For any given bitloop, the difference between any pair of adjacent bits is either one or zero. Because there is no direction from one end of the bitloop to the other, the difference is considered absolute, or positive.

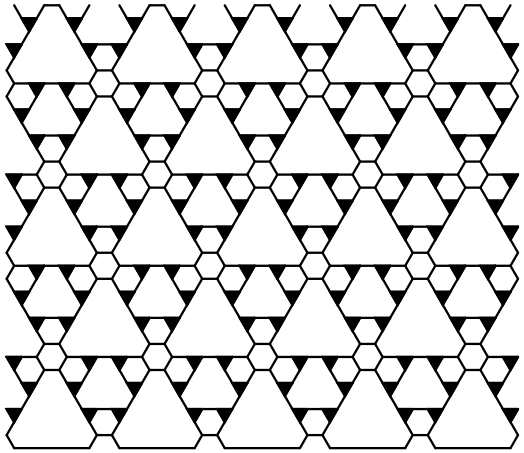
0 1 0 1 1 1 0 0 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
0 0 1 1 0 1 0 0 0 1 0 1 0 1 0 1 1 1 1 1 0 1 0 1 1 1 1 1

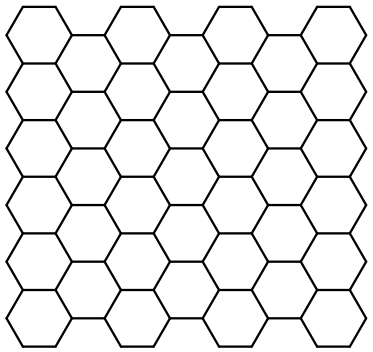
A bitloop with n bits has n differences. These differences have an order just like the the bitloop they are derived from, so they too may be interpreted as the bits of a bitloop.

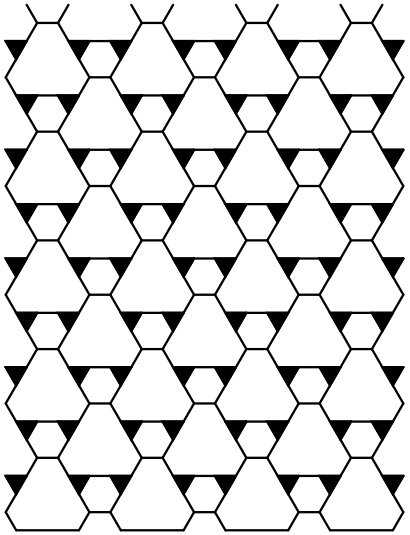
LINK	0 1 0 1 1 1 1 0 0 1	1 1 1 1 0 0 0 0 0
SUBLINK	0 0 1 1 0 1 0 1 1 1	0 1 0 1 0 0 0 0 0

A bitloop that is derived from another bitloop in this way is referred to as a link. The bitoop it is derived from is referred to as a sublink.









Properties of links:

1. For every bitloop there is a corresponding link.
2. Two bitloops with the same link are an inverse pair.
3. A link has an even number of ones.
4. A half inverse bitloop has a link whose corresponding B occurs twice.

LINK	0 0 0 1 0 0 0 1	0 0 1 1 0 0 1 0 0 1 1 0 0 1
HALF INVERSE	0 0 0 0 1 1 1 1	1 1 1 0 1 1 1 0 0 0 1 0 0 0

In rare instances bitloops may be their own links, such as these:

[0] [011] [1110010]

18. Link Notation

The display of links requires particular attention in order to discuss their actual propeties. A unique challenge in the study of bitloops is to distinguish to what extent something is a property of bitloops or is a method to display them.

		0 0 0 0 0 0 0 0	0 1 1 0 1 1 0 1 1
	↑	1 1 1 1 1 1 1 1	1 0 1 1 0 1 1 0 1
LINK ITERATIONS		1 0 1 0 1 0 1 0	0 1 1 0 1 1 0 1 1
	↑	0 0 1 1 0 0 1 1	1 0 1 1 0 1 1 0 1
		0 0 0 1 0 0 0 1	0 1 1 0 1 1 0 1 1

Because bitloops are displayed as a row with two apparent ends, the bits on the ends don't have a convenient place above and between them to display their absolute difference. So a convention is to display the difference on the right or left side of the link, as shown above. Links of links may be stacked upon each other, and in such case left and right alternate each row up.

. . . 0 1 1 1 0 0 1 1 1 0 0 1 1 1 0 0 1 1 1 0 0 1 1 1 0 0 1 1 1 0 0 . . .

Recall again our friend, the indefinitely repeating byte. It has the advantage of displaying every one of its bits between two neighbors. One may wish to display a bitloop a couple extra times, just

to be able to see it without those ends, which are not true properties of the bitloop anyways. Then again, what display would be a true property, and what is a property without its display?

```
 1 0 0 0 1 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1
1 0 0 0 0 1 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0
 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0 0 0 0 1 1
0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1
```

19. Link Orbits

A series of successive links eventually begins to repeat itself, entering into a cycle, or link orbit. The seasons of the year are comparable to four links in a link orbit. Also, every time a season comes around again, it's a little different than the year before. Links usually come back with a rotated order. Bitloops with an even number of bits usually rotate halfway around.

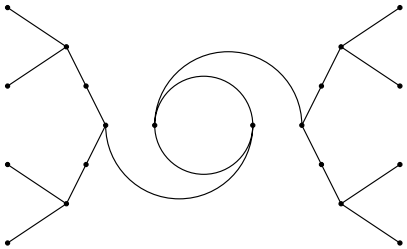
LINK ORBIT: [1110010] → *loop*

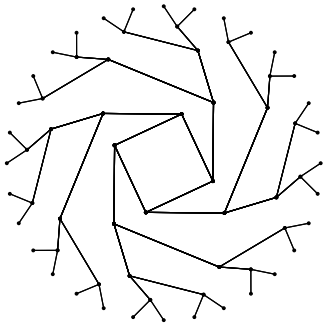
```
1 0 1 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1 1 0 0
 0 0 1 0 1 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1 1
1 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1 1 0 0 1 0
```

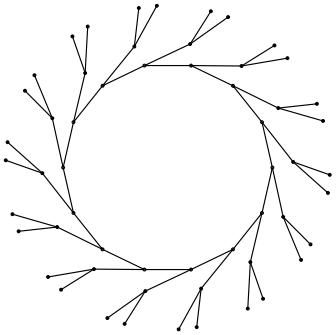
Above, the bitloop [1110010] repeats four times per row. It is its own link, so each iteration is a complete orbit. For every iteration, the bits shift three to the left (or equivalently, four to the right). One could say, for each row the bitloop is represented by a diferent byte in its bitloop class, repeating four times.

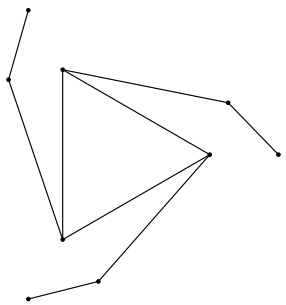
LINK ORBIT: [00011] → [00101] → [10111] → *loop*

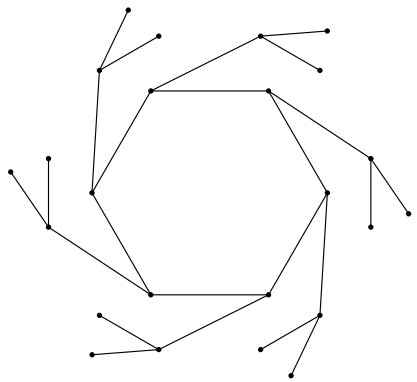
```
 1 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1 1 1 0 1
1 0 1 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 0
 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0
1 0 1 1 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1 1
 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1
0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1
```

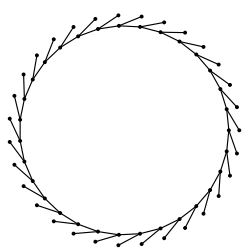







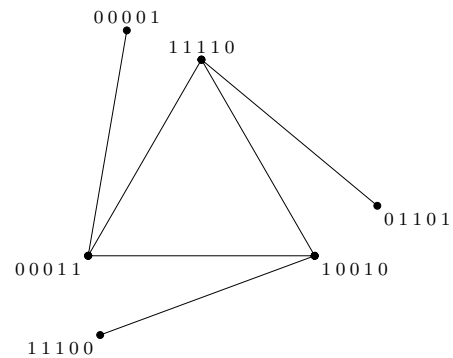






In a link orbit, every bitloop has two neighbors. The bitloop must be the link of one and the sublink of the other; it cannot be the link of both or the sublink of both. This rule is enough to require a direction of transformation (clockwise or counter-clockwise) that is consistent among all adjacent bitloops.

20. Link Orbit Inverses and Sublink Trees

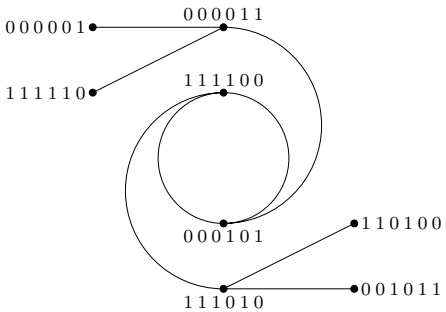


Every link in a link orbit has two sublinks; one of them is in the link orbit, and one is not. The sublink that is not in a link orbit is referred to as a link orbit inverse.

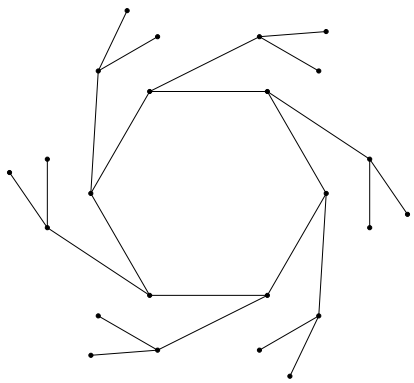
A link orbit inverse may have its own sublink or pair of sublinks, and depending on the bitloops, a succession of sublinks may continue to whatever end. This tree that begins with a link orbit inverse is referred to as a sublink tree. Every sublink tree that corresponds to the same link orbit consists of the same number of bitloops. The set of bitloops in a link orbit and every sublink tree that corresponds is referred to as a bitloop chain.

21. Bitloop Chains

Without reference to link orbits or sublink trees, a bitloop chain may be defined as a set of bitloops, such that every bitloop that is a link or sublink of an included bitloop is also included, and no bitloop that is not a link or sublink of an included bitloop is included.

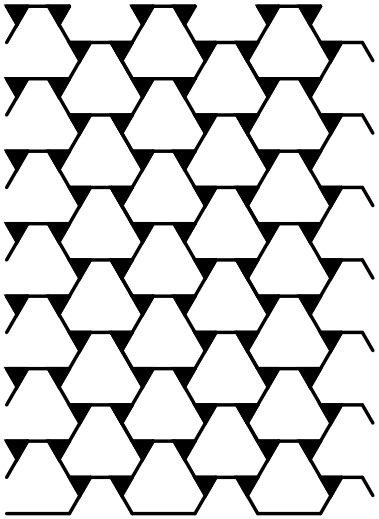


Here is a bitloop chain with six bits per bitloop. It has two bitloops (or links) in its link orbit, and three bitloops (or sublinks) in each of its sublink trees. Each link orbit inverse has two sublinks.



Four different bitloop chains correspond to the graph above. Each chain has a link orbit with six links, and six sublink trees that each consist of three sublinks. The four bitloop chains all have bitloops with ten bits. Each chain is a host to 24 bitloops, each with ten bytes in its bitloop class, so the four chains together account for 960 distinct bytes. The total number of bytes with ten bits (or the cardinality of L_{10}) is 1024, so there are only 64 more bitloops with ten bits, and they belong to other bitloop chains.

Below is the same link orbit again, repeated the same number of times. The difference is how it is displayed. Instead of ones and zeros, lines connect where ones normally would be, and black triangles fill in where there would be two adjacent ones with a one below. All the data is still there, but the interpretation emphasizes some properties that may otherwise be overlooked.



Any tessellation displayed in this fashion consists entirely of these white hexagons and black triangles. The black triangles are always the same size. There are hexagons of different sizes, but only the bottom, upper right, and upper left sides vary in length. It seems to me that a tessellation cannot be constructed with only these materials if it does not correspond to a bitloop chain.

The pages that follow contain a selection of graphs and tessellations that correspond to bitloop chains.

Visit *www.symbolfigures.org* for a more complete catalogue of chains and their corresponding graphs and tessellations.

Recall that the bitloop class of a symmetric bitloop may contain an asymmetric byte as long as the reverse of that byte is also included. This is the case with symmetric chains. A chain is symmetric if either all of its bitloops are symmetric, or for any included asymmetric bitloop, its reverse is also included.

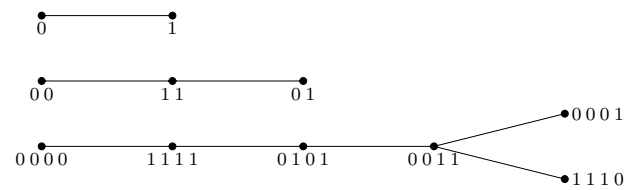
26. Tessellations

Here is a link orbit consisting of two links, displayed once at the top, and then repeated three times per row and fourteen times per orbit.

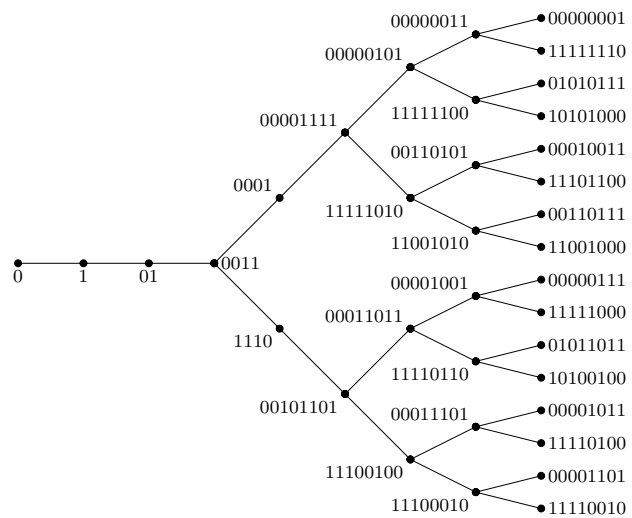
```
000101
111100

101000101000101000
100111100111100111
000101000101000101
111100111100111100
101000101000101000
100111100111100111
000101000101000101
111100111100111100
101000101000101000
100111100111100111
000101000101000101
111100111100111100
101000101000101000
100111100111100111
000101000101000101
111100111100111100
101000101000101000
100111100111100111
000101000101000101
111100111100111100
101000101000101000
100111100111100111
000101000101000101
111100111100111100
101000101000101000
100111100111100111
000101000101000101
111100111100111100
```


Some bitloop chains have a link orbit with few links and sublink trees with many sublinks; some bitloop chains have a link orbit with many links and sublink trees with few sublinks. For any bitloop chain whose bitloops have n bits, such that n is a prime number, its sublink trees consist only of a link orbit inverse with no sublinks.

22. Power of Two Chains



For any L_n such that n is a power of two, all bitloops that partition L_n belong to the same bitloop chain. For any other n, L_n consists of multiple bitloop chains. Above are the chains for L_1, L_2, and L_4. Below is the chain for L_8, with B standing in for each bitloop. For example, the bitloop [00000000] is shown as [0].



23. Partition by Bitloop Chains

Every bitloop belongs to one bitloop chain, thus L_n can be partitioned into bitloop chains. First the bytes of L_n are partitioned into bitloop classes, and then their respective bitloops can be partitioned into bitloop chains.

L_1	L_2	L_3	L_4	L_5	L_6	L_7	L_8	L_9	L_{10}	L_{11}	L_{12}	L_{13}	L_{14}
1	1	2	1	2	3	3	1	5	5	3	7	4	17

Listed above are the number of bitloop chains that partition L, for values of n from one to fourteen.

I can actually only confirm sixteen bitloop chains that partition L_{14}. This amounts to 16,160 bytes, out of a total 16,384, or 2^{14}. I cannot find any of the remaining 224. However, one of the sixteen known chains has 224 bytes, so my guess is the missing bitloops comprise the same kind of chain.

24. Byte Symmetry

Another function of a byte is to reverse the order of its bits from one end to the other. Like the inverse function, the reverse function is bijective and involutary. However, not every byte has a distinct reverse. An asymmetric byte has a reverse, while a symmetric byte does not.

[00000110010] [01001100000] [00011000]

The bytes on the left are asymmetric, and each is the reverse of the other. The byte on the right is symmetric and does not have a distinct reverse.

Symmetric and asymmetric bytes can belong to the same bitloop class. Look for the reverse of every asymmetric byte in the bitloop class below. One byte doesn't have a pair because it is symmetric.

```
 1 1        1101110 1110110 1011011
0   0       1011101 1101101 0110111
1   1       0111011
 1 1
```


25. Bitloop Symmetry

A bitloop is referred to as asymmetric if its bitloop class consists of asymmetric bytes and the reverses of those bytes comprise a different bitloop class. Any other bitloop is symmetric. The bitloop class corresponding to a symmetric bitloop may consist of all, some or no asymmetric bytes. But for any asymmetric byte that is included, its reverse is also included.

```
 1 0 1        0 0 1        0 1 1 1
0     0      0     1      1     1
0     0      0     0      1     1
 1     1      1     0      1     0
  1 1          1 0 0        1 0 0
  SYM.         SYM.         ASYM.
```

A convenient way to determine the symmetry of a bitloop is to first display its bits in a circle. Then, if an axis of symmetry can be drawn across the bitloop, it is considered symmetric. The line may pass through or between one or two bits.

```
00100011  11000100
01000110  01100010
10001100  00110001
00011001  10011000
00110010  01001100
01100100  00100110
11001000  00010011
10010001  10001001
```

Here are two bitloop classes corresponding to two asymmetic bitloops. Reverse pairs of bytes are divided between the two classes. The two bitloops representing these classes are also referred to as a reverse pair.

A bitloop chain that includes an asymmetric bitloop may consist entirely of asymmetric bitloops. If such is the case, the reverses of all of those bitloops comprise their own reverse chain. Thus two chains may form a reverse pair, and each chain is referred to as asymmetric. The list from earlier of how many chains partition each L_n counts one chain for each reverse pair.