11-3-2019

Repeat Length Of Patterns On Weaving Products

Zhuochen Liu

Rose-Hulman Institute of Technology, liuz7@rose-hulman.edu

REPEAT LENGTH OF PATTERNS ON WEAVING PRODUCTS

Zhuochen Liu

Rose-Hulman Institute of Technology
Email: liuz7@rose-hulman.edu

ABSTRACT

Interlacing strands have been used to create artistic weaving patterns. Repeated patterns form aesthetically pleasing products. This research is a mathematical modeling of weaving products in the real world by using Cellular Automata. The research is conducted by observing the evolution of the model to better understand products in the real world. Specifically, this research focuses on the repeat length of a weaving pattern given the rule of generating it and the configuration of the starting row. Previous studies have shown the range of the repeat length in specific situations. This paper will generalize the precise repeat length in one of those situations using mathematical proofs. In the future, the goal is to further generalize the findings to more situations.

1. INTRODUCTION

People began to create aesthetic patterns of weaving products (Fig. 1) in ancient times. A pattern is a region on a weaving product that serves a decorative purpose. No matter how complicated the patterns are, they are composed of strands interlacing each other. To study the rules behind patterns, it is a good idea to model strands that generate those patterns.

Fig. 1: An example of weaving products. (Pictured by USAID Biodiversity and Forestry)

In weaving products, if we define the direction of one strand as straightly upward, the direction of all strands will be either straight or slanted. From aesthetic and practical perspectives, there are only limited number of directions of strands. Usually, there are only two or three directions of strands in a weaving product, so it is safe to simply classify a strand as straight, slanted to left, or slanted to right. With this simplified classification, the states of two strands can be straight or slanted, one crossing with the other or not, and how they cross. There can be two strands such that they cross each other but one is straight and the other is slanted, but to make the problem easy, this situation is not covered in this paper. Therefore, it is reasonable to divide the whole pattern into grids and use cellular automata (Definition 1) to represent them. Normally, the strands are either horizontal or vertical, but to better observe them, the weaving product can be rotated by 45 degrees and then divided into cells, as shown in Fig. 2.

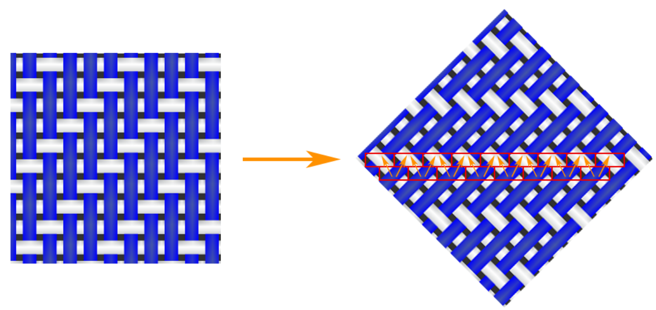

Fig. 2: Rotate the weaving product and then divide it into cells. (Picture by David C Todd, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=71461797)

Not all patterns on a weaving product are unique. In fact, a weaving product usually contains patterns repeating across the surface, as shown in Fig. 3. To better understand these patterns, we want to know when they start to repeat. However, the number of ways to combine strands is large. To narrow the problem down, this paper will only focus on slanted strands.

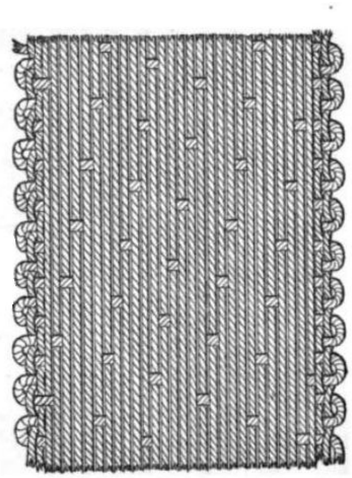

Fig. 3: Patterns on a weaving product. (Picture by Alfred Barlow)

When talking about the row where the pattern starts to repeat, the positions of cells can be either considered or not. In Fig. 4, if positions are not considered, the pattern starts to repeat at row 9 counting from bottom because its configuration is the same as row 1 and the repeat length is 8, but if positions are considered, the pattern does not repeat here because the positions do not match.

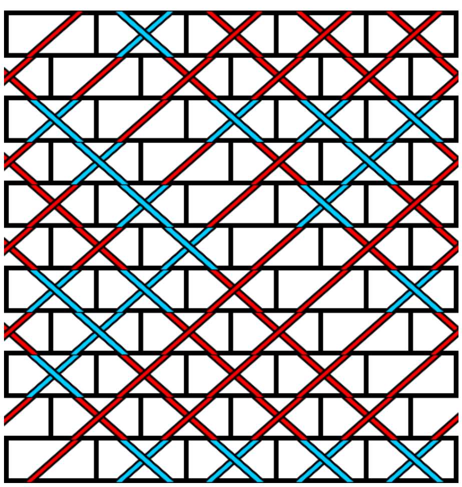

Fig. 4: Positions of cells affect if the pattern starts to repeat or not.

Previous studies about slanted strands have already achieved significant results. In all of these studies, positions of cells in the model are considered. Let m represent the width of the starting row and let $2^k < m \leq 2^{k+1}$, Dr. Joshua Holden [1] proved that if all strands are slanted, the maximum repeat length, w_m, over all crossing rules and starting rows will be $\operatorname{lcm}(2^{k+1}, m) \leq w_m \leq m2^m - 2^m$. In addition, if $m < 5$, then w_m will be exactly $\operatorname{lcm}(2^{k+1}, m)$, and if $m = 23$, then $w_m > \operatorname{lcm}(2^{k+1}, m)$. Moreover, Hao Yang [2] showed that under additive crossing rules (Definition 3), if there is no unpaired strand in the starting row and m is a power of 2, then the maximum repeat length is 2.

This research expands Dr. Holden's work to get the exact repeat length for starting rows with all slanted strands under additive crossing rules or inverse additive crossing rules (Definition 2.4). In this paper, positions of cells are not considered when talking about the repeat length. Repeat patterns are related to Pascals Triangle $\mod 2$. The next section introduces necessary definitions, and section 3 provides the repeat length and the detailed proof.

2. DEFINITIONS

2.1. Stranded Cellular Automata

Definition 1. A *cellular automaton* (pl. cellular automata) is a discrete model consisting of a regular grid of cells, each in one of a finite number of states. For each cell, a set of cells called its neighborhood is defined relative to the specified cell. In our model, each cell has two neighbors, which are the two adjacent cells that generate it, as Fig. 5 shows. The *stranded cellular automaton* (SCA) is a cellular automaton whose cells represent the state of strands. It has 8 states in total [1], as shown in the Fig. 6.

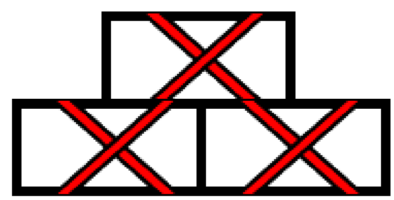

Fig. 5: The two cells at the bottom are neighbors of the cell at the top.

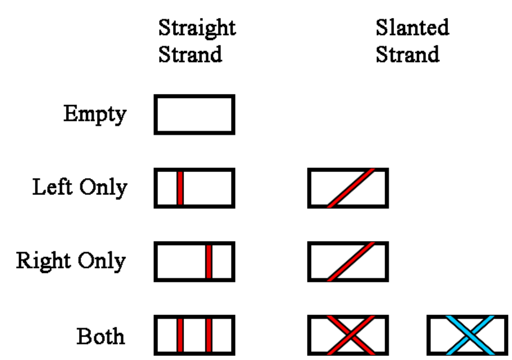

Fig. 6: Eight states of an SCA. (Picture based on J. Holden[1])

2.2. Crossing Rules

When two slanted strands cross, we need to determine which strand is on top of the other. There are three possible states: left strand on top, right strand on top, or strands do not cross. In the last state, if we do not consider the empty cell, there will be just one strand in the cell and we call it an unpaired strand.

Definition 2. In the SCA, a new cell is generated from two adjacent cells in the previous row, as shown in the right part of Fig. 2, and the *crossing rule* is used to determine the crossing state of the new cell based on states of the two adjacent cells.

Because there are 3 possible states for each cell, there are $3 \times 3 = 9$ possible combinations of adjacent cells, and we use 9 bits to represent them. 9 combinations together form a crossing rule. As for bit representations, we use "1" to represent left strand on top and "0" to represent right strand on top. The value of an unpaired strand is undetermined, and is represented as "N". Fig. 7 shows how a crossing rule looks.

Since 9 bits are used to represent one crossing rule, there are a total of $2^9 = 512$ crossing rules.

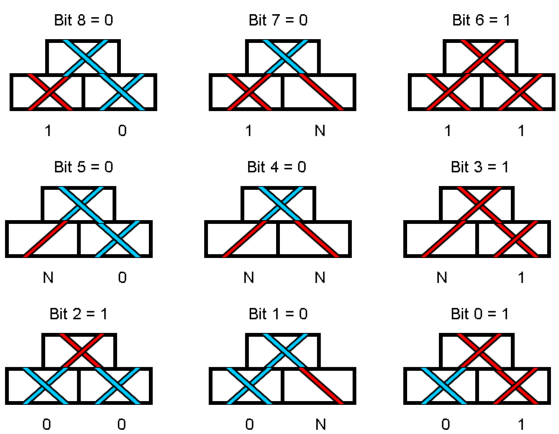

Fig. 7: Bit representations of a crossing rule. The value of an unpaired strand is undetermined and is represented by "N".

Note that the outputs of crossing rules do not contain unpaired strands. The reason is that there are only 8 specific situations that generate unpaired strands, as shown in Fig. 8, and these situations can be applied to all crossing rules. Besides, since the strand is unpaired, we don't need to care about its crossing state.

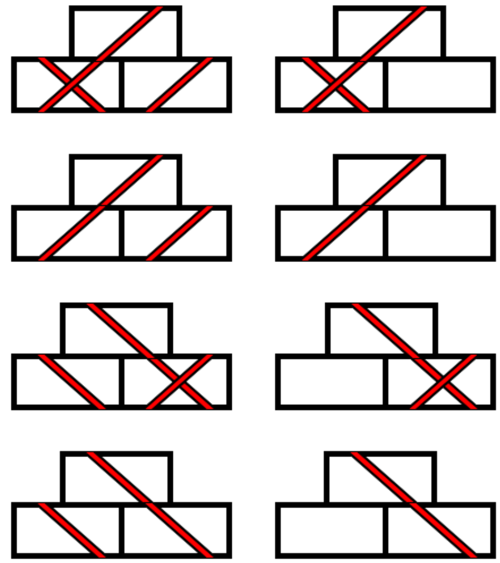

Fig. 8: How an unpaired strand can be generated.

2.3. Additivity

Definition 3. Among all crossing rules, the *additive* rules are those whose values of generated cells are equal to the sum of values in the two cells that generate them modulo 2. If one or both of the two adjacent cells contain an unpaired strand and thus the value is undetermined, then we will regard such situations as satisfying the condition of additivity. Let x_{i-1} and x_i be the two adjacent cells and let $x_i{}'$ be the generated cell. We have $x_i{}' = x_{i-1} + x_i \mod 2$. Note that the value of the unpaired strand is undetermined, so the unpaired strand will not affect the additivity.

Fig. 9 is an example of an additive crossing rule.

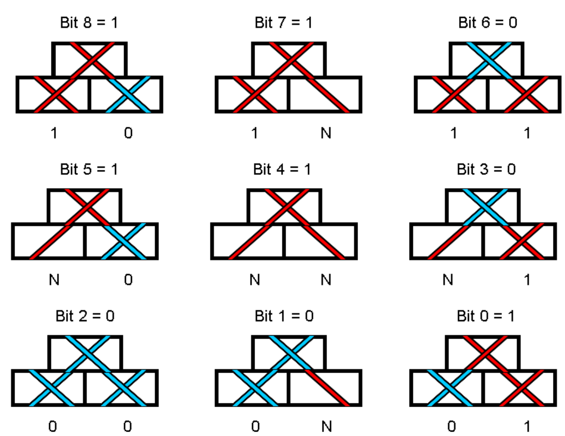

Fig. 9: An example of an additive rule.

Remark 1. Because unpaired strands will not affect the additivity, the additivity is determined by only the 4 situations at the 4 corners in the example shown above. Specifically, additive rules must have a configuration of $1\_0\_\_\_0\_1$ ordered the same as in the example. Hence, there are a total of $2^5 = 32$ additive rules.

2.4. Inverse Additivity

Definition 4. *Inverse additive* rules are similar to additive ones but values of “0” and “1” are flipped. Therefore, in inverse additive rules, we have $x_i{}' = x_{i-1} + x_i + 1 \mod 2$. Accordingly, the configuration of inverse additive rules must be $0\_1\_\_\_1\_0$ and the number of inverse additive rules is also $2^5 = 32$.

Fig. 10 is an example of an inversely additive crossing rule.

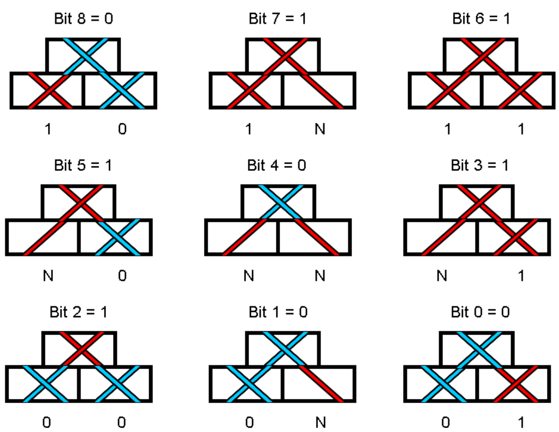

Fig. 10: An example of inversely additive rule.

2.5. Rotation of a Row

In our modeling, a row of a pattern is regarded as a closed loop. Therefore, the cell at one end of the row is adjacent to the cell at the other end. If half of the cell exceeds the border of a row, the other half of the cell will go to the other end of the row.

Definition 5. The row is *rotated* if we put cells from one end to the other end without changing the order of moved cells.

For example, we can obtain the 9^{th} row from bottom in Fig. 4 by rotating the first row. This is why we say the two rows are the same if positions of cells are not considered.

Remark 2. If a starting row is rotated, the generated pattern will be the same as before but also rotated by the same number of cells.

2.6. Effective Width

Although the unpaired strand has an undetermined value, its behavior can still be represented by “1” or “0” based on the values of its adjacent cell and the generated cell. In Fig. 11, for the example at the top, a left unpaired strand and a crossing of value 0 generate a value of 1, so the unpaired strand behaves as a “1” in this situation. For the example at the bottom, a left unpaired strand and a “1” generate a “1”, so the unpaired strand behaves as a “0”. Note that in starting rows with just one unpaired strand, not all cells can change their values during the evolving of pattern because the unpaired strand always exists. For example, if the behavior of the unpaired strand is fixed, the value of this cell will not change. Moreover, if the left unpaired strand behaves as a “0”, then the value x of cell to the right of it will also not change because $0 + x = x$. For the right unpaired strand that behaves as a “0”, the value of cell to its left will not change for the same reason.

Definition 6. Given a starting row with width m, the *effective width*, m_e, is the number of cells whose values will change during the evolving of pattern, plus 1.

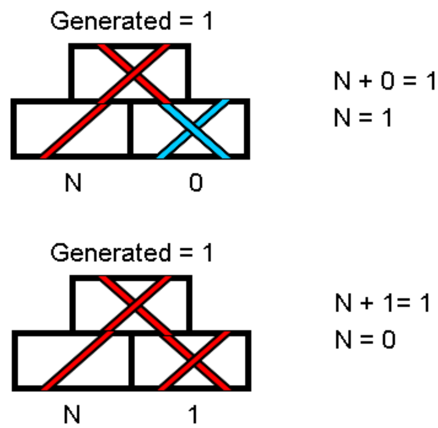

Fig. 11: The value of unpaired strands.

To be more specific, if there is only one unpaired strand, m_e is determined as follows:

1. If the unpaired strand behaves as “1”, then $m_e = m$.
2. If the unpaired strand behaves as “0” and is a left strand, then counting from the unpaired strand to right, $m_e =$ the number of cells from the first “1” to the last cell, which is the cell to the left of the unpaired strand because the row can be rotated.
3. If the unpaired strand behaves as “0” and is a right strand, then counting from the unpaired strand to left, $m_e =$ the number of cells from the first “1” to the last cell, which is the cell to the right of the unpaired strand because the row can be rotated.
4. If the behavior of the unpaired strand can be both “1” and “0” based on the neighboring cell, we treat it as “0”. The reason is that if the unpaired strand behaves as “0” in the starting row, then the value of the neighboring cell will not change during the pattern evolving because for any value x in the neighboring cell, $0 + x = x$. Therefore, the behavior of the unpaired strand will also not change. If it behaves as “1” in the starting row, then the value of the neighboring cell will be flipped in the next row. Afterwards, the behavior of the unpaired strand will also be flipped to “0” and then remain unchanged.

Remark 3. In the latter situation, we need to calculate m_e from the second row because the value change of the cell next to the unpaired strand will affect m_e.

Fig. 12 and Fig. 13 show how the effective width is calculated. Cells whose values are changeable and the leading "1" are represented in yellow-green in the starting row. In the first example, the effective width is 3, and in the second example, the effective width is 5.

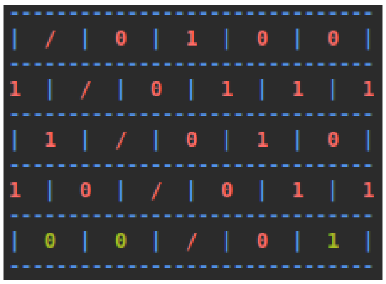

Fig. 12: A starting row with an effective of 3.

Fig. 13: A starting row with an effective of 5.

Remark 4. The rotation of the row will not affect the effective width. Therefore, in the following parts of this paper, I will put the unpaired strand at the beginning or the end of the starting row to make it easier for us to observe.

2.7. Width of a Row in Pascal's Triangle

Definition 7. The *width* of row i in Pascal's Triangle, w_i, is the number of entries in row i in Pascal's Triangle.

Remark 5. In Pascal's Triangle, the amount of entries in each row is incremented by 1 from its previous row, and the first row has 1 entry. Therefore, if we define the first row in Pascal's Triangle as row 0, then $w_i = i + 1$.

3. REPEAT LENGTH

Many factors can affect the length of repeat pattern given the starting row. To narrow the problem down, we will focus on the additive or inverse additive crossing rules and a single unpaired strand in the starting row.

Proposition 1. Under additive crossing rules, if the starting row contains one unpaired strand and one "1", no matter if it is an actual "1" or an unpaired strand equivalent to a "1", and no empty cell, then the generated pattern will be a partial upside down Pascal's Triangle modulo 2. If the width of corresponding row in Pascal's Triangle is bigger than the effective width, the generated pattern will discard the exceeded parts and become a partial Pascal's Triangle modulo 2.

Proof.
In Pascal's Triangle, each number is the sum of two neighboring numbers in the previous row. This is also true for numbers on the borders of Pascal's Triangle if we regard the numbers outside the borders as "0"s. Let $x_i{}'$ denote a number in Pascal's Triangle and let x_{i-1} and x_i denote the two neighboring numbers in the previous row. Then we have $x_i{}' = x_{i-1} + x_i$. Note that $a+b \mod 2 = ((a \mod 2)+(b \mod 2)) \mod 2$, which will soon to be used.

In the starting row of the weaving pattern, there is only one "1". All other cells are filled with "0". According to Definition 3, the value of generated cell in the next row is $x_i{}' = x_{i-1} + x_i \mod 2$. Therefore, the ways of generating Pascal's Triangle and the weaving pattern are the same, and a row in the weaving pattern corresponds to a row in Pascal's Triangle at the same row index.

Because Pascal's Triangle starts with one "1" and there is one "1" in the starting row of the weaving pattern, the generated pattern is an upside-down Pascal's Triangle modulo 2 if the width of corresponding row in Pascal's Triangle is less than or equal to the effective width, m_e.

When the width of corresponding row in Pascal's Triangle exceeds m_e, a crossing that is supposed to neighbor with another crossing will neighbor with the unpaired strand. In this case, an unpaired strand instead of a new crossing is generated (Fig. 8). Therefore, the unpaired strand will remain and isolate the entries of that row in Pascal's Triangle within and outside the effective width. Note that for each generated cell, $x_i{}'$, its value is determined by x_{i-1} and x_i, and $i < m_e$ because all of these cells are in the pattern. Hence, if the corresponding row in Pascal's Triangle exceeds m_e, extra cells will be discarded as shown in Fig. 14, and these cells would not affect the value of cells in the pattern. Consequently, the generated Pascal's Triangle is partial. $\square$

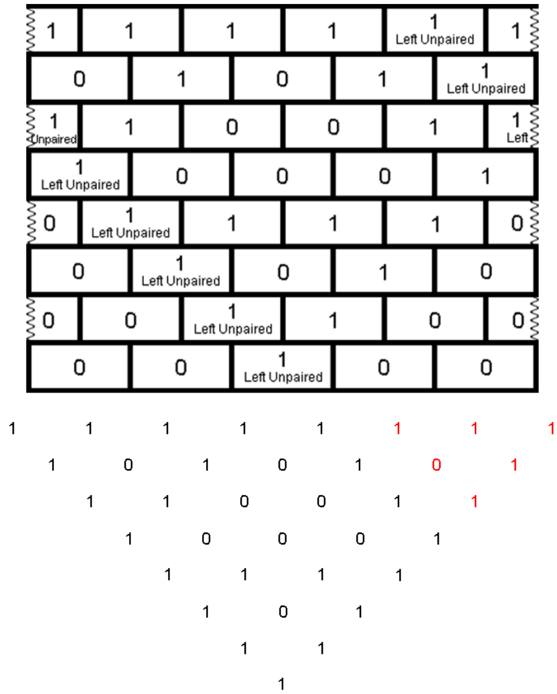

Fig. 14: The generated pattern(top) with $m_e = 5$ and the corresponding Pascal's Triangle(bottom). The red part in the Pascal's Triangle is discarded because it exceeds m_e.

Note that the discarded part is on the left because the unpaired strand is to the left. If the unpaired strand is to the right, the discarded part will be on the right.

Proposition 2. The patterns described in Proposition 1 start to repeat at the row "1 0 ... 0 1" in Pascal's Triangle. The repeat length is 2^{k+1}, where $2^k < m_e \leq 2^{k+1}$.

Proof.
Because there is only one "1" in the starting row of weaving pattern, the pattern will repeat when in the corresponding row in Pascal's Triangle, the total number of consecutive "0"s following the "1" on the border, plus 1, exceeds m_e. Therefore, when the pattern repeats, the first m_e numbers in the corresponding row in Pascal's Triangle are "1 0 ... 0".

Claim: Patterns start to repeat at a row with all "0"s between two "1"s on borders, which looks like "1 0 ... 0 1".

Proof. If we shade the odd numbers in Pascal's Triangle and leave the even numbers blank, we will get a Sierpiński Triangle[3]. Note that we also take the modulus by 2 of Pascal's Triangle so that odd numbers become 1 and even numbers become 0. Therefore, the Sierpiński Triangle is equivalent to Pascal's Triangle modulo 2. Because the Sierpiński Triangle is recursive, Pascal's Triangle modulo 2 is also recursive in the same way[4]. The recursion of Pascal's Triangle modulo 2 is defined as follows:

- Base case: the base case of Pascal's Triangle modulo 2 contains only a single "1".
- Recursive step: the next iteration of the recursion is formed by arranging 3 copies of current iteration as an equilateral triangle and fill all entries in the middle with "0"s.

Fig. 15 and Fig. 16 shows the recursive structure.

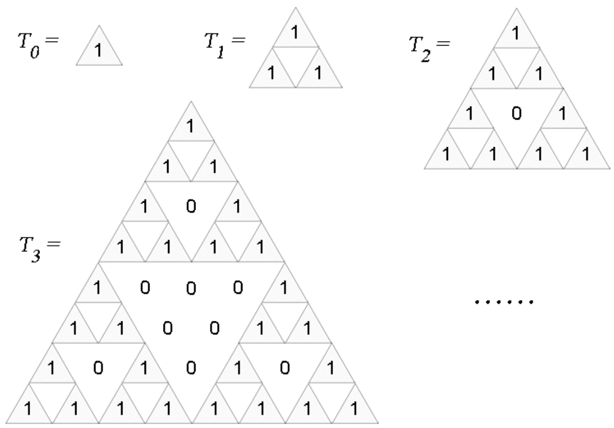

Fig. 15: Base case and first few iterations of the recursive structure.

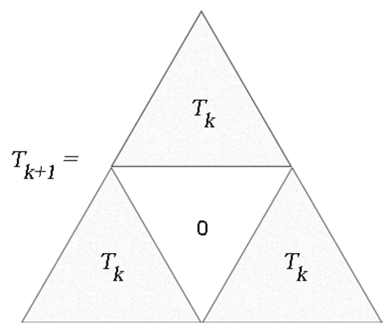

Fig. 16: General recursive structure. (Picture based on Andrew Granville[4])

Suppose, for contradiction, that the pattern starts to repeat at row i with additional "1"s in between the two "1"s on borders. Since Pascal's Triangle is horizontally symmetric[5], we know that row i is "1 0 ... 0 1 ### 1 0 ... 0 1", where "..." is filled with "0"s and "###" is undetermined.

Let l be the width of row i so that $l = i + 1$ and let l' be the width of "1 0 ... 0 1", so $l' \leq \frac{l}{2}$. Because the starting row of the pattern contains only one 1 and the pattern starts to repeat at row i, the first m_e numbers of row i must contain only one 1, which is the 1 on the border. Therefore, $m_e < l'$.

Let T_{k+1} be the sub-triangle in Pascal's Triangle modulo 2 such that row i is in T_{k+1} but not in T_k. According to the recursive structure shown in Fig. 16, row i is made of two copies of a same row, say row j, in T_k, and the middle of the two copies is filled with with "0"s. Note that the "1 0 ... 0 1" cannot be split across the two copies of row j. The reason is that "..." is filled with all 0's, but if it is split across the two copies of row j, then there will be at least one 1 in the "..." part because a row has at least two 1's(except the first row). Therefore, row j contains the first "1 0 ... 0 1" part that is in row i and some of the "$\#\#\#$" part if not all entries in "$\#\#\#$' are "0". Let the width of row j be l_j so that $l_j = j + 1$, so $l_j \geq l' > m_e$. Therefore, the weaving pattern also starts to repeat at row j, but row j is before row i, which contradicts the assumption that the pattern starts to repeat at row i.

Therefore, in the row that the pattern starts to repeat, there must be no "1"s between the two "1"s on borders, which means that the row is "1 0 ... 0 1". □

Claim: All rows of form "1 0 ... 0 1" are located at the $2^n th$ rows in Pascal's Triangle where n is a positive integer, and for every positive integer n, the $2^n th$ row has the form "1 0 ... 0 1".

Proof. Note that the row "1 0 ... 0 1" is generated from the row "1 1 ... 1 1". The reason is that consecutive "0"s must be generated from either consecutive "0"s or consecutive "1"s. Otherwise, adjacent 0 and 1 will generate additional "1"s. Since the borders of Pascal's Triangle are composed of "1"s, the row "1 0 ... 0 1" must be generated from row "1 1 ... 1 1".

Let row "1 0 ... 0 1" be in T_k such that row "1 0 ... 0 1" is in T_k but not in T_{k-1}. Such row corresponds to the two triangles at the bottom in Fig. 16. If we look at the bottom half of the recursive structure, it is clear that there are at least four "1"s in each row except the row at the beginning of the bottom half. Therefore, row "1 0 ... 0 1" must correspond to this row in Pascal's Triangle. Note that the height of any T_k is a power of 2 because its height is twice as the height of T_{k-1}, and the height of the smallest recursive structure, which contains a single "1", is 1. Because the index of the first row in Pascal's Triangle is 0, the index of row "1 0 ... 0 1" is $0 + 1 \times 2^n = 2^n$. For the same reason, for any positive integer n, the 2^n rows starting from top of Pascal's Triangle modulo 2 form a complete recursive iteration $T_{\log_2 n}$, whose last row is of the form "1 1 ... 1 1" and has an index of $2^n - 1$. Therefore, the next row is of the form "1 0 ... 0 1" and has an index of 2^n. □

Note that the width of row i in Pascal's Triangle is equal to $i + 1$ (Remark 5), so the width of row "1 0 ... 0 1" where the pattern starts to repeat and whose index is 2^n for some positive integer n, is $2^n + 1$. When the pattern starts to repeat, the row width must be greater than the effective width m_e, so we get $m_e < 2^n + 1$. Hence, $m_e \leq 2^n$. Also note that because row 2^n is the first row for the pattern to repeat, the pattern does not repeat at row 2^{n-1}, which is also "1 0 ... 0 1". Since the pattern does not repeat at row 2^{n-1}, its width, $2^{n-1} + 1$, must be no longer than the effective width. Therefore, we get $m_e \geq 2^{n-1} + 1$, which means that $m_e > 2^{n-1}$. Let $k = n - 1$, so $2^k < m_e \leq 2^{k+1}$. Because the pattern starts to repeat at the $2^n\ th$ row, the repeat length $= 2^n - 0 = 2^{k+1}$. □

Proposition 3. When in the starting row, there are multiple unpaired strands that are in the same direction, we divide the starting row into sub-rows by unpaired strands and calculate the effective width of each sub-row. The effective width, m_e, of the starting row is equal to the largest effective width among all sub-rows.

Proof. Each unpaired strand has two adjacent cells. It will generate a new crossing with one of the two cells and transfer the unpaired strand to the next row with the other, according to Fig. 8.

Claim: If all unpaired strands are in the same direction, then the number of cells between two unpaired strands will not change in the generated rows.

Proof. If the unpaired strands are in the left direction, we count from left to right; if the unpaired strands are in the right direction, we count from right to left. In this way, if we denote the cell of an unpaired strand as c_i, then c_i and c_{i-1} will generate an unpaired strand and c_i and c_{i+1} will generate a new crossing if c_{i+1} is not an unpaired strand.

Suppose there are k crossings between the two unpaired strand in a given row. Let the first unpaired strand be c_0, the second unpaired strand be $c_0\ '$, and crossings be c_1, c_2, ... , c_k. Let the cell prior to the first unpaired strand be c_{-1}. Therefore, c_{-1} and c_0, and c_k and c_0', will generate an unpaired strand. For $1 \leq i \leq k$, c_{i-1} and c_i will generate a crossing. Accordingly, the number of cells between two unpaired strands does not change in the generated cell. □

Because of the claim we just proved, we can divide the starting row into sub-rows by unpaired strands. Each unpaired strand c_i will be assigned to the sub-row that contains cell c_{i+1}. Crossings in different sub-rows will not affect the value of each other. Hence, we can treat the generated patterns of sub-rows separately.

Each sub-row contains 1 unpaired strand so that we can calculate the repeat length of its generated pattern according to Proposition 2. The generated pattern of the entire starting row is to simply combine patterns of all sub-rows. Therefore, the repeat length of the entire starting row is determined by the sub-row with the largest effective width. In other words, the effective width of the starting row is equal to the largest effective width among all sub-rows. □

4. CONCLUSION

4.1. Results

We have proved the exact repeat length of the generated pattern under additive or inversely additive crossing rules with at least one unpaired strand in the starting row and all unpaired strands in the same direction. To get the repeat length under such situations, we first calculate the effective width, m_e, of the starting row, and the repeat length is equal to 2^{k+1}, where $2^k < m_e \leq 2^{k+1}$. Note that positions of cells are not considered in this paper.

4.2. Future Work

Currently, this paper only covers situations with one unpaired strand or multiple unpaired strands with the same direction. The next thing to do is to generalize the repeat length to starting rows with no unpaired strands or multiple unpaired strands in different directions.

4.3. Conjectures

The following conjectures are for starting rows with one "1", no unpaired strands, its width even and not a power of 2, under additive or inversely additive crossing rules.

- When the pattern starts to repeat, there are two "1"s in that row. Note that the starting row is not contained in the repeat pattern in this case.
- For a row with two "1"s, if it is a row where the pattern starts to repeat, the number of "0"s between the two "1"s is equal to $2^n - 1$ counting from both inner and outer side, $n \in \mathbb{N}$, not necessarily the same for both.
- For a row with two "1"s, if it is not the row where the pattern starts to repeat, the number of "0"s between the two "1"s is equal to $2^n - 1$ counting from either inner or outer side, but not for both, $n \in \mathbb{N}$.

5. CODE FOR SIMULATION

The code for generating a simulated weaving pattern is available at: `https://github.com/kevin1zc/Weaving-Pattern-Simulation`. Instructions of how to use it is in the README.md file.

6. ACKNOWLEDGEMENTS

1. This paper is advised by Dr. Joshua Holden from Rose-Hulman Institute of Technology. Thanks to Dr. Holden for all of the help and inspiration he provided.
2. Thanks to Hao Yang from Rose-Hulman Institute of Technology for providing a practical idea of this paper and the code.

7. REFERENCES

[1] J. Holden, "The complexity of braids, cables, and weaves modeled with stranded cellular automata," *Proceedings of Bridges 2017: Mathematics, Music, Art, Architecture, Education, Culture*, pp. 463–466, 2017.

[2] Hao Yang, "Stranded cellular automaton and weaving products," *Mathematical Sciences Technical Reports (MSTR)*, 2018, Available at: `https://scholar.rose-hulman.edu/math_mstr/168/`.

[3] Wikipedia contributors, "Sierpiski triangle — Wikipedia, the free encyclopedia," `https://en.wikipedia.org/w/index.php?title=Sierpi%C5%84ski_triangle&oldid=914419677`, 2019, [Online].

[4] Andrew Granville, "Zaphod Beeblebrox's brain and the fifty-ninth row of Pascal's triangle," *The American Mathematical Monthly*, vol. 99, pp. 318–331, Apr. 1992.

[5] Rod Pierce, "Pascal's Triangle," `http://www.mathsisfun.com/pascals-triangle.html`, July 2018, [Online].