Iterated Strings and Cellular Automata

Oleksiy Andriychenko
and
Marc Chamberland[1]
Department of Mathematics and Computer Science
Grinnell College
Grinnell, IA, 50112-0806
E-mail: chamberl@math.grin.edu

1 Introduction

In 1996, Sir Bryan Thwaites[4] posed two open problems with prize money offered for solutions. The first problem (with a £1000 reward) is the well-known $3x+1$ problem which has received attention from many quarters. This easily-stated problem has alluded mathematicians for about fifty years; for more information, see Lagarias[1] and Wirsching[5]. Thwaites's other problem (with a £100 reward) has no clear origin. He states it as follows:

> *Take any set of N rational numbers. Form another set by taking the positive differences of successive members of the first set, the last such difference being formed from the last and first members of the original set. Iterate. Then in due course the set so formed will consist entirely of zeros if and only if N is a power of two.*

[1] Send correspondence to Marc Chamberland.

Thwaites concludes his note by saying that "Although neither I, nor others who have been equally intrigued, have yet proved [the second problem], one's instinct is that here is a provable conjecture; and so the prize for the first successful proof, or disproof, is a mere hundred pounds." The present paper offers an elementary proof of this second problem. In the process, binomial coefficients and cellular automata are encountered.

We will let $(a_1, a_2, \ldots, a_n)$ represent a *string* of length n where a_i is rational for all i. Upon iteration, its successor will be $(|a_1 - a_2|, |a_2 - a_3|, \ldots, |a_{n-1} - a_n|, |a_n - a_1|)$. A string containing only zeros will be called the *zero-string* while a string containing only ones will be called the *one-string*. The way the problem was posed by Thwaites is somewhat imprecise; the one-string iterates to the zero-string regardless of the string's length. We restate the (proper) theorem to be proved formally:

Theorem 1.1 *All strings of length n will eventually iterate to the zero-string if and only if $n = 2^k$ for some $k \in \mathbb{Z}^+$.*

Half of the proof comes easily:

Theorem 1.2 *If the string's length n is not a power of two, then there exist strings which will never iterate to the zero-string.*

Proof: The proof considers 0-1 strings, strings whose terms take only the values 0 or 1. Since the set of 0-1 strings is forward-invariant under our iterative process, this proof will restrict its attention to only 0-1 strings.

First we prove the case when n is odd. Working backwards, note that the only predecessor of the zero-string is the one-string. The only predecessor of the one-string has terms which alternate between 0 and 1 which is impossible since n is odd. Therefore the only 0-1 strings of odd length iterating to the zero-string are the zero-string itself and the one-string, thus completing the proof when n is odd.

If n is an even number which is not a power of two, it must have an odd prime factor, say p. Create a string of length n by concatenating n/p substrings of length p, each of which is the string starting with a one then having all zero terms. For example, if $n = 12$, take $p = 3$ and create the string

$$1\,0\,0\;\;1\,0\,0\;\;1\,0\,0\;\;1\,0\,0$$

The periodic nature of the iterative process implies that each substring iterates as if it was the whole string:

$$1\,0\,0 \;\;\rightarrow\;\; 1\,0\,1$$

$$1\,0\,0\;\;1\,0\,0\;\;1\,0\,0\;\;1\,0\,0 \;\;\rightarrow\;\; 1\,0\,1\;\;1\,0\,1\;\;1\,0\,1\;\;1\,0\,1$$

Since each of the (odd-length) substrings will never iterate to the zero string, neither will the whole string, which completes the proof. □

The previous proof considered only the set of 0-1 strings. To prove the other half of Theorem 1.1, we argue that considering only 0-1 strings is sufficient. First note that by scaling a string by a constant, the dynamics do not change, so multiply each element in the string by the appropriate integer (the least common multiple of the denominators) to yield an integer string. Also, one iteration on a string yields a non-negative string, so we can assume from here on that the string consists only of non-negative integers. Next, we show that it is sufficient to consider strings whose values are only 0 and possibly one other (positive) value. To do this, we show that if the string contains at least two distinct positive values, the maximum value (denoted henceforth by m) will eventually decrease. If there is no zero value, the maximum value will automatically decrease after one iteration, so we may assume there is at least one zero value. Consider any substring whose terms are only zero or m (with at least one m), and assume this substring is maximal so that it takes the form

$$b\; a_j\; a_{j+1}\;\; \cdots \;\; a_{j+l}\; c$$

Figure 1: Iterating the string (1 1 0 0 1 1 0 0)

where a_k equals zero or m (with at least one m) for all k and $0 < b, c < m$. After one iteration, the substring has one less term. Note that such substrings (with at least one m) cannot be created, so after a finite number of iterations, these substrings all vanish. This process forces the maximum of the whole string to decrease, leaving us (dynamically) with two possibilities: either this descent continues until all the terms are zero, or the string iterates until all its terms are either 0 or possibly one positive value. Dividing each term by this positive value (which leaves the string dynamically unaltered) yields a string whose terms are only 0 or 1.

Figure 1 shows iterations of the string (1 1 0 0 1 1 0 0), where the black dots represent 1 and the white dots represent 0.

At this point, it is worth pointing out that iterating a 0-1 string mirrors the dynamics used in generating the Sierpinski Gasket with cellular automata. Consider the "rules" in Figure 2. For each rule, the parity (black or white) of the upper squares determines the parity of the lower square. Starting with an infinite row with only one black square, one generates the Sierpinski Gasket in a stretched form. Figure 3 shows the first few rows. The black cells in this figure correspond to the odd terms in Pascal's triangle, where the top black cell corresponds to the apex of the triangle. Details of the mathematics may be

Figure 2: Cellular automata “Rules”

Figure 3: Generating the Sierpinski Gasket

found in Peitgen *et al.*[3]. The dynamics of our 0-1 strings are similar with the important difference that the string is periodic.

To finish the proof of Theorem 1.1, one is required to show that 0-1 strings whose length is a power of 2 eventually iterate to the zero string. The analysis is simplified if we replace 0 (resp. 1) with 1 (resp. -1) and instead of using the absolute value of the difference, simply consider the product. For example, before we had the successive terms $(1 \;\; 0)$ produce $|1-0| = 1$, whereas now we have $(-1 \;\; 1)$ produce $(-1)(1) = -1$. One may easily verfiy that the dynamics are equivalent; we are simply representing the group $\mathbf{Z}_2$ in a different way. The rest of the proof will work with this new system. We will let $a_{i,j}$ denote the value of the j^{th} element of the string after i iterations. For ease of notation, it will be understood that if $kn < j \leq (k+1)n$ for some $k \in \mathbf{Z}^+$, then $a_{i,j} = a_{i,j-kn}$.

Lemma 1.1 *If a string has length n, then*

$$a_{i,j} = a_{0,j}^{\binom{i}{0}} a_{0,j+1}^{\binom{i}{1}} \cdots a_{0,j+i}^{\binom{i}{i}}$$

for $1 \leq i \leq n$, $1 \leq j \leq n-i$.

The proof is based on induction.

Lemma 1.2

$$\begin{pmatrix} 2^k - 1 \\ j \end{pmatrix} \textit{ is odd if } 0 \leq j \leq 2^k - 1.$$

The proof is easily derived by expanding the binomial coefficient. We note that this lemma has a generalization (see, for example, [2]): for any prime p,

$$p \textit{ does not divide } \begin{pmatrix} p^k - 1 \\ j \end{pmatrix} \textit{ if } 0 \leq j \leq p^k - 1$$

These two lemmas lead to the last step in the proof of Theorem 1.1:

Theorem 1.3 *If the string's length is $n = 2^k$, then $a_{n,j} = 1$ for all j.*

Proof: The first step is to show

$$a_{n-1,j} = a_{0,1}\ a_{0,2}\ \cdots\ a_{0,n}$$

for $j = 1, \ldots, n$. Using Lemmas 1.1 and 1.2 successively, we have

$$\begin{aligned} a_{n-1,j} &= a_{0,j}^{\binom{n-1}{0}} a_{0,j+1}^{\binom{n-1}{1}} \cdots a_{0,j+n-1}^{\binom{n-1}{n-1}} \\ &= a_{0,j} a_{0,j+1} \cdots a_{0,j+n-1} \\ &= a_{0,1} a_{0,2} \cdots a_{0,n} \end{aligned}$$

If $a_{n-1,j} = 1$ for all j, we are finished. If all the terms are -1, one more iteration forces $a_{n,j} = 1$ for all j.

□

Acknowledgement: The authors would like to thank Grinnell College for financially supporting O.A. to work with M.C. during the summer of 1999.

References

[1] J.C. Lagarias. *The* $3x+1$ *Problem and its Generalizations. American Mathematical Monthly*, 92:3–23, 1985.

[2] I. Niven, H.S. Zuckerman and H.L. Montgomery. *An Introduction to the Theory of Numbers.* Wiley, 1991.

[3] H.-O. Peitgen, H. Jürgens and D. Saupe. *Chaos and Fractals.* Springer-Verlag, 1992.

[4] B. Thwaites. *Two Conjectures or how to win £1100. Mathematical Gazette*, 80:35–36, 1996.

[5] G.J. Wirsching. *The Dynamical System Generated by the* $3n+1$ *Function.* Lecture Notes in Mathematics, 1681, Springer-Verlag, 1998.