Automata 2010 — 16th Intl. Workshop on CA and DCS DMTCS proc. **AL**, 2010, 1–18

Faster Methods for Identifying Nontrivial Energy Conservation Functions for Cellular Automata

Leemon Baird and Barry Fagin

Academy Center for Cyberspace Research, Department of Computer Science, US Air Force Academy, Colorado Springs, Colorado USA 80840

The biggest obstacle to the efficient discovery of conserved energy functions for cellular auotmata is the elimination of the trivial functions from the solution space. Once this is accomplished, the identification of nontrivial conserved functions can be accomplished computationally through appropriate linear algebra.

As a means to this end, we introduce a general theory of trivial conserved functions. We consider the existence of nontrivial additive conserved energy functions (”nontrivials”) for cellular automata in any number of dimensions, with any size of neighborhood, and with any number of cell states. We give the first known basis set for all trivial conserved functions in the general case, and use this to derive a number of optimizations for reducing time and memory for the discovery of nontrivials.

We report that the Game of Life has no nontrivials with energy windows of size 13 or smaller. Other 2D automata, however, do have nontrivials. We give the complete list of those functions for binary outer-totalistic automata with energy windows of size 9 or smaller, and discuss patterns we have observed.

Keywords: nontrivial conserved energy function, trivial conserved energy function, 1D cellular autamata, 2D cellular automata, Game of Life

1 Preliminaries: basic definitions

We consider cellular automata with k states in n dimensions. The *neighborhood* of a cellular automaton is the region of surrounding cells used to determine the next state of a given cell. The *window* of an energy function for a cellular automaton is the region of adjacent cells that contribute to the function. Both neighborhoods and windows are n-dimensional tensors, with the size of each dimension specified as a positive integer. Given the size of such a tensor, it is useful to define the following 3 sets of tensors.

Definition 1.1. Cellular automata are composed of cells, each of which is in one of k states (or *colors*) at any given time. The set $\mathcal{C}$ is the set of such colors, and the set $\mathcal{C}_*$ is that set augmented with another color named *. (* denotes a special state with certain properties that simplify our proofs. It is explained in more detail in the pages that follow.)

$$\mathcal{C} = \{0, 1, 2, \ldots, k-1\} \tag{1.1}$$

$$\mathcal{C}_* = \mathcal{C} \cup \{*\} \tag{1.2}$$

It is sometimes useful to choose one color to be treated specially. In all such cases, the color 0 will be chosen.

Definition 1.2. An n-dimensional *cellular automaton rule* is a function R that gives the color of a given cell on the next time step as a function of a neighborhood of cells centered on that cell on the current time step. The neighborhood is an n-dimensional tensor of size $w_1 \times \cdots \times w_n$, where each w_i is an odd, positive integer.

$$R : C^{w_1 \times \cdots \times w_n} \to \mathcal{C} \tag{1.3}$$

Definition 1.3. An n-dimensional *cellular automaton* is an n-dimensional tensor whose elements are in $\mathcal{C}$, and which is updated on each time step according to a cellular automaton rule R, applied to every cell in parallel. The rule is a function applied to each cell and its neighbors, where neighbors wrap toroidally (i.e. the top edge is considered adjacent to the bottom, the left edge is adjacent to the right, and so on for each dimension).

Definition 1.4. The *successor function* advances a region within a cellular automaton one time step by applying a rule R to a region M of size $s_1 \times \cdots \times s_n$

$$T : (\mathcal{C}^{w_1 \times \cdots \times w_n} \to \mathcal{C}) \times \mathcal{C}^{s_1 \times \cdots \times s_n} \to \mathcal{C}^{(s_1 - w_1 + 1) \times \cdots \times (s_n - w_n + 1)}$$

which is defined as:

$$T(R, M) = M' \text{ where } M'_{i_1, \ldots, i_n} = R(M_{(i_1 \ldots i_1 + w_1 - 1), \ldots, (i_n \ldots i_n + w_n - 1)}) \tag{1.4}$$

Note that $T(R, M)$ is defined for an M that is only a portion of the cells, and so it does not wrap around toroidally. Instead, it returns a tensor that is smaller than M in each dimension. Also note that the ellipses on the right side of the equation are used in two different ways. Each element of the result comes from applying the R function to only a portion of the M tensor, which includes those elements of M whose first coordinate is in the range $[i_1, i_1 + w_1 - 1]$, and whose second coordinate is in the range $[i_2, i_2 + w_2 - 1]$, and so on up to the nth coordinate being in the range $[i_n, i_n + w_n - 1]$.

Definition 1.5. A *linear additive energy function* (or *energy function*) is a function $f : \mathcal{C}^{s_1 \times \cdots \times s_n} \to \mathbb{R}$ that assigns a real number to a window of size $s_1 \times \cdots \times s_n$ within a cellular automaton.

Definition 1.6. The *total energy* $e_{tot} : \mathcal{C}^{u_1 \times \cdots \times u_n} \to \mathbb{R}$ of a given state U of an entire cellular automaton universe with $u_1 \times \cdots \times u_n$ cells, with respect to a given energy function f, is

$$e_{tot}(U) = \sum_W f(U_W) \tag{1.5}$$

where U is the universe state for a cellular automaton, W is the position of the energy window within that universe, and U_W is that window within the universe, which wraps toroidally at the edges of the universe.

Definition 1.7. A *conserved linear additive energy function* (or a *conserved function*) for a given cellular automaton rule is an energy function that for a universe of any size, and for any given state of that universe, will assign the same total energy to that universe for both that state and its successor.

Definition 1.8. A *trivial conserved linear additive energy function* (or a *trivial*) is an energy function that for a universe of any size, will assign the same total energy to that universe regardless of its state. A *nontrivial conserved linear additive energy function* (or a *nontrivial*) for a given cellular automaton rule is a conserved energy function that is not trivial.

Definition 1.9. Given n positive integers $s_1, \ldots, s_n$ defining the size of an n-dimensional tensor, the set $\mathcal{B}(s_1, \ldots, s_n)$ is the set of all tensors over $\mathcal{C}$ of that size. This set is partitioned into two sets, $\mathcal{Z}(s_1, \ldots, s_n)$, the *zero-sided tensors*, which have at least one side that contains the origin element and is filled entirely with zero elements, and $\bar{\mathcal{Z}}(s_1, \ldots, s_n)$, the *non-zero-sided tensors*, which do not have such a side. The *origin element* is the element of the tensor at location $(1, 1, \ldots, 1)$.

$$\mathcal{B}(s_1, \ldots, s_n) = \mathcal{C}^{s_1 \times \cdots \times s_n} \tag{1.6}$$
$$\mathcal{Z}(s_1, \ldots, s_n) = \{T \in \mathcal{B}(s_1, \ldots, s_n) \mid \exists i \forall j \forall s_j \; T_{s_1, \ldots, s_{i-1}, 1, s_{i+1}, \ldots, s_n} = 0\} \tag{1.7}$$
$$\bar{\mathcal{Z}}(s_1, \ldots, s_n) = \mathcal{B}(s_1, \ldots, s_n) \setminus \mathcal{Z}(s_1, \ldots, s_n) \tag{1.8}$$

So in 1 dimension, the zero-sided vectors are those whose with a 0 as the first element. In 2 dimensions, the zero-sided matrices are those with a top row of all zeros, or a leftmost column of all zeros, or both.

It is useful to define a matching function H that can be used in the construction of various functions over these tensors. The function returns 1 iff two tensors have elements that match, where the * symbol is treated as matching any color.

Definition 1.10. Given n-dimensional tensors over $\mathcal{C}_*$, the function $H : \mathcal{C}_*^{s_1 \times \cdots \times s_n} \times \mathcal{C}_*^{s_1 \times \cdots \times s_n} \to \{0, 1\}$ is defined as

$$H(A, B) = \begin{cases} 1 & \text{if } \forall i \forall s_i \; A_{s_1, \ldots, s_n} = B_{s_1, \ldots, s_n} \\ & \qquad \vee A_{s_1, \ldots, s_n} = * \\ & \qquad \vee B_{s_1, \ldots, s_n} = * \\ 0 & \text{otherwise} \end{cases} \tag{1.9}$$

Given an n-dimensional tensor, it is useful to unwrap it into a 1D string of characters. This will be done in *row major order*. For matrices, this means the elements will be read from left to right across the top row, then left to right across the second row, and so on down to the bottom row. Tensors of other dimensionalities are unwrapped similarly, with the last dimension changing most quickly, and the first dimension changing most slowly. It is useful to have a function $V_{num}(T)$ that unwraps the elements of tensor T, then converts the resulting string to an integer by treating it as a number in base c, with the first element being the most significant digit, and the last being the least significant.

Definition 1.11. An n-dimensional tensor $\mathcal{A}$ with elements in $\mathcal{C}$ can be converted to an integer by the function $V_{num} : \mathcal{C}^{s_1 \times \cdots \times s_n} \to \mathbb{Z}$, which treats the elements of the tensor as digits base k, where the elements are taken in row major order, treating the first as the least significant digit, and the last as the most significant.

$$V_{num}(A) = \sum_{i_1=1}^{s_1} \sum_{i_2=1}^{s_2} \cdots \sum_{i_n=1}^{s_n} A_{i_1,i_2,\ldots,i_n} \prod_{j=1}^{n} k^{(i_j-1)\prod_{m=j+1}^{n} s_m} \tag{1.10}$$

For this definition, the rightmost product is understood to be 1 for all cases where the lower bound exceeds the upper.

Definition 1.12. An n-dimensional tensor with elements in $\mathcal{C}$ can be converted to a binary vector by the function $V_t : \mathcal{C}^{s_1\times\cdots\times s_n} \to \{0,1\}^{(k^{s_1 s_2 \cdots s_n})}$, which is defined as

$$V_t(M) = v \text{ where } v_i = \begin{cases} 1 & \text{if } i = V_{num}(M)+1 \\ 0 & \text{otherwise} \end{cases} \tag{1.11}$$

The vector $V_t(M)$ has one element for each possible color pattern for a tensor of the same size as M. That vector will be all zeros, except for a 1 in the position corresponding to the pattern M.

Definition 1.13. A function $f : \mathcal{C}^{s_1\times\cdots\times s_n} \to \mathbb{R}$ can be converted to a real vector with $k^{s_1 s_2\cdots s_n}$ elements by the function $V : (\mathcal{C}^{s_1\times\cdots\times s_n} \to \mathbb{R}) \to \mathbb{R}^{k^{s_1 s_2\cdots s_n}}$, which is defined as

$$V(f) = \sum_{M\in\mathcal{B}(s_1,\ldots,s_n)} f(M)\cdot V_t(M) \tag{1.12}$$

This vector is a convenient way to represent an energy function. It completely specifies the energy function, by listing the output of the function for every possible input. We will define various classes of energy functions by simultaneous linear equations, treating the elements of this vector as the variables.

Note that the energy function window is independent of the CA neighborhood. Energy functions can be defined over regions different from the scope of the transition rule of the CA. Our work with 1D CAs in [1], for example, has identified conserved energy functions with windows of size 1×5, 1×6 and larger, for CAs that have neighborhoods of size 1×3.

Definition 1.14. Given tensor M of size $m_1\times\cdots\times m_n$, which is a region within an n-dimensional universe, and given an energy window size of $s = (s_1,\ldots,s_n)$, a vector representing the total energy of all energy windows that fit within M can be found with the function

$$e : \mathbb{Z}^n \times \mathcal{C}^{m_1\times\cdots\times m_n} \to \mathbb{Z}^{k^{s_1 s_2\cdots s_n}}$$

which is defined as

$$e(s,M) = \sum_{i_1=1}^{m_1-s_1+1} \sum_{i_2=1}^{m_2-s_2+1} \cdots \sum_{i_n=1}^{m_n-s_n+1} V_t(M_{i_1\ldots i_1+s_1-1,\ldots,i_n\ldots i_n+s_n-1}) \tag{1.13}$$

The $e(s,M)$ function slides the energy window to all possible positions that fit entirely within the matrix M, and finds the energy at each position. It then sums all the energies coming from identical patterns, and constructs a vector with the total energy derived from each possible pattern. The sum of the elements of this vector would simply be the total energy of M. But it is useful to maintain the vector of separate values when generating sets of linear equations that define the trivials, the nontrivials, or the conserved functions.

Definition 1.15. For a positive integer n, the function $N : \mathbb{Z}^n \to \mathbb{Z}$ is defined as

$$N(s_1, \ldots, s_n) = \sum_{b=1}^{2^n-1} k^{\prod_i s_i - b_i} (-1)^{1+\sum_i b_i} \tag{1.14}$$

where b_i is the ith bit of integer b written in binary, with bit 1 being least significant and bit n being most.

In 1 and 2 dimensions this reduces to:

$$N(c) = k^{c-1} \tag{1.15}$$

$$N(r, c) = k^{(r-1)c} + k^{r(c-1)} - k^{(r-1)(c-1)} \tag{1.16}$$

It will be proved below that this gives the cardinality of many of the sets that will be considered here. It equals the number of zero-sided tensors of a given size, the number of trivials, and the number of unit complements. And when subtracted from a simple power of 2, it yields the number of non-zero-sided tensors, the number of equations defining the conserved functions, and the number of equations defining the nontrivials. These terms are defined and the counts proved below.

Definition 1.16. In n dimensions, the seven transforms that operate on tensors of size $s_1 \times \cdots \times s_n$

$$P_C : \mathcal{C}_*^{s_1 \times \cdots \times s_n} \to \mathcal{C}_*^{s_1 \times \cdots \times s_n} \tag{1.17}$$

$$P_* : \mathbb{Z} \times \mathcal{C}_*^{s_1 \times \cdots \times s_n} \to \mathcal{C}_*^{s_1 \times \cdots \times s_n} \tag{1.18}$$

$$P_{rot} : \mathbb{Z} \times \mathcal{C}_*^{s_1 \times \cdots \times s_n} \to \mathcal{C}_*^{s_1 \times \cdots \times s_n} \tag{1.19}$$

$$P_{LD} : \mathbb{Z} \times \mathcal{C}_*^{s_1 \times \cdots \times s_n} \to \mathcal{C}_*^{s_1 \times \cdots \times s_n} \tag{1.20}$$

$$P_{RD} : \mathbb{Z} \times \mathcal{C}_*^{s_1 \times \cdots \times s_n} \to \mathcal{C}_*^{s_1 \times \cdots \times s_n} \tag{1.21}$$

$$P_L : \mathcal{C}_*^{s_1 \times \cdots \times s_n} \to \mathcal{C}_*^{s_1 \times \cdots \times s_n} \tag{1.22}$$

$$P_R : \mathcal{C}_*^{s_1 \times \cdots \times s_n} \to \mathcal{C}_*^{s_1 \times \cdots \times s_n} \tag{1.23}$$

are defined to be:

$$P_C(M) = M' \text{ where } M'_{i_1,\ldots,i_n} = \begin{cases} 0 & \text{if } \forall j \; i_j = \lceil s_j/2 \rceil \\ M_{i_1,\ldots,i_n} & \text{otherwise} \end{cases} \tag{1.24}$$

$$P_*(d, M) = M' \text{ where } M'_{i_1,\ldots,i_n} = \begin{cases} * & \text{if } i_d = 1 \\ M_{i_1,\ldots,i_n} & \text{otherwise} \end{cases} \tag{1.25}$$

$$P_{rot}(d, M) = M' \text{ where } M'_{i_1,\ldots,i_n} = M_{i_1,\ldots,i_{d-1},\ 1+(i_d \bmod s_d)\ ,i_{d+1},\ldots,i_n} \tag{1.26}$$

$$P_{LD}(d, M) = \begin{cases} P_*(d, M) & \text{if } \forall j \forall i_j \; M_{i_1,\ldots,i_{d-1},1,i_{d+1},\ldots,i_n} \in \{0, *\} \\ M & \text{otherwise} \end{cases} \tag{1.27}$$

$$P_{RD}(d, M) = \begin{cases} P_{rot}(P_*(d, M)) & \text{if } \forall j \forall i_j \; M_{i_1,\ldots,i_{d-1},1,i_{d+1},\ldots,i_n} \in \{0, *\} \\ M & \text{otherwise} \end{cases} \tag{1.28}$$

$$P_L(M) = P_{LD}(1, P_{LD}(2, \ \ldots \ P_{LD}(n, M) \ \ldots)) \tag{1.29}$$

$$P_R(M) = P_{RD}(1, P_{RD}(2, \ \ldots \ P_{RD}(n, M) \ \ldots)) \tag{1.30}$$

The function $P_{rot}(d, M)$ rotates the elements of tensor M along dimension d, so that one side that included the origin moves to the opposite side. The function P_C sets the central element to zero. The function P_L transforms a zero-sided tensor by replacing the 0 elements on each all-zero side with * elements. And P_R does the same, then rotates it so each modified side moves to the opposite side. The functions P_*, P_{LD}, and P_{RD} are only used here to define the other functions, and won't be used again.

The following gives three examples of P_L and P_R applied to zero-sided matrices of size 3×5. In each example, M is a zero-sided matrix, where the all-zero side is on the left, top, and both, respectively:

$$M = \begin{array}{ccccc} 0&1&1&1&1 \\ 0&0&1&0&1 \\ 0&0&1&0&0 \end{array} \qquad P_L(M) = \begin{array}{ccccc} *&1&1&1&1 \\ *&0&1&0&1 \\ *&0&1&0&0 \end{array} \qquad P_R(M) = \begin{array}{ccccc} 1&1&1&1&* \\ 0&1&0&1&* \\ 0&1&0&0&* \end{array} \tag{1.31}$$

$$M = \begin{array}{ccccc} 0&0&0&0&0 \\ 1&0&1&1&1 \\ 0&0&0&0&1 \end{array} \qquad P_L(M) = \begin{array}{ccccc} *&*&*&*&* \\ 1&0&1&1&1 \\ 0&0&0&0&1 \end{array} \qquad P_R(M) = \begin{array}{ccccc} 1&0&1&1&1 \\ 0&0&0&0&1 \\ *&*&*&*&* \end{array} \tag{1.32}$$

$$M = \begin{array}{ccccc} 0&0&0&0&0 \\ 0&1&0&0&0 \\ 0&1&0&1&0 \end{array} \qquad P_L(M) = \begin{array}{ccccc} *&*&*&*&* \\ *&1&0&0&0 \\ *&1&0&1&0 \end{array} \qquad P_R(M) = \begin{array}{ccccc} 1&0&0&0&* \\ 1&0&1&0&* \\ *&*&*&*&* \end{array} \tag{1.33}$$

Definition 1.17. The function $P_Z : \mathcal{C}^{s_1 \times \cdots \times s_n} \to \mathcal{C}^{(2s_1-1) \times \cdots \times (2s_n-1)}$ takes a small n-dimensional tensor and pads it with zero elements on many of its sides to create a large n-dimensional tensor. In each dimension, if the small tensor was of size s_i in that dimension, then the large tensor will be of size $2s_i - 1$ in that dimension. The zero elements are added in such a way that the last nonzero element in the original tensor becomes the center element in the new tensor.

For example,

$$P_Z \left(\begin{array}{ccccc} 1&0&1&1&0 \\ 0&1&0&0&0 \\ 0&0&0&0&0 \end{array} \right) = \begin{array}{ccccccccc} 0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&1&1&0&0 \\ 0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \end{array} \tag{1.34}$$

In this 2D example, the small matrix M is of size 3×5, and $P_Z(M)$ is of size $(2\cdot3-1)\times(2\cdot5-1) = 5\times9$. Note that this M happens to have 4 nonzero elements, arranged in a sort of V shape. If the elements of M are read in row major order (i.e. left to right across the top row, then left to right on the second row, etc.), then the last nonzero element to be read is the bottom of the V. The P_Z function pads with zeros in such a way as to yield a large matrix of the correct size, with that last nonzero element in the exact center of the large matrix.

Definition 1.18. For a given tensor size $s_1 \times \cdots \times s_n$, the set $\mathcal{T}$ is defined to be the following set of functions

$$\mathcal{T}(s_1, \ldots, s_n) = \{ f_M \mid M \in \mathcal{Z}(s_1 \times \cdots \times s_n) \} \tag{1.35}$$

where

$$f_M(x) = \begin{cases} 1 & \text{if } M = \mathbf{0} \\ H(x, P_L(M)) - H(x, P_R(M)) & \text{otherwise} \end{cases} \tag{1.36}$$

2 Theoretical results

Proofs of the theorems below are provided in a separate appendix available from the authors.

Theorem 2.1. *The cardinality of the set* $\mathcal{Z}(s_1, \ldots, s_n)$ *is* $N(s_1, \ldots, s_n)$.

Theorem 2.2. *The cardinality of the set* $\bar{\mathcal{Z}}(s_1, \ldots, s_n)$ *is* $k^{s_1 s_2 \ldots s_n} - N(s_1, \ldots, s_n)$.

Theorem 2.3. *The set of coefficient vectors for one minimal set of linear equations that define the trivial conserved functions with energy windows of size* $s = (s_1, \ldots, s_n)$ *is* $\{e(s, P_Z(A)) - e(s, P_C(P_Z(A))) \mid A \in \bar{\mathcal{Z}}(s_1, \ldots, s_n)\}$.

Theorem 2.4. *The set of coefficient vectors for one set of linear equations that defines the conserved functions with energy windows of size* $s = (s_1, \ldots, s_n)$ *for cellular automaton rule* R *with neighborhood of size* $w = (w_1, \ldots, w_n)$ *is*

$$\{e(s, P_Z(A)) - e(s, P_C(P_Z(A)) - e(s, T(R, P_Z(A))) + e(s, T(R, P_C(P_Z(A)))) \\ \mid A \in \bar{\mathcal{Z}}(s_1 + w_1 - 1, \ldots, s_n + w_n - 1)\}$$

Theorem 2.5. *The set* $\mathcal{T}(s_1, \ldots, s_n)$ *is a basis set for the space of all trivial additive conserved functions with energy windows of size* $s_1 \times \cdots \times s_n$.

Theorem 2.6. *A complement of the coefficient vectors for the equations defining the trivials for energy windows of size* $s_1\times, \ldots, \times s_n$ *is* $\{V_t(M) \mid M \in \mathcal{Z}\}$.

Note that by the definition of complements, this implies that when searching for conserved functions, without loss of generality we can constrain the energy functions to assign an energy of 0 to any window that is a zero-sided tensor. This corresponds to deleting certain columns in the matrix that defines the conserved functions. After that deletion, there will be solutions to those equations if and only if nontrivials exist. If such solutions do exist, then those solutions are guaranteed to be nontrivial conserved functions, and the union of those solutions with the trivials will span the space of conserved functions. This allows faster searches for nontrivials.

Figure 1 summarizes all the theorems of this paper, giving four examples of the M matrix for each concept. Figure 2 applies the ideas of this paper to the results of [1] and [3], expressing the basis functions as a linear sum of the matching H-functions of Definition 1.10.

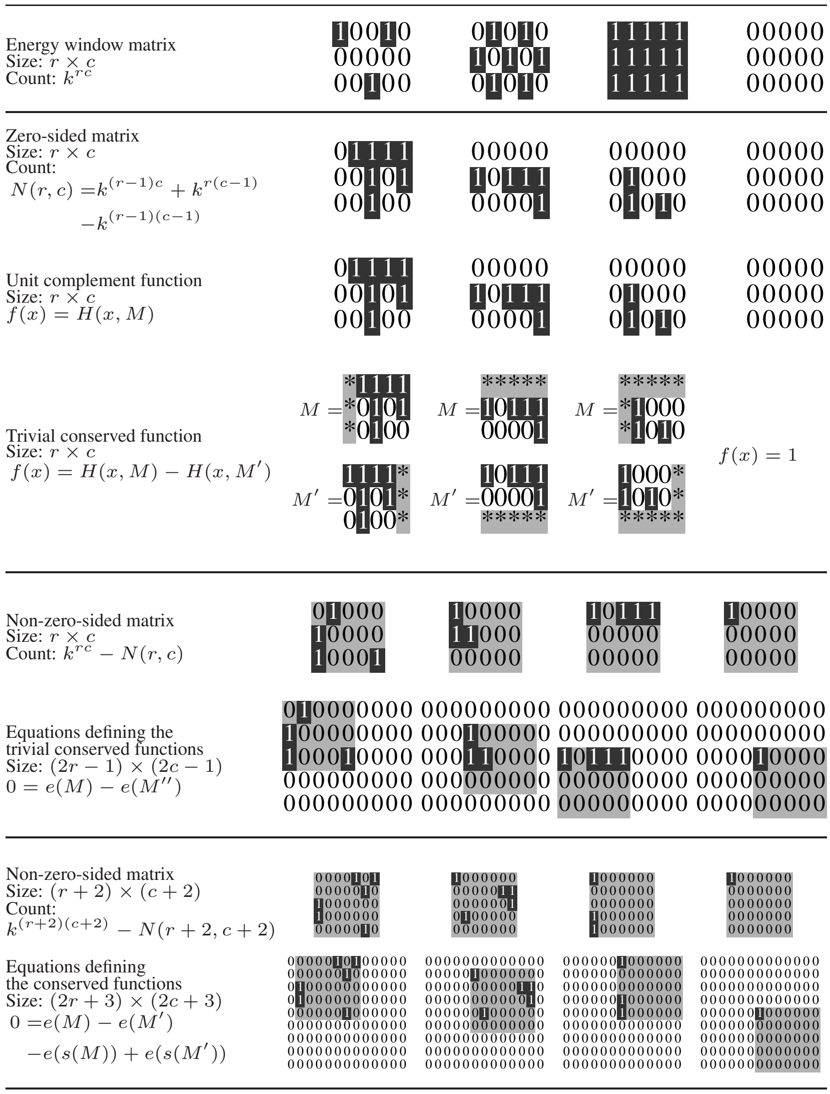

Fig. 1: Summary of the main theoretical results of this paper, with four examples of each concept. The proofs are for arbitrary dimensions, neighborhood sizes, and number of colors, but the figure shows only 2D examples, for a CA with a 3×3 neighborhood, and $k = 2$ colors. In each case, M' is M with the central bit set to 0. For the equations, the large matrix is formed by padding the small matrix with zeros such that the last 1 bit ends up in the center of the large matrix (where "last" is the last 1 found when traversing the elements in row major order). In each of the four sections, the listed concepts all have the same count. For example, the number of zero-sided matrices of a given size equals the number of unit complement functions, which equals the number of trivials.

CA	Basis
170 184 204	$f(x) = H(x, 1)$
12 14 15 34 35 42 43 51 140 142	$f(x) = H(x, 10)$
200	$f(x) = H(x, 11)$
2 3 172	$f(x) = H(x, 100)$
4	$f(x) = H(x, 010)$
10	$f(x) = H(x, 1{*}0)$
56 76	$f(x) = H(x, {*}10)$ $+H(x, 110)$
138	$f(x) = H(x, 100)$ $+H(x, 11{*})$
1	$f(x) = H(x, 1000)$
11 27	$f(x) = H(x, 100{*})$ $+H(x, 1011)$
29	$f(x) = H(x, {*}100)$ $+H(x, 1100)$ $+H(x, 101{*})$
38 46	$f(x) = H(x, 100{*})$ $+H(x, 1101)$
72	$f(x) = H(x, 0110)$
5	$f_1(x) = H(x, 010{*}1)$ $+2H(x, 10{*}0{*})$ $-H(x, 1010{*})$ $+H(x, 10010)$ $f_2(x) = H(x, 1{*}0{*}0)$
19	$f(x) = H(x, 10100)$ $+H(x, 1100{*})$
24	$f(x) = H(x, 1000{*})$ $+H(x, 10010)$ $+H(x, 1101{*})$ $+H(x, 11100)$
36	$f(x) = H(x, 00100)$ $+H(x, 11011)$
108	$f(x) = H(x, {*}0100)$ $+H(x, 10100)$ $+H(x, 10111)$ $+2H(x, 1100{*})$ $+H(x, 11101)$
132	$f(x) = H(x, 01010)$
23	$f(x) = H(x, 001100)$ $+H(x, 110011)$
50 178	$f(x) = H(x, 011001)$ $+H(x, 100110)$
77	$f_1(x) = H(x, 011001)$ $f_2(x) = H(x, 100110)$
232	$f_1(x) = H(x, 011000)$ $-H(x, 101100)$ $+H(x, 110010)$ $-H(x, 111001)$ $f_2(x) = H(x, 110011)$
44	$f(x) = H(x, 1000{*}{*}{*}{*})$ $+H(x, 101001{*}{*})$ $+H(x, 10101101)$ $+H(x, 1011101{*})$ $+H(x, 11001{*}{*}{*})$ $+H(x, 111101{*}{*})$
73	$f(x) = H(x, 01100110)$
7	$f(x) = H(x, 00011000{*})$ $+H(x, 000110010)$ $+H(x, {*}01100011)$ $+H(x, {*}11001011)$ $+H(x, 11100011{*})$
33	$f(x) = H(x, 0001000100{*}{*})$ $-H(x, 00010001001{*})$ $+2H(x, 000100011{*}{*}{*})$ $-H(x, 0001000111{*}{*})$ $-H(x, 00010001100{*})$ $-H(x, 000100011010)$ $+H(x, 000101100011)$ $+H(x, 000101110111)$ $+H(x, 0001101001{*}{*})$ $+2H(x, {*}00110110000)$ $-H(x, 100110110000)$ $+H(x, 00011011001{*})$ $+H(x, 00011011100{*})$ $+H(x, 000110111010)$ $+H(x, 0001101111{*}{*})$ $+H(x, 001000110100)$ $+H(x, 0{*}1011000100)$ $+H(x, 0{*}1011101001)$ $+H(x, 0{*}1011101100)$ $+3H(x, 00110110001{*})$ $-H(x, 001101100010)$ $+H(x, 001101110100)$ $+3H(x, 00110111011{*})$ $-H(x, 001101110110)$ $+H(x, 10010001000{*})$ $+H(x, 100100011011)$ $+2H(x, 1001011000{*}{*})$ $-H(x, 10010110000{*})$ $-H(x, 100101100010)$ $+H(x, 100101110100)$ $+2H(x, 10010111011{*})$ $-H(x, 100101110110)$ $-H(x, 100110110001)$ $-H(x, 100110111011)$ $+H(x, 101101100011)$ $+H(x, 101101110111)$ $+H(x, 110001000{*}{*}{*})$ $+H(x, 11000110100{*})$ $+H(x, 1100011011{*}{*})$ $+2H(x, 111011000{*}{*}{*})$ $-H(x, 1110110000{*}{*})$ $-H(x, 111011000101)$ $+2H(x, 11101110100{*})$ $-H(x, 111011101000)$ $+2H(x, 1110111011{*}{*})$ $-H(x, 111011101101)$
164	$f(x) = H(x, 00100100100{*}{*})$ $+H(x, 0110110110110)$
94	$f(x) = H(x, 01110010111101)$ $+H(x, 1010010111101{*})$ $+H(x, 1011110100101{*})$ $+H(x, 10111101001110)$
104	$f_1(x) = H(x, 00101011010100)$ $+H(x, 001011110100{*}{*})$ $+H(x, 00110011000{*}{*}{*})$ $+H(x, 001100110010{*}{*})$ $+H(x, 00110011001101)$ $+H(x, 0011001100111{*})$ $f_2(x) = H(x, 00110011001100)$

Fig. 2: 1D Basis functions. For each CA, this lists the lowest-order nontrivial conserved functions. The given functions, combined with the trivials, constitute a basis set for the space of all conserved functions for that CA. The table contains all 88 of the non-isomorphic primitive CAs, except those that are known to have no nontrivials (0,8,32,40,128,136,160,168,60,30,90,154), and those that have no known nontrivials and have been proved to have none at least up to and including size 16 energy windows (106,150,6,9,13,18,22,25,26,28,37,41,45,54,57,58,62,74,78,105,110, 122,126,130,134,146,152,156,162).

3 Computational results

The challenge in identifying cellular automata with a nontrivial additive energy conservation function (hereafter referred to as a ”nontrivial”) is the enumeration of the trivial functions and their elimination from the solution space. The actual calculation of the nontrivials can then be reduced to the calculation of the null space of the system of corresponding state space equations. Thus the theorems and definitions of the previous section may be used as the basis for computational identification of cellular automata with nontrivials of various orders. Computationally, this proceeds as follows:

1) Choose a CA and energy window size (s_1, s_2).

2) For all possible matrices M given by Theorem 2.4, generate the corresponding state space equations.

3) To remove the trivials from the solution space, delete the columns associated with the zero-sided tensors as determined by Theorem 2.6. This has the additional benefit of significantly reducing the size of the energy vectors and, therefore, the state space matrix as a whole.

4) Determine the rank of the resulting matrix. If it is full rank, the system of equations has no solution, and therefore no nontrivial exists for the given CA and window size. If the matrix is rank-deficient, a nontrivial exists. It is completely characterized by the basis vectors that are the columns of the matrix's null space.

In [1], we gave a complete taxonomy of binary nontrivials for 1D cellular automata up for energy windows up to size 16. Using the definitions and theorems previously presented, we now extended these results to binary 2D automata, for energy windows up to size 9.

There are a total of k^{k^9} k-colored 2D cellular automata (ignoring isomorphic entries). This number is so large that any investigation other than a random sampling is effectively impossible. Accordingly, drawing substantive conclusions about unrestricted 2D cellular automata seems to the authors extraordinarily difficult. To reduce the scope of the problem and make a more complete investigation possible, we consider only *outer totalistic* CAs: Those for which the next state of the cell is a function only of the total number of colors of a given type in the region surrounding the cell and the cell itself. For binary CAs, this means that only the total number of 1's in a cell's neighborhood (including its own value) must to be calculated to determine the cell's next state. Conway's Game of Life is a cellular automaton of this type.

Restricting the search space to outer totalistic automata significantly reduces the size of the problem. For a 2D CA, the neighborhood is of size 9, and therefore the total number of occupied cells in a cell's neighborhood ranges from 0 through 8. For binary automata, one of four outcomes are possible: (S)ame, (B)irth, (D)eath, and (F)lip (Flip changes 0 to 1 and vice versa). Thus any outer totalistic CA can be represented as a character string of the form S,B,D,F. Using this notation, if we count the neighbors from 0 to 8 from left to right, Conway's Game of Life would be written as ”DDSBDDDDD”. We refer to this description at the CA's *rule vector*. Note that the use of symbols S and F permits the incorporation of the central state into the transition rule.

It is known that renumbering the colors of a CA in reverse order and changing the outcomes correspondingly produces an CA identical to the original, up to isomorphism. Using the proposed notation, this corresponds to reversing the order of the letters, swapping S with F, and swapping B with D. The rule vector of every CA can be manipulated in this way to produce a unique and distinct isomorph, so the total number of unique totalistic binary CAs is $4^9/2 = 2^{17}$. This is considerably smaller than the non-totalistic case.

The definitions and theorems in this paper give the dimensions of the matrices to be analyzed as a function of the energy window (independent of the CA being analyzed). We show the matrix sizes for some 2D examples in Table 1.

Energy window height (s_1)	Energy window width (s_2)	$\lceil log_2\, rows \rceil$	$\lceil log_2\, cols \rceil$
1	2	16	1
1	3	19	2
1	4	23	3
2	2	20	4
1	5	26	4
1	6	29	5
2	3	25	6
1	7	32	6
1	8	35	7
2	4	29	8
1	9	39	8
3	3	30	9

Tab. 1: State matrix sizes for various energy windows

Column three shows the ceiling of the log base 2 of the maximum number of energy vectors needed to determine the existence of a nontrivial. Column four shows the number of entries in each vector. This is given by the total number of possible energy function values ($2^{s_1 s_2}$) minus the number of zero-sided tensors given by Definition 1.15.

Because these matrices have far more rows than columns, we expect almost all of them to be full rank, and therefore few nontrivial conservation functions should exist over the range of cellular automata. Since full rank can be determined very quickly while rank-deficiency cannot be known until all the possible state space vectors given by Theorem 2.3 have been examined for linear independence, it would be inefficient to build the full state space matrix for each CA and then calculate its rank. Instead, we sift the sands of cellular automata through a three-stage computational sieve.

The first stage uses a "quick and dirty" algorithm to discard automata with no nontrivials. This eliminates over 99% of the candidates. The second stage takes automata that have passed the first stage and performs a little more work to try and drive the set of state space matrices to full rank. This eliminates about another 90% of the candidates it analyzes. The third stage operates only on automata that have passed the first two stages, performing exact arithmetic using all the optimizations of Theorem 2.3 to determine whether or not a given CA has a nontrivial conservation function. If it does, its basis is calculated and reported. Each stage is implemented in MATLAB.

In stage I, we compute the energy vector of Definition 1.14 for one tensor at a time, attempting to add it to an existing energy vector set via Gaussian elimination to ensure that the rows in the state space matrix at any time are always linearly independent. Before such addition, however, we delete the columns corresponding to the zero-sided tensors for the indicated energy window. The total number of deleted columns is given by Definition 1.15. None of the optimizations discussed in the proof of Theorem 2.3 are performed at this stage. Instead, universe states are generated randomly, the energy vectors of their corresponding tensors are calculated, and Gaussian elimination is performed on each vector relative to those energy vectors already admitted into the state space matrix. When the number of linearly independent energy vectors is equal to the number of columns (the number of possible energy function values minus the number of zero-sided tensors), full rank has been achieved, and the CA/energy window pair under test is known not to correspond to a nontrivial conservation function.

Since states are generated randomly in this stage, as opposed to exhaustive enumeration of the appropriate tensors as given by Theorem 2.3, the number of states N to try before giving up on the possibility of reaching full rank is a user-definable parameter. Empirically, we have found that setting N at 32x the maximum rank of the matrix gives a good tradeoff between quick computation on the one hand and admitting too many false positives on the other.

During this stage, all arithmetic is performed modulo a small prime, to eliminate the possibility of roundoff error or overflow. If full rank is reached, the matrix would be full rank in exact arithmetic as well, so the answer is correct. If full rank is not reached within the indicated time window, the matrix may or may not be rank-deficient, so the CA is marked as a candidate for stage II computation.

In stage II, candidate CA/window pairs that pass through the first stage are subject to repeated random state generation with a larger value of N for multiple attempts. No other optimizations are performed at this time. If no full rank matrix is produced (i.e. no linearly independent energy vector set of the cardinality given by Definition 1.14 is found), the pair is marked for analysis by stage III.

Stage III computation employs on-the-fly Gaussian elimination for one-at-a-time energy vector generation, similar to the first two stages, but using double precision arithmetic and enumerating the state space exactly as described in the proof of Theorem 2.3. To keep the computations from overflowing, vectors are reduced modulo the GCD of all their nonzero entries during this process, which means this stage is the most computationally intensive. If Gaussian elimination on the entire set of energy vectors does not produce a linearly independent set of Definition 1.14 cardinality, then constructed state space matrix has a null space. That null space is calculated, and reported as the basis for all nontrivial conservation functions for that particular CA/window combination.

To guard against the possibility of numerical error, the largest value observed during stage III calculation is tracked and reported, to ensure that any possibility of overflow or loss of precision will be detected. For all calculations reported here, this maximum value has always been well below that which could induce error in double precision arithmetic. So we are confident our results are correct. Nonetheless, as an added safety check, we have implemented code which accepts as input a CA, an energy window, and a stage III basis set reported as characterizing a nontrivial. It tests each vector in the basis set over large numbers of randomly selected states by evaluating the energy function through brute force dot product calculation. In all cases, the resulting functions reported by stage III were conserved.

Table 2 shows the results of our computations for all outer totalistic binary 2D cellular automata up to isomorphism, for all energy windows up to order 9. It extends [1] to give a complete taxonomy of conservation functions for all automata of this type. Figures 3 and 4 are similar to Figure 2, extended to two dimensions. Figure 5 summarizes our current knowledge of 1D conservation functions.

The first three columns of Table 2 are all different ways of identifying the same automaton. The first column is the decimal integer represented by a CA's rule vector, obtained by treating the symbols S,B,D,F as the integers 0,1,2,3 respectively, and viewing the rule vector as a number in base 4 with the most

CA#	Rule	rule vec (num neighbors)									min NCF	basis size	comments
		0	1	2	3	4	5	6	7	8			
0	S0123456789	S	S	S	S	S	S	S	S	S	1x1	n/a	identity, conserves all
2	S12345678	D	S	S	S	S	S	S	S	S	1x2	1	conserves [11] pairs
8	S02345678	S	D	S	S	S	S	S	S	S	2x2	5	conserves 2x2 patterns with ≥ 3 1's
10	S2345678	D	D	S	S	S	S	S	S	S	2x2	5	identical to 8
21	B012/S012345678	B	B	B	S	S	S	S	S	S	3x3	1	
32	S01345678	S	S	D	S	S	S	S	S	S	2x2	1	conserves 2x2 pattern with all 1's
34	S1345678	D	S	D	S	S	S	S	S	S	2x2	1	identical to 32
40	S0345678	S	D	D	S	S	S	S	S	S	2x2	1	identical to 32
42	S345678	D	D	D	S	S	S	S	S	S	2x2	1	identical to 32
16386	B7/S12345678	D	S	S	S	S	S	S	B	S	2x2	4	
16387	B07/S12345678	F	S	S	S	S	S	S	B	S	3x3	11	
21845	B01234567/S8	B	B	B	B	B	B	B	B	S	3x3	1	conserves ring of 1's around a 0
65532	B1234567/S8	S	F	F	F	F	F	F	F	S	2x3	1	
65533	B01234567/S08	B	F	F	F	F	F	F	F	S	2x3	1	identical to 65532
65534	B1234567/S8	D	F	F	F	F	F	F	F	S	2x3	1	identical to 65532
65535	B01234567/S8	F	F	F	F	F	F	F	F	S	2x3	1	identical to 65532
65537	B08/S012345678	B	S	S	S	S	S	S	S	B	2x3	7	
65538	B8/S12345678	D	S	S	S	S	S	S	S	B	2x2	8	
65539	B08/S12345678	F	S	S	S	S	S	S	S	B	2x3	7	identical to 65537
65541	B018/S012345678	B	B	S	S	S	S	S	S	B	3x3	1	conserves [001 011 010]
65545	B08/S234567	B	D	S	S	S	S	S	S	B	2x3	1	conserves the difference between [101 111] and [111 101]
65546	B8/S234567	D	D	S	S	S	S	S	S	B	2x2	4	conserves 2x2 patterns with ≥ 3 1's
65547	B08/S2345678	F	D	S	S	S	S	S	S	B	2x3	1	identical to 65545
65549	B018/S02345678	B	F	S	S	S	S	S	S	B	3x3	1	identical to 65541
81921	B078/S012345678	B	S	S	S	S	S	S	B	B	2x3	1	
81923	B078/S12345678	F	S	S	S	S	S	S	B	B	2x3	1	identical to 81921
131069	B012345678/S08	B	F	F	F	F	F	F	F	B	2x3	1	identical to 65532
131070	B12345678/S8	D	F	F	F	F	F	F	F	B	2x3	1	identical to 65532
131071	B012345678/S8	F	F	F	F	F	F	F	F	B	2x3	1	identical to 65532
131073	B0/S01234567	B	S	S	S	S	S	S	S	D	2x3	2	
131075	B0/S1234567	F	S	S	S	S	S	S	S	D	2x3	2	identical to 131073
131077	B01/S01234567	B	B	S	S	S	S	S	S	D	3x3	1	identical to 65541
131081	B0/S0234567	B	D	S	S	S	S	S	S	D	3x3	9	
131083	B0/S234567	F	D	S	S	S	S	S	S	D	3x3	9	identical to 131081
131085	B01/S234567	B	F	S	S	S	S	S	S	D	3x3	1	identical to 65541
147459	B07/S1234567	F	S	S	S	S	S	S	B	D	3x3	9	
163483	B0/S123456	F	S	S	S	S	S	S	D	D	3x3	1	conserves [011 100 101]
180227	B07/S123456	F	S	S	S	S	S	S	F	D	3x3	1	identical to 163843
196605	B01234567/S0	B	F	F	F	F	F	F	F	D	2x2	4	
196607	B01234567	F	F	F	F	F	F	F	F	D	2x2	4	Identical to 196605
196611	B08/S1234567	F	S	S	S	S	S	S	S	F	2x3	2	Identical to 131073
196619	B08/S234567	F	D	S	S	S	S	S	S	F	3x3	9	Identical to 131081
262143	B012345678	F	F	F	F	F	F	F	F	F	1x2	1	Conserves [10] pairs

Tab. 2: Conservation functions of order ≤ 9 for 2D CA's

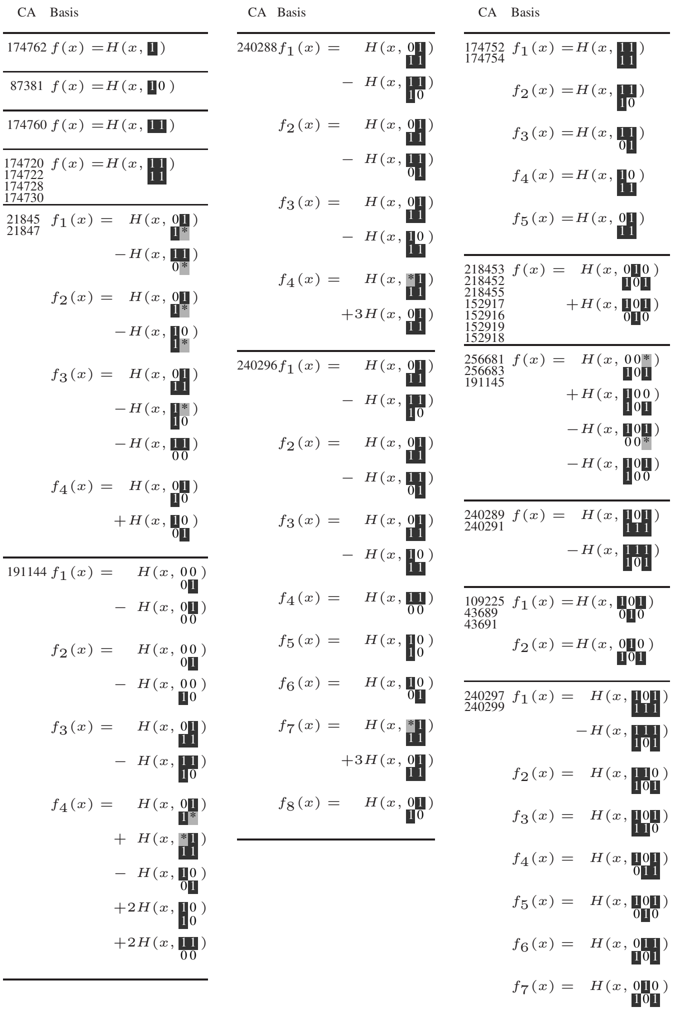

Fig. 3: 2D Basis functions. For each CA, this lists the lowest-order nontrivial conserved functions. The given functions, combined with the trivials, constitute a basis set for the space of all conserved functions for that CA. The table contains all of the non-isomorphic, 2-color, 3×3 neighborhood, outer totalistic CAs that have nontrivials of size 2×3 or smaller (the 3×3 nontrivials are shown in Figure 4).

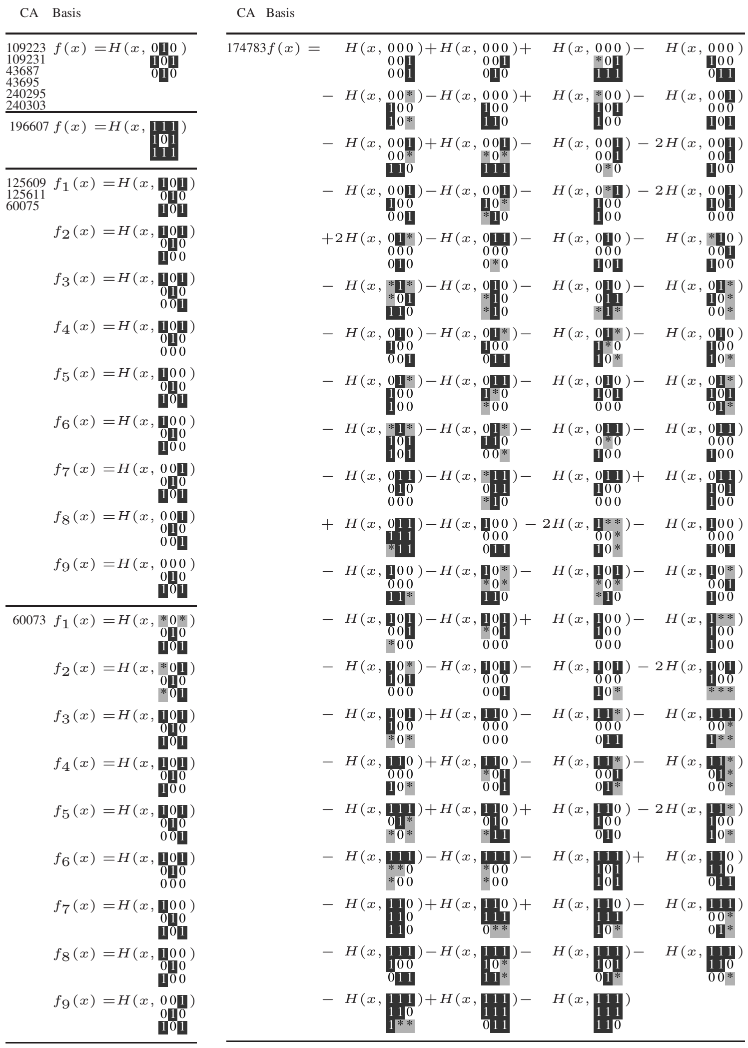

Fig. 4: 2D Basis functions (continued). These are the 3×3 nontrivials, continued from figure 3.

Order	CA(isomorphs)	Rule	11110000 11001100 10101010	Order	CA(isomorphs)	Rule	11110000 11001100 10101010
∞	0(255)	0	00000000	2	12(68,207,221)	Xy	00001100
∞	8(64,239,253)	Xyz	00001000	2	14(84,143,213)	X+XYZ	00001110
∞	30(86,135,149)	X+YZ	00011110	2	15(85)	X	00001111
∞	32(251)	xYz	00100000	2	34(48,187,243)	Yz	00100010
∞	40(96,235,249)	xz+yz	00101000	2	35(49,59,115)	xYZ+Y	00100011
∞	60(102,153,195)	x+y	00111100	2	42(112,171,241)	xyz+z	00101010
∞	90(165)	x+z	01011010	2	43(113)	xY+Xz+YZ	00101011
∞	106(120,169,225)	xy+z	01101010	2	51	Y	00110011
∞	128(254)	xyz	10000000	2	140(196,206,220)	xyZ+y	10001100
∞	136(192,238,252)	yz	10001000	2	142(212)	xy+Xz+yZ	10001110
∞	150	x+y+z	10010110	2	200(236)	XyZ+y	11001000
∞	154(166,180,210)	xY+z	10011010	3	2(16,191,247)	XYz	00000010
∞	160(250)	xz	10100000	3	3(17,63,119)	XY	00000011
∞	168(224,234,248)	XYz+z	10101000	3	4(223)	XyZ	00000100
>16	6(20,159,215)	Xy+Xz	00000110	3	10(80,175,245)	Xz	00001010
>16	9(65,111,125)	Xy+XZ	00001001	3	56(98,185,227)	xY+Xyz	00111000
>16	13(69,79,93)	X+XYz	00001101	3	76(205)	xyz+y	01001100
>16	18(183)	xY+Yz	00010010	3	138(174,208,244)	xYz+z	10001010
>16	22(151)	X+Xyz+YZ	00010110	3	172(202,216,228)	Xy+xz	10101100
>16	25(61,67,103)	Xyz+YZ	00011001	4	1(127)	XYZ	00000001
>16	26(82,167,181)	xYZ+Xz	00011010	4	11(47,81,117)	X+XyZ	00001011
>16	28(70,157,199)	Xy+xYZ	00011100	4	27(39,53,83)	Xz+YZ	00011011
>16	37(91)	xYz+XZ	00100101	4	29(71)	Xy+YZ	00011101
>16	41(97,107,121)	X+XyZ+Yz	00101001	4	38(52,155,211)	XyZ+Yz	00100110
>16	45(75,89,101)	X+Yz	00101101	4	46(116,139,209)	Xy+Yz	00101110
>16	54(147)	XZ+Y	00110110	4	72(237)	xy+yz	01001000
>16	57(99)	Xz+Y	00111001	5	5(95)	XZ	00000101
>16	58(114,163,177)	xY+Xz	00111010	5	19(55)	xYz+Y	00010011
>16	62(118,131,145)	x+XYz+y	00111110	5	24(66,189,231)	xYZ+Xyz	00011000
>16	74(88,173,229)	xyZ+Xz	01001010	5	36(219)	xYz+XyZ	00100100
>16	78(92,141,197)	Xz+yZ	01001110	5	108(201)	xz+y	01101100
>16	105	x+y+Z	01101001	5	132(222)	xy+yZ	10000100
>16	110(124,137,193)	Xyz+y+z	01101110	6	23	xY+XZ+Yz	00010111
>16	122(161)	x+xYz+z	01111010	6	50(179)	XYZ+Y	00110010
>16	126(129)	xY+Xz+yZ	01111110	6	77	xy+XZ+yz	01001101
>16	130(144,190,246)	xz+Yz	10000010	6	178	xy+xZ+Yz	10110010
>16	134(148,158,214)	X+XYZ+yz	10000110	6	232	xy+xz+yz	11101000
>16	146(182)	x+xyZ+Yz	10010010	8	44(100,203,217)	Xy+xYz	00101100
>16	152(188,194,230)	xYZ+yz	10011000	8	73(109)	X+XYz+yZ	01001001
>16	156(198)	xZ+y	10011100	9	7(21,31,87)	X+Xyz	00000111
>16	162(176,186,242)	Xyz+z	10100010	12	33(123)	xY+YZ	00100001
1	170(240)	z	10101010	13	164(218)	XyZ+xz	10100100
1	184(226)	xY+yz	10111000	14	94(133)	x+XyZ+z	01011110
1	204	y	11001100	14	104(233)	x+xYZ+yz	01101000

Fig. 5: Summary of results for the primitive CAs (1D, 2-color, neighborhood of 3 cells). In each half of the table, the first column gives the energy window size for the smallest nontrivial. A value of ∞ indicates that it is known no nontrivial can exist. A value of >16 indicates that no nontrivial exists with energy window of size 16 or below. The next column has the CA name, and the names of the isomorphic CAs. The next is the formula for the successor function, where cells have state 0 or 1, three consecutive cells are called x, y, z (with capitalized inverses, so X=1-x etc.), and the formula modulo 2 gives the new state for y. Finally, the successor function is shown graphically, giving the new state as a function of the state in that cell and its immediate neighbors (shown at the top of the column).

significant digit on the right. The second column shows the CA rule using the notation in [8]. Column three is the CA's rule vector.

Columns four through six describe the nontrivial conservation function found. Column four shows the dimensions of the energy window at which the first nontrivial was discovered. Column five shows the number of basis vectors in the null space of the CA's state matrix for an energy window of the indicated size. Column six contains, where appropriate, comments describing the conservation function. A blank entry in this column means that either no simple description exists or that describing the pattern would be too complex to fit within the indicated space.

Symmetry arguments will show that analogous conservation functions for any $m \times n$ window can also be found for one that is $n \times m$. Thus the only energy windows examined were those that were at least as wide as they were tall.

4 Analysis

Some patterns are clearly visible in Table 2, Figure 3, Figure 4 and Figure 5. For all CA's for which non-trivial conservation functions exist, there is a great deal of homogeneity in the middle range of neighbor counts. For example, any given CA in the table has the same transition rules for neighbor counts 3-6, and most have identical transition rules for neighbor counts 2-7. We conjecture this is combinatorically driven. That is, for the middle range of neighbor counts, there are so many different ways to distribute a fixed number of neighbors among eight cells that a low-order conservation function cannot incorporate them all. By contrast, there is only one way to arrange zero or eight neighbors around a cell, eight ways to arrange one or seven, and so forth. Near the minimum and maximum of the neighbor count range, the number of possible configurations is sufficiently small that a low-order conservation function is more likely to emerge.

We also note that all CA's with rule vectors of the form xFFFFFFFx, xSSSSSSSB, and xDSSSSSSB have nontrivial conservation functions. All CA's of the form xSSSSSSSx have a nontrivial as well, unless exactly one of the x's is 'S'.

Finally, our results show that all known nontrivials correspond to energy windows for which the width and the height differ by no more than one. Whether this holds true for all nontrivials remains an open question.

5 The Game of Life

Because of the special significance of Conway's Game of Life (CA #174666, rule B3/S23, rule vector DDSBDDDDD), we have examined it for nontrivial energy conservation functions up to order 13. None have been found.

6 Conclusions and Future Work

Table 2 and Figures 3 through 5 represent a complete taxonomy of all known nontrivial conservation functions for 1- and 2-dimensional binary cellular automata up to isomorphism. We have discussed some of the patterns we have observed.

[1] introduced the notion of core nontrivials, recognizing that cellular automata could exhibit different nontrivials of higher orders that are not simple extensions of lower ones. We have yet to apply this idea to the automata shown here. Thus the functions we report are only the first core nontrivials found. The existence of multiple cores for 2D binary cellular automata remains an open question. Detecting such cores requires only well-understood modifications to our existing code, and is on our list of future enhancements.

Number-conserving 1D cellular automata [2] are automata with transition rules that conserve the sum of the number of states in a neighborhood. A number-conserving function is one kind of energy conservation function defined in Definition 1.8, where the function is simply the sum of all terms in the window. Our work therefore includes number-conservation as a special case. The theory described here applies to all cellular automata with finite states and arbitrary dimensionality. The results for 2D automata are all new.

Continuing improvements in computing power and further refinements of our codes should enable us to identify nontrivials at increasingly higher orders. The existence of nontrivialss for $m \times n$ energy windows with $|m - n| > 1$ remains an open question. Higher dimensional CAs, non-totalistic CAs, and k-colored CAs could also be explored.

As yet, an elegant, unifying description of cellular automata relating their decision rules and a given energy window to a nontrivial conservation function remains elusive. While the general problem is undecidable, we have mapped out the space for lower orders and binary outer totalistic CAs well enough to suggest some ideas for a more elegant classification scheme than the present ad hoc one we are currently forced to adopt. Such a scheme may in fact exist, or it may remain forever elusive, an fundamentally complex property inherent in the nature of computational automata. We hope further work may yet resolve this question.

7 Errata and Acknowledgments

Readers unfamiliar with automata conservation functions may wish to review [1]. In the course of preparing this paper, we noticed errors in the first three tables of our previous results. For the sake of completeness, we present the necessary corrections to [1] here:

TABLE 1: Replace 98 with 94, replace 40 with 46
TABLE 2: Replace 136 with 200
TABLE 3: Replace 136 with 200, replace 248 with 232

The authors are grateful for the support of the Air Force Academy Center for Cyberspace Research, and to the reviewers for their helpful comments.

References

[1] L. Baird and B. Fagin, *Conservation functions for 1-d automata: Efficient algorithms, new results, and a partial taxonomy*, Journal of Cellular Automata **3** (2008), no. 4, 271–288.

[2] Nino Boccara and Henryk Fuks, *Number-conserving cellular automaton rules*, Fundam. Inform. **52** (2002), no. 1-3, 1–13.

[3] B. Fagin and L. Baird, *New higher-order conservation functions for 1-d cellular automata*, Proceedings of the IEEE Symposium on Artificial Life, April 1-5 2007.

[4] H. Fuks, *Remarks on the critical behavior of second order additive invariants in elementary cellular automata*, Fundamenta Informaticae **78** (2007), 329–341.

[5] T. Hattori and S Takesue, *Additive conserved quantities in discrete-time lattice dynamical systems*, Physica D **49** (1991), 295–322.

[6] L. Kotze and W.H. Steeb, *Finite dimensional integrable nonlinear dynamical systems*, pp. 333–346, World Scientific Publishing, New Jersey, 1998.

[7] M. Pivato, *Conservation laws in cellular automata*, Nonlinearity **15** (2002), 1781–1793.

[8] Wikipedia, *Life-like cellular automaton*, `http://en.wikipedia.org/wiki/Life-like_cellular_automaton`, March 2010.

[9] S. Wolfram, *A new kind of science*, Wolfram Media Inc., 2002.