Complex Systems **2** (1988) 649–661

Additive Automata On Graphs

Klaus Sutner
Stevens Institute of Technology, Hoboken, NJ 07030, USA

Abstract. We study cellular automata with additive rules on finite undirected graphs. The addition is carried out in some finite abelian monoid. We investigate the problem of deciding whether a given configuration has a predecessor. Depending on the underlying monoid this problem is solvable in polynomial time or **NP**-complete. Furthermore, we study the global reversibility of cellular graph automata based on addition modulo two. We give a linear time algorithm to decide reversibility of unicyclic graphs.

1. Introduction

In this paper we study cellular automata on graphs: the vertices of the graph are the cells and the neighborhood of a cell v consists of the vertices adjacent to it. The behavior of the automaton is determined by a local transition rule: the state of cell v at time $t+1$ depends only the states of the cells in the neighborhood of v at time t. We consider local rules of a very simple algebraic nature: states are elements of a finite abelian monoid and the next state of a cell v is the sum of the states of all neighboring cells. Any finite abelian monoid $\mathbf{F}$ thus gives rise to a class of cellular automata on graphs. We will always assume that the graphs under consideration are locally finite (i.e., the neighborhood of any vertex is finite). A graph together with a finite abelian monoid $\mathbf{F}$ is called a $\mathbf{F}$-automaton. We denote the global rule by $\rho_{\mathbf{F}}$.

One of the basic questions to pose about such dynamic systems is the reversibility problem: Can one reconstruct every configuration X from its successor $\rho_{\mathbf{F}}(X)$? Note that the rules $\rho_{\mathbf{F}}$ are in general locally irreversible, i.e., different local configurations lead to the same state in one particular cell. Globally, however, these systems may well be reversible.

In the simplest case the states are 0 and 1 and the monoid operation is addition modulo two. The only other operation that imposes a monoid-structure on $\{0,1\}$ is logical or (addition modulo 2 may also be construed as exclusive or). We denote the corresponding rules by σ and $\vee$ respectively. If every cell is included in its own neighborhood we write σ^+ and $\vee^+$ for the resulting rule. It is well known that the evolution of a one point seed configuration on a one-dimensional cellular automaton with rules σ and σ^+

leads to rather surprisingly complicated patterns with fractal dimensions $\log_2 3$ and $\log_2(1+\sqrt{5})$. Classical cellular automata with these rules were studied for example in [12], [13], and [11]. For cycles and paths see [6] and [10]. In [5] automata on trees with rule σ^+ are introduced. The reversibility of these tree automata is investigated in [1]. We develop a number of reduction techniques that allow to express the reversibility of graphs under rules σ and σ^+ in terms of the reversibility of smaller graphs. Applying these techniques to unicyclic graphs we obtain a linear time algorithm for the reversibility problem of these simple graphs.

A somewhat related problem is to determine whether a given configuration has a predecessor. This combinatorial problem is referred to as Predecessor Existence Problem (PEP) in [8]. It is shown there that PEP is solvable in polynomial time on one-dimensional finite cellular automata but becomes **NP**-complete for two-dimensional automata. For additive cellular automata on graphs we will see that PEP is solvable in polynomial time if the number of states is restricted to two, i.e., for the rules σ, σ^+, $\vee$, and $\vee^+$. The same result holds whenever $\mathbf{F}$ is an abelian group. In general, however, the problem is **NP**-complete. In fact, there is a aperiodic monoid with only three elements for which PEP is **NP**-complete. For rules the σ, σ^+, $\vee$, and $\vee^+$ we will show that a modified version of PEP where the predecessor configuration is required to have the minimal number of non-zero cells is also **NP**-hard.

This paper is organized as follows. In section 2 we give formal definitions for additive cellular automata and point out several simple properties of these systems. Section 3 deals with the existence of predecessors in $\mathbf{F}$-automata for various monoids $\mathbf{F}$. Related results are also obtained in [8]. Lastly, section 4 contains the reversibility results for the modulo 2 rules. To keep this paper reasonably short we will not introduce any graph theoretic terminology used here. The reader should consult [2] or [3]. In particular results about the characteristic polynomial of a graph — which will be used in section 4 — can be found in [7].

2. Definitions

Although the focus in this paper is on finite cellular automata the definitions given in this section also cover the infinite case. In what follows we always assume that $G = \langle V, E\rangle$ is a locally finite undirected graph without isolated points. Locally finite here means that every vertex in G is adjacent to only finitely many vertices. The vertices of G are construed as the cells of a cellular automaton. The *neighborhood* $\Gamma(v)$ of a cell v is defined as the set of all vertices u such that $\{u,v\}$ is an edge in G. We will write $\Gamma^+(v)$ for the closed neighborhood $\Gamma(v) \cup \{v\}$. Let S be the set of states and $\mathbf{F} = \langle S, \circ, 0\rangle$ an abelian monoid. A *configuration* of the graph $G = \langle V, E\rangle$ is a function $X : V \to S$. A *local configuration* at v on G is a function $Z : \Gamma(v) \to \mathbf{F}$. We let $\mathcal{C}_G$ denote the collection of all configurations. Any configuration X determines a local configuration X_v at v, for each vertex v, by $X_v(u) := X(u)$, $u \in \Gamma(v)$. A *local rule* ρ is a map from local configurations into the alphabet

S. The *additive rule* determined by $\mathbf{F}$ has the form

$$\rho_{\mathbf{F}}(X_v) := X(u_1) \circ X(u_2) \circ \ldots \circ X(u_k)$$

where $\Gamma(v) = \{u_1, \ldots, u_k\}$. Note that this is well-defined: since $\circ$ is assumed to be associative and commutative the enumeration of the vertices in the neighborhood of v plays no role. Since we allow vertices of degree 1 we need a default value for $\Gamma(v) = \{u\} : \rho_{\mathbf{F}}(X_v) := X(u)$. Similarly a local rule ρ_F^+ is defined by using the closed neighborhood $\Gamma^+(v)$ instead of $\Gamma(v)$. Now let $G = \langle V, E \rangle$ be some locally finite graph. Define the global rule $\rho_{\mathbf{F}} : \mathcal{C}_{\mathrm{G}} \to \mathcal{C}_{\mathrm{G}}$ by $\rho_{\mathbf{F}}(X)(v) := \rho_{\mathbf{F}}(X_v)$ for all v in V.

A $\mathbf{F}$-*automaton* is a pair $\langle G, \mathbf{F} \rangle$ where G is a graph and $\mathbf{F} = \langle S, \circ, 0 \rangle$ an abelian monoid: the global rule is understood to be ρ_F. Similarly a $\mathbf{F}^+$-*automaton* uses rule ρ_F^+. $Y = \rho_F(X)$ is called the successor of X (with respect to ρ_F) and conversely X is a predecessor of Y.

An interesting special case arises when the underlying abelian monoid $\mathbf{F}$ is in fact a group. In this case $\mathbf{F}$ is a direct sum of summands of the form $\mathbf{Z}/(p^e)$ where p is a prime and $e \geq 1$. Hence $\mathbf{F}$ can be thought of as a ring $\mathbf{F} = \langle S, +, *, 0, 1 \rangle$ and, correspondingly, $\mathcal{C}_{\mathrm{G}}$ as a free $\mathbf{F}$-module. Furthermore, the adjacency matrix A of G can be construed as a matrix over the ring $\mathbf{F}$. The matrix A has as usual a 1 in position i, j iff $\{i, j\}$ is an edge in G. However, the entries 0 and 1 are elements of $\mathbf{F}$ rather than integers or booleans, as is more common in graph-theory. Considering a configuration X for the moment to be a column vector over $\mathbf{F}$ it is clear that $\rho_F(X) = A \cdot X$. Hence ρ is a linear operator from $\mathcal{C}_{\mathrm{G}}$ to itself. We will call additive rules of this type *linear rules*. Let $\mathbf{B} = \langle \{0, 1\}, \vee, 0 \rangle$ denote the boolean monoid with operation or, thus $1 \vee 1 = 1$. One can expand $\mathbf{B}$ to the semiring of booleans $\langle \{0, 1\}, \vee, \wedge, 0, 1 \rangle$. Interpreting all algebraic operations over this semi-ring we get $\vee(X) = A \cdot X$. Hence rules $\vee$ as well as $\vee^+$ are linear. Lastly for $n \geq 0$ let $[n] := \{1, 2, \ldots, n\}$.

3. Existence of Predecessor Configurations

One of the basic problems in the study of the evolution of configurations is to determine whether a given configuration has a predecessor. For a number of results on the complexity of this problem on classical cellular automata see [14] for infinite automata and [8] for finite automata. Formally, define the following combinatorial problem:

Problem: **Predecessor Existence Problem (PEP)**

Instance: An additive automaton $\langle G, \mathbf{F} \rangle$ and a target configuration Y.

Question: Is there a configuration X such that $\rho_F(X) = Y$?

Here G is assumed to be a finite graph. As is pointed out in [8] PEP is naturally in **NP** for finite cellular automata: one may guess the predecessor configuration X and then verify in a polynomial number of steps that indeed

$\rho_{\mathbf{F}}(X) = Y$ where Y is the given target configuration. Polynomial here and in the sequel always refers to the size of the instance. For our applications the monoid $\mathbf{F}$ will always be fixed. Hence it is convenient to assume that the size of an instance is the number of vertices in the underlying graph G.

Whenever the underlying abelian monoid $\mathbf{F}$ is in fact a group PEP can be solved in polynomial time. To see this recall from the introduction that in this case $\rho_{\mathbf{F}}$ is a linear map, $\rho_{\mathbf{F}} : \mathcal{C}_G \to \mathcal{C}_G$. Note that configuration Y has a predecessor X iff $det(A) \cdot X = A^* \cdot Y$ where A is the adjacency matrix of G (construed as a matrix over $\mathbf{F}$) and A^* is the adjugate of A. If $\det(A)$ has an inverse in $\mathbf{F}$ the predecessor is $X = \det(A)^{-1} \cdot A^* \cdot Y$. In particular, rule $\rho_{\mathbf{F}}$ is surjective on G iff rule $\rho_{\mathbf{F}}$ is injective on G iff $\det(A)$ has a multiplicative inverse in $\mathbf{F}$. The same holds for the rule ρ_F^+.

To see that PEP is also solvable in polynomial time for the rule $\vee = \rho_B$ suppose we are given a target configuration Y on some graph G. Set $V_0 := \bigcap\{\Gamma(x) \mid Y(x) = 0\}$. Clearly Y has a predecessor under rule $\vee$ iff for all vertices x such that $Y(x) = 1$ we have $\Gamma(x) \cap V_0 = \emptyset$. The argument for $\vee^+$ is entirely the same.

We note in passing that in fact PEP is solvable in polynomial time for any associative, commutative operation $\circ$ on $\{0,1\}$, regardless of whether $\langle\{0,1\}, \circ\rangle$ is a monoid or not.

By way of contrast we will show that PEP becomes **NP**-complete for an abelian monoid of cardinality three. To this end let $\mathbf{M}_3$ denote the three-element abelian monoid generated by $\alpha : [3] \to [3]$ where $\alpha(i) := \max(i+1, 3)$. Clearly $\alpha^2 = \alpha^3$, thus $\mathbf{M}_3$ has indeed cardinality three and is aperiodic. We will write 1 for α^0 and 0 for α^2. Note that $0 \circ x = 0$, $\alpha \circ \alpha = 0$. Hence for all $x_1,\ldots,x_k$ in $\mathbf{M}_3$ we have $x_1 \circ \ldots \circ x_k \neq 0$ implies $x_i \neq 0$ for all $i = 1, \ldots, k$. Also, $x_1 \circ \ldots \circ x_k = \alpha$ implies $x_i = \alpha$ for exactly one $i \in [k]$ and $x_j = 1$ for all $j \neq i$, $j \in [k]$.

Theorem 3.1. *The Predecessor Existence Problem is* **NP**-*complete for* $\mathbf{M}_3$-*automata.*

Proof. Membership in **NP** is again obvious. **NP**-hardness can be established using an embedding of a version of 3SAT called One-in-Three SAT, see [4]. An instance of One-in-Three SAT is a boolean formula $\Phi = \varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_m$ in 3-conjunctive normal form, using variables in $X = \{x_1, \ldots, x_n\}$. The problem here is to determine whether there exists a satisfying truth assignment that makes exactly one of the literals in each clause true. Suppose clause φ_i is $z_{i,1} \vee z_{i,2} \vee z_{i,3}$ where the $z_{i,j}$ are literals over X.
Now define a graph G on vertices $x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n, u_1, \ldots, u_n, v_1, \ldots, v_m$ and $b_1, \ldots, b_m$. G has edges $\{x_i, u_i\}$, $\{\bar{x}_i, u_i\}$, $i = 1, \ldots, n$, and, for every $j = 1, \ldots, m$, vertex v_j is connected to those x_i and $\bar{x}_i$ that correspond to literals in the $j-th$ clause. The target configuration Y is defined by

$$Y(v) = \begin{cases} \alpha & \textit{if } v \in \{u_1, \ldots, u_n, v_1, \ldots, v_m\}, \\ 1 & \textit{otherwise.} \end{cases}$$

Now suppose X is a predecessor of Y on the $\mathbf{M}_3$-automaton on G. It follows from the remark preceding the theorem that $X(v) \in \{1, \alpha\}$ for all v. Similarly, since $Y(u_i) = \alpha$, exactly one of x_i and $\bar{x}_i$ must be in state α in X. Hence X can be interpreted as a truth assignment τ_X. To be more explicit, set $\tau_X(x_i) := true$ iff $X(x_i) = \alpha$. Since $Y(v_j) = \alpha$ exactly one of the literals in φ_j, $j \in [m]$, is satisfied by τ_X. Conversely, any satisfying truth assignment can be translated into a predecessor configuration. Hence One-in-Three 3SAT is polynomial time reducible to PEP for $\mathbf{M}_3$ -automata. ∎

As mentioned above, PEP is solvable in polynomial time in any $\mathbf{F}_k$-automaton. For any configuration $X : V \to \{0, \dots, k-1\}$ define the support of X to be the cells not in state 0. We will show that it is **NP**-hard to determine a predecessor with minimal support even on a $\mathbf{F}_2$-automaton. More precisely we will show that the following modified version of PEP is **NP**-complete.

Problem: **Bounded Predecessor Existence Problem (BPEP)**

Instance: An additive cellular automaton $\langle G, \mathbf{F}\rangle$, a target configuration Y and a bound β.

Question: Is there a configuration X such that $\rho_{\mathbf{F}}(X) = Y$ and the support of X has cardinality at most β?

Theorem 3.2. *The Bounded Predecessor Existence Problem is* **NP**-*complete for* $\mathbf{F}_2$-*automata. The problem remains* **NP**-*complete even if the target configuration is fixed to be* $\mathbf{1}$, $\mathbf{1}(v) := 1$ *for all* v *in* V. *The same holds for* $\mathbf{F}_2^+$-*automata.*

Proof. Membership in **NP** is obvious: one may guess a predecessor X and verify in polynomial time that indeed $\sigma(X) = Y$ and that the support of X has the desired size. To show **NP**-hardness we will embed the combinatorial problem 3SAT, see [4]. An instance of 3SAT is a boolean formula in 3-conjunctive normal form and one has to decide whether the formula is satisfiable. So let $\Phi = \varphi_1 \wedge \varphi_2 \wedge \dots \wedge \varphi_m$ be a boolean formula in 3-conjunctive normal form using variables in $X = \{x_1, \dots, x_n\}$. Suppose again that clause φ_i is $z_{i,1} \vee z_{i,2} \vee z_{i,3}$ where the $z_{i,j}$ are literals over X. The argument is very similar for rules σ and σ^+, we will therefore only show how to construct a $\mathbf{F}_2^+$-automaton. The underlying graph G has vertices $x_1, \dots, x_n, \bar{x}_1, \dots, \bar{x}_n$ and there is an edge between x_i and $\bar{x}_i$. These vertices are the truth setting vertices. Furthermore, for each of the clauses φ_j in Φ, $j \in [m]$, G contains a component H_j that is connected to some of the truth setting vertices but to no other components. The generic component H contains three a-vertices, seven b-vertices and is shown in figure 1.

In addition to the edges shown there is an edge between any two of the b-vertices; so the subgraph induced by these vertices is a K_7. The connecting edges between H_j and the truth setting vertices are defined as follows: there is an edge between $a_{j,\nu}$ and $z_{j,\nu}$, $j \in [m]$ and $\nu \in [3]$. The target configuration

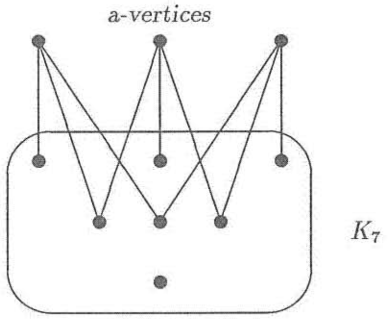

Figure 1: A component of the graph used to imbed 3-SAT.

is fixed to be $\mathbf{1}$, $\mathbf{1}(v) = 1$ for all v in V. Lastly define the bound to be $\beta := n + m$.

Let us identify a configuration $X : V \to \{0,1\}$ with the set of vertices $\{v \in V \mid X(v) = 1\}$. Observe that for any predecessor X of $\mathbf{1}$ under rule σ^+ and any vertex x we have $|X \cap \Gamma^+(x)|$ is odd. Hence X contains either x_i or $\bar{x}_i$, $i \in [n]$, but not both. X may therefore be interpreted as a truth-assignment α_X for Φ. To be more explicit, set $\alpha_X(x_i)$:=true iff x_i is in X. Since $b_{i,7}$ is adjacent only to vertices $b_{j,1}, \ldots, b_{j,6}$ an odd number of these vertices must belong to X in each component H_j, $j \in [m]$. Now suppose that X has in addition support of cardinality at most $\beta = n + m$. Since $|X \cap \{x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n\}| = n$ it follows that exactly one of the b-vertices in H_j is in X for all $j \in [m]$. But then, by the definition of H_j, at least one of the a-vertices in H_j must be adjacent to a truth-setting vertex z in X. Thus $\alpha_X(\varphi_j) =$ true for all $j \in [m]$ and α_X satisfies Φ. Conversely, if Φ is satisfiable, it is easy to define a predecessor of $\mathbf{1}$ with support of cardinality $\beta = n+m$. Thus Φ is satisfiable iff G has a predecessor of $\mathbf{1}$ with support of cardinality $n + m$ and we are through. ∎

We note that for any locally finite graph G the "all-ones" configuration $\mathbf{1}$ has a predecessor under rule σ^+, see [9]. Finding minimal predecessors therefore is **NP**-hard on a σ^+-automaton even if the target configuration is fixed and known to have a predecessor.

Theorem 3.3. *The Bounded Predecessor Existence Problem is* **NP**-*complete for* B-*automata as well as for* $\mathbf{B}^+$-*automata.*

Proof. Membership in **NP** is again obvious. As in the last argument let us identify a configuration $X : V \to \{0,1\}$ with the set of vertices $\{v \in V \mid X(v) = 1\}$. Note that X is a dominating set for G iff $\vee^+(X) = \mathbf{1}$. Hence it is **NP**-complete to decide whether there is a configuration X whose support has cardinality at most β such that $\vee^+(X) = \mathbf{1}$, see [4]. The argument for rule $\vee$ again uses an imbedding of 3SAT. Assume the notation from the last theorem and let G be the graph of theorem 3.1. The target configuration is $\mathbf{1}$ and the bound is n, the number of variables in Φ. Now suppose X is a predecessor of $\mathbf{1}$ on G under rule $\vee$ of cardinality at most n. We must have $X \cap \Gamma(u_i) \neq \emptyset$ for all $i \in [n]$. Hence, as in theorem 3.1, X contains either x_i or $\bar{x}_i$ but not both and can therefore be interpreted as a truth assignment which clearly satisfies Φ. Conversely, any satisfying truth assignment is readily converted into a predecessor of $\mathbf{1}$ of cardinality n and we are through. ∎

4. Reversibility and Reductions

We now turn to the question whether the evolution of configurations on some additive automaton is reversible. In other words, given some monoid $\mathbf{F} = \langle S, \circ \rangle$ and a graph G, when is the map $\rho_F : \mathcal{C}_G \to \mathcal{C}_G$ injective? Note that for finite G reversibility is equivalent with surjectivity. Thus the existence of predecessors in finite reversible cellular automata is trivial. By way of contrast the infinite one-dimensional path P_∞ together with, say, rule σ is irreversible. Nonetheless a simple compactness argument shows that any configuration on this automaton has a predecessor.

This section is devoted to the development of methods that help to determine the reversibility with respect to the linear rules σ and σ^+. For very special graphs that correspond to the grid of cells in traditional finite cellular automata such as rectangular grids, cylinders, and tori, one can exploit their geometric properties to explicitly determine their reversibility at least for rule σ. For example it is shown in [10] that the dimension of the kernel of σ on a $n \times m$ grid graph is $\gcd(m+1, n+1) - 1$. Hence every configuration on such a grid has either no predecessor or $2^{\gcd(m+1,n+1)-1}$ predecessors. We are currently unable to obtain a corresponding result for rule σ^+ even for this class of graphs.

A different line of approach is to establish reduction techniques that allow in certain cases to express the reversibility of a graph G in terms of the reversibility of a smaller graph H. To this end recall that the characteristic polynomial $\Phi(x; G)$ of a graph G is defined as $\Phi(x; G) := |xI - A|$ where A is the adjacency matrix of G. $\Phi(x; G)$ is a polynomial with integer coefficients of degree $|V|$ and it follows from the symmetry of A that all eigenvalues of A are real; thus all the roots of $\Phi(x; G)$ are real algebraic numbers. $\Phi_2(x; G)$ will denote the image of $\Phi(x; G)$ under the natural quotient map $\mathbf{Z} \to \mathbf{F}_2$, so $\Phi_2(x; G)$ is the determinant of $A + xI$ computed over $\mathbf{F}_2$. Then clearly $\Phi_2(x; G)$ has a root 1 [respectively 0] over $\mathbf{F}_2$ iff G is irreversible under rule σ^+ [respectively σ]. Let us adopt the following notational convention: the triple equality sign $\equiv$ indicates that an equation holds over $\mathbf{F}_2$. As usual we will write $G - v$ for the graph obtained from G by deleting the vertex v and $G - e$ for the graph obtained from G by deleting the edge e. The following two results are simple corollaries to theorem 3.2 and 3.4 in [7].

Theorem 4.1. *Deleting a Vertex*

If v is any vertex of G then

$$\Phi_2(x;G) \equiv x\Phi_2(x;G-v) + \sum_{w\in\Gamma(v)} \Phi_2(x;G-v,w).$$

In particular let v be a cutpoint of degree m and let the components of $G-v$ be $G_1+\cdots+G_m$. Let G_i^- denote the graph obtained from G_i by deleting the vertex adjacent to v. Then

$$\Phi_2(x;G) \equiv x\prod_{i\in[m]} \Phi_2(x;G_i) + \sum_{i\in[m]} \Phi_2(x;G_i^-)\prod_{i\neq j}\Phi_2(x;G_j).$$

Theorem 4.2. *Deleting an Edge*
If e is the edge between vertex v and w then

$$\Phi_2(x;G) \equiv \Phi_2(x;G-e) + \Phi_2(x;G-v,w).$$

Here as well as in the sequel we assume for the sake of consistency that the empty graph K_0 with no vertices is reversible under both rules. We begin with a brief comment on rule σ. It is easy to see that for any graph to be reversible under rule σ all the neighborhoods must be distinct, i.e., for all vertices $x \neq y$ we must have $\Gamma(x) \neq \Gamma(y)$. Thus a graph with, say, double endpoints is always irreversible for rule σ. As an immediate consequence of theorem 4.1 we have the following lemma.

Lemma 4.1. *Deleting an Endpoint*
If G has an endpoint v, adjacent to some vertex w, then G is reversible under rule σ iff $G-v,w$ is reversible under rule σ.

Repeated application of this lemma shows that path P_m is reversible iff m is odd. Note that cycles are always irreversible with respect to σ. Now suppose G is acyclic and repeatedly apply lemma 4.3 until the resulting graph H does not allow ant further reductions. It is easy to see that H is either empty or consists of a number of isolated points. In the former case H and therefore G are reversible, in the latter case they are irreversible. In fact this reduction process can easily be carried out in $O(n)$ steps where is the number of vertices in G. We will describe the corresponding and more complicated algorithm for rule σ^+ below (see theorem 4.4) and therefore omit a detailed description of the method for rule σ. The algorithm can also be used to determine the reversibility of unicyclic graphs (graphs containing only one cycle): in this case H may also consist of a cycle. G is then irreversible. Hence we have the following result.

Theorem 4.3. *There is a linear time algorithm to determine whether a unicyclic graph is reversible under rule σ.*

The situation is slightly more complicated with respect to rule σ^+. There are several reduction techniques that allow to delete certain points and/or edges in a graph without changing its reversibility with respect to rule σ^+. We will only prove lemma 4.2, the argument in any other case is entirely similar. We begin with rule σ^+.

Lemma 4.2. *Let G be a graph that is obtained from some other graph H by subdividing edge $e = \{a, b\}$ into a 4-path a,u,v,w,b. Then G is reversible under rule σ^+ iff H is reversible under rule σ^+.*

Proof. It suffices to show that $\Phi_2(1; G) \equiv \Phi_2(1; H)$. Using theorem 4.1 and 4.2 we get $\Phi_2(x; G) \equiv x\Phi_2(x; G - v) + \Phi_2(x; G - u, v) + \Phi_2(x; G - v, w)$
$\equiv x^2\Phi_2(x; G-u, v)+x\Phi_2(x; G-a, u, v)+\Phi_2(x; G-u, v)+x\Phi_2(x; G-u, v, w)+$
$\Phi_2(x; G - a, u, v, w)$
$\equiv (x^2+1)\Phi_2(x; G-u, v)+x^2\Phi_2(x; G-a, u, v, w)+x\Phi_2(x; G-a, u, v, w, b)+$
$x\Phi_2(x; G - u, v, w) + \Phi_2(x; G - a, u, v, w)$
$\equiv (x^2+1)(\Phi_2(x; G-u, v)+\Phi_2(x; G-a, u, v, w))+x(\Phi_2(x; H-a, b)+\Phi_2(x; H-$
$e))$
$\equiv (x^2 + 1)(\Phi_2(x; G - u, v) + \Phi_2(x; G - a, u, v, w)) + x\Phi_2(x; H)$.
Substituting $x = 1$ we obtain $\Phi_2(1; G) \equiv \Phi_2(1; H)$ as desired. ■

The next lemma summarizes two reductions applicable to endpoints in a graph. A vertex with degree 2 is called a *pre-endpoint* if it is adjacent to an endpoint.

Lemma 4.3. *Deleting Double Endpoints / Deleting a Pre-Endpoint*
If G has endpoints u,v, both adjacent to w, then G is reversible under rule σ^+ iff $G - u, v$ is reversible under rule σ^+.
If G has a pre-endpoint v adjacent to endpoint u and some other point w, then G is reversible under rule σ^+ iff $G - u, v, w$ is reversible under rule σ^+.

Repeated application of the above reduction rules shows that the path on m points P_m is reversible with respect to rule σ^+ iff $m \not\equiv 2 \pmod 3$. The cycle on m points C_m is reversible with respect to rule σ^+ iff $m \not\equiv 0$ (mod 3).

Call a graph G σ^+*-irreducible* iff none of the above reductions (4-path, double endpoint, pre-endpoint) can be applied to it. For the special case where G is acyclic note that none of the reductions introduces cycles, hence the graph remains acyclic. The next proposition describes shows the only irreducible connected acyclic graphs are the empty graph, isolated points and single edges K_2.

Lemma 4.4. *Let T be non-empty, irreducible and acyclic. Then T consists exclusively of isolated points and isolated edges (i.e., copies of K_1 and K_2).*

Proof. Suppose T is irreducible, acyclic, and not empty. Let d be the diameter of T and pick a path $v_0, v_1, v_2, \ldots, v_d$. If $d \leq 1$ there is nothing to show, so suppose $d \geq 2$. Since v_0 is an endpoint adjacent to v_1 it follows from the irreducibility of T that no other vertex adjacent to v_1 can be an endpoint. But $v_0, v_1, v_2, \ldots, v_d$ is a path of maximal length in T, hence v_1 must be a pre-endpoint. This contradicts the irreducibility of T. ∎

Connected acyclic graphs are usually referred to as trees and correspondingly acyclic graphs as forests. Hence we have the following corollary.

Corollary 4.1. *A tree T is reversible under rule σ^+ iff it is reducible to a forest of isolated points.*

In [1] Andrasfai gives an algorithm to determine reversibility of a tree under rule σ^+ that has quadratic running time. Our next result shows that in fact a linear algorithm exists.

Theorem 4.4. *There is a linear time algorithm to determine whether a tree is reversible under rule σ^+.*

Proof. According to the last corollary it suffices to show that one can implement a reduction procedure to delete double endpoints and pre-endpoints in linear time. So let $T = \langle V, E \rangle$ be a tree represented by its adjacency lists. In a precomputation the algorithm first determines the degree $d[v]$ for each vertex v and the set V_1 of all endpoints. This can clearly be accomplished in $O(n)$ steps where $n := |V|$. The algorithm then successively deletes all endpoints. To be more explicit, suppose x is an endpoint and y is adjacent to x. Then x is deleted and the degree of y decremented. To find double endpoints, vertex y is marked when x is encountered and deleted. If later another endpoint $\bar{x}$ adjacent to y is found, y is unmarked. Unmarked vertices can also be subject to a pre-endpoint reduction (which occurs whenever $d[y] = 1$). In short the algorithm can be described as follows.

```
while $V_1 \neq \emptyset$ do
pick an endpoint $x$ from $V_1$, let $y$ be the vertex adjacent to $x$;
if $y$ is marked with, say, $\bar{x}$
then
        delete $x$ and $\bar{x}$, unmark $y$;
        $d[y] := d[y] - 2$;
        if $d[y] = 1$ then add $y$ to $V_1$;
else
                $d[y] \geq 3$:mark $y$ with $x$;
                $d[y] = 2$:let $z$ be the other vertex $y$ is adjacent to
                        delete vertices $x$,$y$,$z$;
                        for all vertices $u$ adjacent to $z$, $u \neq y$ do
                                $d[u] := d[u] - 1$;
                                if $d[u] = 1$ then add $u$ to $V_1$;
                                if  $u$ is marked with $\bar{x}$ then unmark $u$,
                                add $\bar{x}$ to $V_1$;
                $d[y] = 1$; return( "irreversible" );
return( "reversible" );
```

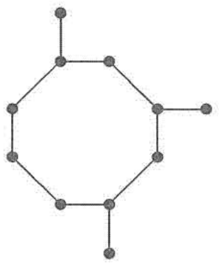

Figure 2: The cogwheel denoted by $\alpha\beta\alpha\beta\alpha\beta\alpha\alpha$.

The algorithm uses a bit-vector to keep track of deleted vertices and two arrays for the degree and markings. V_1 is implemented as a stack or queue. It is easy to see that its total running time (including the precomputation) is $O(n)$. ■

We now expand the last result to unicyclic graphs. It is easy to see that reducing a graph with respect to double-endpoint or pre-endpoint reductions results in an irreducible graph whose connected components are isolated points, K_2's and possibly one *cogwheel*. A cogwheel is a unicyclic connected graph whose vertices all have degrees 1, 2, or 3, and the degree 2 and 3 vertices all lie on the cycle. Hence the endpoints are attached to the degree 3 points. It is convenient to represent a cogwheel by a word over the alphabet $\{\alpha, \beta\}$: symbol α corresponds to a degree 2 vertex and β corresponds to a degree 3 point with its attached endpoint. Thus the word $\beta\alpha\beta\alpha\beta\alpha\alpha\alpha$ represents the cogwheel shown in figure 2. This representation is unique up to shift and reversal, for the sake of simplicity we will identify a word with its equivalence class under these operations. Using theorem 4.1 and 4.2 it is straightforward – though somewhat tedious – to verify that deletion of any of the following five subwords does not affect the reversibility of a cogwheel with respect to rule σ^+: $\alpha\alpha\alpha$, $\beta\beta$, $(\alpha\beta)^2$, $(\alpha\alpha\beta)^2$, $(\alpha\beta\alpha\alpha\beta)^3$.

Thus one may delete three consecutive degree 2 points from a cogwheel without affecting its reversibility under rule σ^+ (this is simply lemma 5.3). For consistency let us say that the cogwheel corresponding to the empty word ε is irreversible. For later reference let

$$R := \{\alpha\alpha\alpha, \beta\beta, (\alpha\beta)^2, (\alpha\alpha\beta)^2, (\alpha\beta\alpha\alpha\beta)^3\}.$$

W in $\{\alpha, \beta\}^*$ is *reducible* to W' iff W' can be obtained from W by finitely many deletions according to 4.10. W is irreducible iff W cannot be reduced to any W' shorter than W. Note that in general a word W can be reduced to to several different irreducible words. For example, $W = \alpha\alpha\alpha\beta\alpha\beta\beta$ can be reduced by to both $\alpha\beta$ and $\alpha\alpha\beta$. It is not hard to see that any irreducible cogwheel must be one of the following:

ε	$\alpha\beta$
α	$\alpha\alpha\beta$
β	$\alpha\beta\alpha\alpha\beta$
$\alpha\alpha$	$(\alpha\beta\alpha\alpha\beta)^2$

Strictly speaking, the first five words denote multi-graphs. However, as far as reversibility is concerned, one may simply remove double edges to obtain corresponding graphs. Of these irreducible graphs only ε, β, $\alpha\beta$ and $\alpha\alpha\beta$ are irreversible under rule σ^+, all others are reversible. Thus W is an irreversible cogwheel iff W reduces to one of ε, β, $\alpha\beta$ and $\alpha\alpha\beta$.

Next we will show that one can construct an irreducible cogwheel from a given one in $O(n)$ steps where n is the number of points in the graph. Hence we have the following extension of theorem 4.4.

Theorem 4.5. *There is a linear time algorithm to determine whether a unicyclic graph is reversible under rule* σ^+.

Proof. We may use the algorithm of theorem 4.4 to reduce the given graph to a cogwheel in linear time. The cogwheel is represented by a word $W \in \{\alpha, \beta\}^*$ as described above. Now consider the following finite transition system $\mathfrak{R}$. $\mathfrak{R}$ has as set of states Q all the prefixes of irreducible words. The transition function is define by

$$\delta(q,\sigma) := \begin{cases} q\sigma & \text{if } q\sigma \in Q, \\ p & \text{if } q\sigma \notin Q \vee \exists w \in R(pw = q\sigma). \end{cases}$$

Since no word in R is a postfix of another δ is well-defined and $\mathfrak{R}$ is deterministic. The initial state is $q_0 = \varepsilon$. A straightforward induction shows that W reduces to $\delta(q_0, W)$. Thus if $\delta(q_0, W)$ is irreducible we are done: W is irreversible iff $\delta(q_0, W)$ is one of ε, β, $\alpha\beta$, and $\alpha\alpha\beta$. On the other hand, if $\delta(q_0, W)$ is still reducible, one can determine a corresponding irreducible word by a simple table look-up. For example, let $W = \alpha\beta\alpha\beta\alpha\beta\alpha\alpha$. The $\delta(q_0, W) = \alpha\beta\alpha\alpha$ which further reduces to β.

Hence one can determine reversibility of W and therefore reversibility of G in linear time. This finishes the argument. ∎

It seems rather difficult to characterize general graphs containing many cycles with respect to reversibility under rules σ and σ^+. It would be interesting to know whether there is a fast algorithm – say linear in the size of the graph G – that can determine the reversibility of σ and σ^+ on G. Note that his excludes the brute force solution of computing the determinant of the adjacency and neighborhood matrices over $\mathbf{F}_2$.

References

[1] B. Andrasfai. "Cellular automata in trees", *Finite and Infinite Sets* **37**, Colloquia Mathematica Societatis Janos Bolyai, Eger, Hungary, 1981.

[2] C. Berge, *Graphs and Hypergraphs* (North-Holland, 1973).

[3] N. Christofides, *Graph Theory* (London: Academic Press, 1975).

[4] M. R. Garey and D. S. Johnson, *Computers and Intractability* (Freeman, 1979).

[5] A. Lindenmayer, "Mathematical models for cellular interactions in development", *J. Theoret. Biol.* **18**, 1986, 280–299.

[6] O. Martin, A. M. Odlyzko, and S. Wolfram, "Algebraic properties of cellular automata", *Commun. Math. Phys.* **93**, 1984, 219–258.

[7] A. J. Schwenk and R. J. Wilson, "On the eigenvalues of a graph" in *Selected Topics in Graph Theory*, eds. L. W. Beineke and R. J. Wilson, editors (Academic Press, 1970).

[8] K. Sutner, "The complexity of finite cellular automata", submitted.

[9] K. Sutner, "Linear cellular automata and the Garden-of-Eden", to appear.

[10] K. Sutner, "On σ-automata", *Complex Systems* **2:1**, 1988, 1–28.

[11] M. Sved, "Divisibility – with visibility", *Mathematical Intelligencer* **10:2**, 1988, 56–64.

[12] S. Wolfram, "Statistical mechanics and cellular automata", *Rev. Modern Physics* **55:3**, 1983, 601–644.

[13] S. Wolfram, "Geometry of binomial coefficients", *American Mathematical Monthly* **91**, 1984, 566 – 571.

[14] T. Yaku, "The constructibility of a configuration in a cellular automaton", *Journal of Computers and System Science* **7**, 1973, 481–496.