

# Linear Cellular Automata and Fischer Automata

Klaus Sutner

*Carnegie Mellon University*

*Pittsburgh, PA 15213*

sutner@cs.cmu.edu

<http://www.cs.cmu.edu/~sutner>

## Abstract

We study the sizes of minimal finite state machines associated with linear cellular automata. In particular, we construct a class of binary linear cellular automata whose corresponding minimal automata exhibit full exponential blow-up. These cellular automata have Hamming distance 1 to a permutation automaton. Moreover, the corresponding minimal Fischer automata as well as the minimal DFAs have maximal complexity. By contrast, the complexity of higher iterates of a cellular automaton always stays below the theoretical upper bound.

## 1 Introduction

Every linear cellular automaton  $\rho$  can be associated with a regular language  $\mathcal{L}(\rho)$  of finite words:  $\mathcal{L}(\rho)$  is the collection of all finite subwords of configurations that arise after one application of the global map of the cellular automaton. Discussions of the language theoretic aspects of linear cellular automata and sofic systems, in particular with respect to their relation to the topology of the space of configurations, can be found in [8], [10] and [7]. In this paper, we will study two measures of complexity associated with  $\mathcal{L}(\rho)$  that are based on minimal finite state machines of a certain type. The first is simply the size of the minimal automaton for  $\mathcal{L}(\rho)$ , or, equivalently, the number of left quotients of this language. For the second measure, one can exploit the fact that the languages  $\mathcal{L}(\rho)$  are not only regular but also factorial and transitive. As a consequence, there is a deterministic, transitive semiautomaton that accepts such a language, see [3]. We refer to any deterministic, transitive semiautomaton as a *Fischer automaton*. We will show that there is a natural embedding of the minimal Fischer automaton into the minimal automaton. In fact,

the minimal Fischer automaton turns out to be a strongly connected component of the minimal automaton. The component is characterized by the fact that it has exits only to the sink of the minimal automaton. The other components of the minimal automaton, if they exist, construed as semiautomata, accept proper subsets of the acceptance language of the whole machine. This explains an observation by Wolfram that the minimal automaton associated with a linear cellular automaton may contain additional “transient subgraphs,” see [24]. It was shown by Beauquier [2] that the minimal Fischer automaton is also minimal in the sense of homomorphisms: there is a homomorphism from any Fischer automaton for a fixed transitive factorial language to the minimal Fischer automaton for that language.

This work is motivated in part by a problem posed by Wolfram in [26]. The iterates  $\rho^t$  of a linear cellular automaton  $\rho$  define a descending sequence of regular languages  $\mathcal{L}(\rho^t)$ . Wolfram’s empirical studies led him to the conjecture that the complexity of these languages is in general non-decreasing. As a matter of fact, it appears that for most Wolfram class *I* and *II* automata,  $\mu(\rho^t)$  increases polynomially, but for class *III* and *IV*, complexities increase exponentially. A table of  $\mu(\rho^t)$  for elementary cellular automata can be found in [24]. It is noted in the reference that  $\mu(\rho^t)$  “usually stays far below the upper bound.”

To see what this upper bound is, note that there is a transitive semiautomaton  $B(\rho)$  that accepts  $\mathcal{L}(\rho)$  whose underlying diagram is a de Bruijn graph. This de Bruijn automaton has  $k^{w-1}$  states and  $k^w$  transitions where  $k$  is the size of the alphabet and  $w$  the width of the local map of the linear cellular automaton. Hence, if we let  $\mu(\rho)$  be size of the minimal automaton for  $\mathcal{L}(\rho)$ , we immediately obtain the upper bound  $\mu(\rho) \leq 2^{k^{w-1}}$ . It follows that the iterates of  $\rho$  obey the bound  $\mu(\rho^t) \leq 2^{k^{t(w-1)}}$ . Let us write  $\mu_F(\rho)$  for the size of the minimal Fischer automaton for  $\mathcal{L}(\rho)$ . From our embedding result for Fischer automata, we have  $\mu_F(\rho) \leq \mu(\rho)$ . Equality occurs only in the trivial case  $\mu_F(\rho) = \mu(\rho) = 1$ , i.e., when  $\mathcal{L}(\rho) = \Sigma^*$ . By compactness, the latter condition is equivalent with the global map of the cellular automaton being surjective. Thus, for non-surjective global maps, we have

$$1 \leq \mu_F(\rho) < \mu(\rho) \leq 2^{k^{w-1}}.$$

We will construct a fairly large class of binary cellular automata of arbitrary width that shows that both bounds are tight: for these cellular automata the corresponding minimal automaton has size  $2^{2^{w-1}}$ , and the corresponding minimal Fischer automaton differs in only one state, the sink of the minimal automaton. The cellular automata are constructed from a permutation automaton by changing the label of one transition. Thus, they have Hamming distance 1 to the nearest permutation automaton. We will see that any cellular automaton with Hamming distance larger than 1 fails to have Fischer automata of maximal size. It is quite straightforward to show that in particular the iterates  $\rho^t$ ,  $t \geq 2$ , of any non-permutation automaton  $\rho$  all must have Hamming distance larger than 1 to the nearest permutation automaton. Therefore, the regular languages

associated with these cellular automata exhibit a relative decrease in their complexity. In general, there appears to be a rather close connection between the Hamming distance of  $\rho$  to the nearest permutation automaton, and the size of the minimal automaton, but we are currently unable to give a detailed analysis.

Note that a cellular automaton whose de Bruijn automaton  $B(\rho)$  is a permutation automaton is surjective, in fact; the global map is open. Surjectivity of the global map is also equivalent to a strong balance property of  $B(\rho)$ : every word has to have the same multiplicity in  $B(\rho)$ , see [4] and [16]. In particular, the number of transitions in  $B(\rho)$  labeled by any particular symbol in the alphabet is the same as for any other symbol. It was suggested by Langton to use the simple numerical parameter

$$\lambda(\rho) := \frac{k^w - k_0}{k^w}$$

as a measure for the complexity of a CA, see [13], [12], [11] and also [14]. Here  $k_0 := |\rho^{-1}(0)|$  is the number of words mapped to the quiescent state 0 by the local map. Langton studied randomly generated cellular automata and it appears that automata whose  $\lambda$ -values are near a critical value  $\lambda_c$  tend to exhibit extremely complicated behavior, in particular they seem to belong to Wolfram's class *IV*, see [25]. Langton has coined the term "edge-of-chaos" to characterize the phenomenon that in certain regions of the space of all local rules small changes seem to be associated with a transition to highly complex behavior. A similar phenomenon is reported in [15]. There the authors study the connections between dynamical behavior and computational capabilities of cellular automata. They use a genetic algorithm to evolve a class of cellular automata that are able to perform a specialized computation, in this case the computation of majorities in the initial configuration. As it turns out, these automata have  $\lambda$ -values near  $1/2$ , the value characteristic for surjective rules. The cellular automata constructed below that demonstrate exponential blow-up also have  $\lambda$ -values near  $1/2$ : since only one label is changed in a permutation automaton, we have  $\lambda = 1/2 \pm 2^{-w}$ .

Because of the potentially exponential blow-up of the deterministic finite state machine associated with a linear cellular automaton, any surjectivity testing algorithm based on minimization will in general have exponential space complexity. Note, though, that  $B(\rho)$  can be used to determine surjectivity (as well as openness and injectivity) in quadratic time. The fast algorithm uses products of semiautomata rather than the power automaton construction needed for minimization, see [1] and [21]. On the other hand, there is no hope of obtaining a strict classification of cellular automata along the lines of Wolfram's classes by means of easily computable parameters: questions relating to the Wolfram hierarchy are in general highly undecidable, even if we restrict our attention to finite cellular automata, see [9], [19], and [20]. As it turns out, even computing the size of the power automaton, or the minimal automaton, of a given nondeterministic machine is hard. For example, it is shown in [22] that it is PSPACE-hard to test whether the size of the accessible

part of power automaton of a given nondeterministic semiautomaton exceeds a given bound. Using the Immerman-Szelepsényi machinery, it is not hard to see that that the problem can be solved in  $\text{NSPACE}(n)$ , and is therefore  $\text{PSPACE}$ -complete. Strictly speaking, the underlying alphabet has to have cardinality at least 2, otherwise the problem is trivially solvable in polynomial time. Needless to say, the construction given in the reference does not apply to the de Bruijn automata that arise in the context of linear cellular automata. Nonetheless, we conjecture that is hard to compute the size of the power automaton even for  $B(\rho)$ .

This paper is organized as follows. In section 2 we briefly review some of the concepts of the theory of finite state machines that will be used in the following. In section 3 we prove our embedding result for minimal Fischer automata and construct a class of semiautomata that exhibit exponential blow-up during deterministic simulation and minimization. Lastly, in section 4 we apply these results to cellular automata and give some experimental results.

## 2 Preliminaries

Let  $\Sigma$  be a finite alphabet. A *linear cellular automaton* (or CA, for short) with alphabet  $\Sigma$  is a continuous, shift-invariant map  $\rho_\infty : \Sigma^\infty \rightarrow \Sigma^\infty$  where  $\Sigma^\infty = \{X \mid X : \mathbb{Z} \rightarrow \Sigma\}$  is the space of all *configurations* of the CA. For the sake of simplicity, we choose  $\Sigma$  to be a standard alphabet  $\Sigma_k = \{0, 1, \dots, k-1\}$ . We can describe a CA as a finite data structure  $\langle \rho, w, k \rangle$  where  $k$  is the size of the alphabet,  $w$  is the *width* of the CA, and  $\rho : \Sigma_k^w \rightarrow \Sigma_k$  is the *local map* or *local rule* of the automaton. If the values of  $w$  and  $k$  are obvious from context, we will simply write  $\rho$  for the CA. The iterates of  $\rho$  are denoted by  $\rho^t$ , so  $\rho^t : \Sigma_k^{t(w-1)} \rightarrow \Sigma_k$ . In the following, we will mostly deal with *binary* CAs whose alphabet is  $\Sigma_2 = \{0, 1\}$ . For our purposes, the *global map*  $\rho_\infty$  is related to the local map by

$$\rho_\infty(X)(i) = \rho(X(i) \dots X(i+w-1)).$$

Since we are mostly interested in the languages associated with linear cellular automata, there is no need to consider more general neighborhoods here.

We now briefly review a few concepts from the theory of finite state machines that are needed in the following discussion. The reader is referred to [6] and [17] for more background information. A *semiautomaton* (or SA) over alphabet  $\Sigma$  is a triple  $\langle Q, \Sigma, \tau \rangle$  where  $Q$  is a finite set of *states* and  $\tau \subseteq Q \times \Sigma \times Q$  is a *transition relation*.  $\tau$  naturally extends to a relation  $\tau^* \subseteq Q \times \Sigma^* \times Q$ . As usual, we omit the asterisk and write  $\tau$  instead of  $\tau^*$ . Also, for any subset  $P$  of  $Q$ , let  $\tau(P, x) := \{q \in Q \mid \exists p \in P : \tau(p, x, q)\}$ . In order to associate a language with a semiautomaton, one selects two sets of states, the *initial states*  $I$  and the *final states*  $F$ . The automaton *accepts* a word  $x$  if  $\tau(I, x) \cap F \neq \emptyset$ , and its *acceptance language*  $\mathcal{L}(M)$  is the set of all such words  $x$ . A semiautomaton

augmented with initial and final states  $\langle Q, \Sigma, \tau, I, F \rangle$  is a *finite automaton* (or FA). An automaton is *accessible* if  $\tau(I, \Sigma^*) = Q$ . For a plain semiautomaton we always assume  $Q = I = F$  whenever we need to make reference to initial or final states. Thus, a semiautomaton  $M$  accepts  $x$  iff  $\tau(Q, x) \neq \emptyset$  and acceptance is really a path-existence problem in the underlying diagram  $\Delta_M$  of the automaton:  $M$  accepts a word  $x$  iff there is a path in  $\Delta_M$  labeled  $x$ . Lastly, an automaton is *binary* if its alphabet is  $\Sigma_2 = \{0, 1\}$ .

It will often be convenient to identify states  $p$  with the corresponding singleton sets  $\{p\}$ . The *behavior*  $\llbracket P \rrbracket$  of a set of states  $P$  is the set of all words  $x$  such that  $\tau(P, x) \cap F \neq \emptyset$ . Thus, the acceptance language of an automaton is the behavior of the initial states of the machine. An automaton is *reduced* if all its states have distinct behavior. The *reverse machine*  $\text{rev}(M)$  is obtained by reversing all transitions and interchanging the initial and final states.

A state  $p$  in  $M$  is *nondeterministic* if there are transitions  $\tau(p, a, q)$  and  $\tau(p, a, q')$  where  $q \neq q'$ ,  $a \in \Sigma$ , and *deterministic* otherwise. State  $p$  is *complete* if  $\forall a \in \Sigma \exists q : \tau(p, a, q)$ . The machine is deterministic (complete) if all its states are deterministic (complete). A complete, deterministic machine with exactly one initial state is called a *deterministic finite automaton* (or DFA). A *sink* in a DFA is a state  $\perp$  that fails to be final and such that  $\tau(\perp, a, p)$  implies  $p = \perp$  for all  $a \in \Sigma$ . A *sink automaton* is a DFA that has a sink and all states other than the sink are final. A state is *codeterministic* if it is deterministic in the reverse machine and *bideterministic* if it is both deterministic and codeterministic. Similarly, a machine is codeterministic (bideterministic) if all its states are codeterministic (bideterministic). Bideterministic machines play an important role in algebraic automata theory and in particular in the Krohn-Rhodes decomposition theorem. In this context, it is customary to refer to them as *permutation automata*, see [5].

Two machines are *equivalent* if their acceptance languages coincide. It is well-known that for every FA there exists an equivalent DFA. Among the equivalent DFAs, there is a uniquely determined one (up to isomorphism) with the minimum possible number of states. This DFA is referred to as the *minimal automaton* of the corresponding regular language. Moreover, the size of the minimal automaton is at most  $2^n$  where  $n$  is the size of any equivalent FA.

In a deterministic, complete automaton the transition relation can be thought of as a right action  $Q \times \Sigma^* \rightarrow Q$  that turns  $Q$  into a semimodule over the monoid  $\Sigma^*$ . Similarly, in an arbitrary automaton,  $\tau$  gives rise to a semimodule structure on  $\text{pow}(Q)$ . To lighten notation, we will write  $P \cdot x$  rather than  $\tau(P, x)$  for  $P \subseteq Q$ . A homomorphism of machines is a homomorphism of the corresponding semimodules. In particular, for DFAs a map  $f : Q_1 \rightarrow Q_2$  is a homomorphism if  $f(p \cdot a) = f(p) \cdot a$  for all symbols  $a \in \Sigma$ . One can show that for every accessible DFA  $M$  there is an epimorphism from  $M$  to the corresponding minimal automaton. In fact, this map preserves the initial as well as the final states.

For our arguments below, we need two standard ways of constructing the minimal automaton for a regular language. The first method assumes that we are given a nondeterministic automaton  $M$  for  $L$  and consists of two independent steps. First, convert  $M$  into an accessible DFA  $M'$ , the *power automaton* of  $M$ . Here, as well as throughout this paper, the power automaton is understood to be the accessible part of the full power automaton. Thus, the size of the power automaton lies between 1 and  $2^n$  where  $n$  is the number of states of the nondeterministic machine. Second, identify states in  $M'$  with the same behavior to obtain a factor automaton  $M'' = M'/\approx$ . Here  $\approx$  is the congruence relation induced by the behavior map  $p \mapsto \llbracket p \rrbracket$ . Then  $M''$  is (isomorphic to) the minimal automaton for  $L$ .

Alternatively, we can construct the minimal automaton directly from the language  $L$  using *quotients*. To this end, define the quotient of a language  $L$  by a word  $x$  to be the language

$$x^{-1}L := \{y \mid xy \in L\}.$$

Thus  $x^{-1}L$  is the collection of all words that can be obtained from a word in  $L$  by deleting its prefix  $x$ . Also let  $K^{-1}L := \bigcup \{x^{-1}L \mid x \in K\}$  for any language  $K$ . The set  $Q_L := \{x^{-1}L \mid x \in \Sigma^*\}$  of all quotients of a regular language  $L$  is finite and can be turned into a DFA  $M_L$  by defining the action  $K \cdot a := a^{-1}K$ ,  $I = \{L\}$  and  $F = \{K \in Q_L \mid \varepsilon \in K\}$ . This DFA is (isomorphic to) the minimal automaton for  $L$  and the behavior of state  $K$  in  $M_L$  is  $K$  itself. Given an arbitrary accessible DFA  $M$  for  $L$ , the map  $p \mapsto \llbracket p \rrbracket$  is an epimorphism from  $M$  to  $M_L$ . The number of all quotients of a regular language  $L$  (or, equivalently, the size of the minimal automaton) is the *complexity* of  $L$  and is denoted  $\mu(L)$ . Thus, we have the following theorem.

**Theorem 2.1** *Minimal Automaton*

*For every regular language  $L$ , there is a uniquely determined accessible reduced DFA  $M$  (up to isomorphism), the minimal automaton of  $L$ . The states of  $M$  can be identified with the quotients of  $L$ ; hence the size of  $M$  is  $\mu(L)$ .*

If the minimal automaton is a sink automaton, it is usually more convenient to consider the corresponding partial automaton that is obtained by deleting the sink. We will refer to this machine as the *partial minimal automaton*.

### 3 Subshifts and the Language of a CA

A *subshift* is a subset  $\mathcal{X}$  of  $\Sigma^\infty$  that is closed and shift-invariant. The connection between subshifts and language theory is established by associating a configuration  $X \in \Sigma^\infty$  with its *cover*, the set of all finite factors of  $X$ . Formally,  $\text{cov}(X) := \{X(i) \dots X(j) \mid i \leq j\} \cup \{\varepsilon\} \subseteq \Sigma^*$ . For a set  $\mathcal{X}$

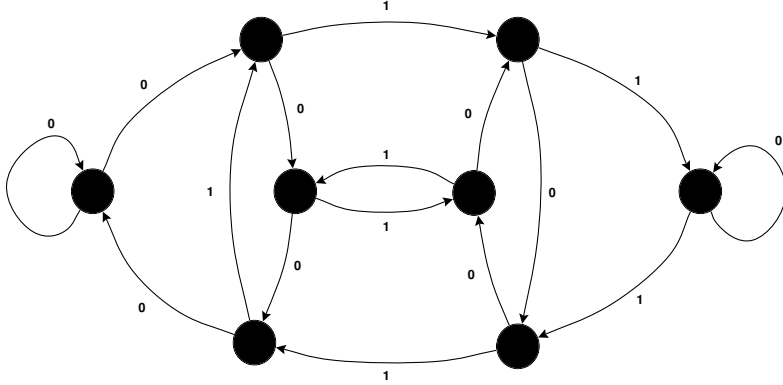


Figure 1: The de Bruijn automaton  $B(22184, 4, 2)$ . Disregarding the self-loop at 000, the machine is a permutation automaton with three 0-cycles and two 1-cycles.

of configurations define its cover  $\text{cov}(\mathcal{X})$  to be the union of all the covers  $\text{cov}(X)$  where  $X \in \mathcal{X}$ . Note that a subshift can be reconstructed from its cover, and it may be more convenient to work with the cover rather than the space of biinfinite configurations directly. For example, a subshift of finite type is characterized by the fact that its cover is a finite complement language, that is, there is a finite set  $F$  of words over  $\Sigma^*$  such that  $\text{cov}(\mathcal{X}) = \Sigma^* - \Sigma^*F\Sigma^*$ , see [23]. Likewise, a subshift  $\mathcal{X}$  is a *sofic system* if  $\text{cov}(\mathcal{X})$  is a regular language. The proper homomorphisms for subshifts are given by continuous maps that commute with the shift, that is, by cellular automata. It is shown in the reference that every sofic system is the homomorphic image of a subshift of finite type.

The image of a cellular automaton is plainly a subshift, so we may instead study its cover. More formally, define the *language of  $\rho$*  to be  $\mathcal{L}(\rho) := \text{cov}(\text{rg } \rho_\infty)$ . To see that these languages are indeed regular, note that we can use the local map  $\rho$  to label the edges of a suitable *de Bruijn graph*. The resulting semiautomaton accepts  $\mathcal{L}(\rho)$ : define the *de Bruijn automaton*  $B(\rho, w, k)$ , or simply  $B(\rho)$ , by

$$\langle \Sigma_k^{w-1}, \Sigma_k, \{ (ax, \rho(axb), xb) \mid a, b \in \Sigma, x \in \Sigma_k^{w-1} \} \rangle.$$

The de Bruijn automaton  $B(22184, 4, 2)$  is shown in figure 1, see below for an explanation of the numbering convention used. Every biinfinite path in  $B(\rho)$  corresponds to a unique configuration  $X$  and the label sequence of the path is  $\rho_\infty(X)$ . The factors of  $\rho_\infty(X)$  therefore occur as the label sequences of finite paths in  $B(\rho)$  and, as a semiautomaton,  $B(\rho)$  accepts  $\mathcal{L}(\rho)$ . Note that as a data structure, the size of the de Bruijn semiautomaton and the CA are essentially the same. Clearly,  $1 \leq \mu(\rho) \leq 2^{k^{w-1}}$  and the lower bound is reached iff the global map of the CA is surjective.

De Bruijn automata can also be used to construct semiautomata for subshifts of finite type by deleting some of the edges. For many cellular automata  $\rho$ , the languages  $\mathcal{L}(\rho)$  turn out to be finite complement, so there may well be several natural machines associated with  $\mathcal{L}(\rho)$ .

The languages associated with subshifts have a number of special properties. A language  $L$  is

factorial if  $L$  is closed with respect to factors ( $uxv \in L$  implies that  $x \in L$ ). A language  $L \neq \emptyset$  is *extensible* if  $x \in L$  implies that there are symbols  $a, b \in \Sigma$  such that  $axb \in L$ . Lastly, let  $L \neq \emptyset, \{\varepsilon\}$ .  $L$  is *transitive* if  $u, v \in L$  implies that  $uxv \in L$  for some  $x$ . Note that any transitive factorial language is also extensible. The cover of any subshift is factorial and extensible, and for sofic systems also regular. In particular the language of a cellular automaton is regular, factorial and transitive, and is accepted by the canonical de Bruijn automaton  $B(\rho)$ .

To describe the finite state machines corresponding to the covers of sofic systems in general, call a semiautomaton *non-transient* if all its states lie on a biinfinite path in the diagram of the automaton. The machine is *transitive* if its transition diagram has only one strongly connected component. A *pre-Fischer automaton* is a deterministic, non-transient semiautomaton, and a *pre-Fischer automaton* is a deterministic, transitive semiautomaton. It is clear that every pre-Fischer automaton accepts a regular, factorial and extensible language, and the acceptance language of a Fischer automaton is in addition transitive. For the opposite direction, we first establish the following proposition.

**Proposition 3.1** *A language  $L \subseteq \Sigma^*$  is transitive iff for all words  $x \in L$  we have  $L \subseteq (x\Sigma^*)^{-1}L$ . If the language is in addition factorial, equality holds.*

*Proof.* Suppose  $L$  is transitive and let  $x \in L$ . Consider an arbitrary word  $y \in L$ , and pick a word  $u$  such that  $xuy$  is in  $L$ . Then  $y = (xu)^{-1}xuy$  is in  $(x\Sigma^*)^{-1}L$ . On the other hand, consider two words  $x, y \in L$ . Since  $L \subseteq (x\Sigma^*)^{-1}L$ , we must have  $xuy \in L$  for some  $u$ , hence  $L$  is transitive. The second claim follows easily from the first.  $\square$

**Lemma 3.1** *Let  $L$  be a regular, factorial and extensible language. Then there is a pre-Fischer automaton that accepts  $L$ . If  $L$  is in addition transitive, then there is a Fischer automaton that accepts  $L$ .*

*Proof.* We may safely assume that  $L$  is different from  $\Sigma^*$ . Hence, the minimal automaton for  $L$  is a sink automaton  $M_\perp$ . Let  $M$  denote the corresponding partial automaton and denote by  $M_0$  the subautomaton of  $M$  that is induced by the non-transient part of  $M$ . We claim that  $M_0$  accepts  $L$ . To see this, note that by extensibility any word  $x$  in  $L$  can be embedded in a word  $uxv \in L$  where both  $u$  and  $v$  are arbitrarily long. In particular, we can choose  $u$  and  $v$  to be longer than any transient part of  $M$ . Hence, the part of the accepting computation of  $M$  on  $uxv$  that deals with  $x$  lies entirely within  $M_0$ , and we are done.

Now consider the case where  $L$  is also transitive. Let  $\Delta_M$  be the diagram of  $M$  and let  $G$  be the *collapse* of  $\Delta_M$ , i.e., the acyclic digraph whose nodes are the strongly connected components of  $\Delta_M$ , and whose edges are induced by the edges between these components. Since  $G$  is acyclic, we

can pick a node  $C$  in  $G$  of out-degree zero. Let  $M_C$  be the subautomaton of  $M$  induced by  $C$ . Clearly,  $M_C$  is a Fischer automaton, and it remains to show that  $M_C$  accepts  $L$ . To see this, let  $q_0$  denote the initial state of  $M$  and consider a string  $x \in L$  such that  $q_0 \cdot x \in C$ . Let  $v \in L$  be arbitrary and choose another word  $v'$  such that  $vv'$  is still in  $L$  but  $v'$  is longer than any simple path in  $\Delta_M$ . By the proposition, there is some word  $u$  such that  $vv' \in \llbracket p \rrbracket$  where  $p = q_0 \cdot xu$ . Since  $L$  is also factorial, the path from  $p$  to  $p \cdot vv'$  lies indeed in  $M$ . In particular, the path from  $p$  to  $p \cdot v$  lies entirely within  $C$ . Hence  $v$  is accepted by the semiautomaton  $M_C$ .  $\square$

Note that the last argument also shows that any edge leaving the component  $C$  must have the sink as target, i.e., there are no intermediate states between  $C$  and the sink in  $M_\perp$ .

Let  $M = \langle Q, \Sigma, \tau \rangle$  be a pre-Fischer automaton and consider its power automaton  $\text{pow}(M)$ . Since  $M$  is deterministic we have  $|P \cdot a| \leq |P|$  for any  $P \subseteq Q$ . Now define  $\kappa(M)$  to be the least cardinality of  $P \cdot x \neq \emptyset$  as  $x$  ranges over  $\Sigma^*$ . Let  $M_\kappa$  be the subautomaton of  $\text{pow}(M)$  induced by all  $P$  of cardinality  $\kappa(M)$ . We refer to  $M_\kappa$  as the *kernel automaton* of  $M$ . Let us say that a word  $x$  *focuses*  $P \subseteq Q$  if  $|P \cdot x| = \kappa(M)$ .

**Proposition 3.2** *For any reduced pre-Fischer automaton  $M$  we have  $\kappa(M) = 1$ . If  $M$  is a Fischer automaton, then the kernel automaton  $M_\kappa$  is isomorphic to  $M$ .*

*Proof.* Suppose  $P \subseteq Q$ ,  $|P| > 1$ , and pick  $p_1, p_2 \in P$  and a word  $u$  such that, say,  $u \in \llbracket p_1 \rrbracket - \llbracket p_2 \rrbracket$ .  $u$  exists since  $M$  is reduced. Then  $1 \leq |P \cdot u| < |P|$ , and, repeating this procedure, we obtain the desired word  $x$  by concatenation.

If  $M$  is also transitive, let  $x$  be a word the focuses  $Q$ , say,  $Q \cdot x = p$ . By transitivity, we can find words  $u$  such that  $Q \cdot xu = q$ , for any state  $q \in Q$ . Hence  $M_\kappa$  is an isomorphic copy of  $M$ .  $\square$

A pre-Fischer automaton is *minimal* if there is no pre-Fischer automaton with fewer states that accepts the same language, and likewise for Fischer automata. Note that any minimal pre-Fischer automaton must be reduced, since one could otherwise use state-merging to obtain a smaller machine.

**Theorem 3.1** *Embedding Minimal Fischer Automata*

*For any regular, factorial, transitive language  $L$  there is exactly one minimal Fischer automaton that accepts  $L$ , up to isomorphism. Moreover, this minimal Fischer automaton is isomorphic to a strongly connected component in the minimal automaton for  $L$ . Hence, a Fischer automaton is minimal if and only if it is reduced.*

*Proof.* It suffices to show that the subautomaton  $M_C$  constructed in the last lemma has the

required properties. To show that this automaton is unique up to isomorphism, we establish the following two claims.

*Claim 1:* Let  $F$  be an arbitrary reduced Fischer automaton and  $M$  the partial minimal DFA for  $L$ . Then  $F$  is isomorphic to a strongly connected component of  $M$  that has out-degree zero.

*Claim 2:* Let  $C_1$  and  $C_2$  be two strongly connected components of  $M$  that have out-degree zero. Then  $C_1 = C_2$ .

To prove claim 1, let  $M' = \text{pow}(F)$  be the power automaton of  $F$  (i.e., the accessible part of the full power automaton) and let  $M'' = M'/\approx$  be the corresponding minimal automaton, where  $\approx$  is the congruence relation induced by the behavior map. Then  $M''$  is isomorphic to  $M_\perp$ , and we may safely identify the states of  $M''$  with the behaviors of the states in  $M'$ . By the last proposition, the kernel automaton of  $F$  is isomorphic to  $F$ . Also, there is the natural epimorphism  $\eta : M' \rightarrow M''$  which maps states to their behaviors. Since  $F$  is reduced, the restriction of  $\eta$  to the kernel automaton of  $F$  must be injective. Hence,  $M''$  also contains an isomorphic copy of  $F$ . Moreover, in the collapse of the diagram of  $M''$ , this copy of  $F$  forms a node with out-edges leading only to the sink (recall that  $F$  is deterministic). For suppose  $Q \cdot u = p$  in  $M'$ . Then  $Q \cdot u = p$  also in  $F$ , and for any word  $v$  such that  $p \cdot v \notin F$  in machine  $M'$  we must have  $Q \cdot uv = \emptyset$ . But  $\emptyset$  is the sink in  $M''$ . This concludes the proof of claim 1.

For the second claim, suppose we have two strongly connected components  $C_1$  and  $C_2$  in  $M$  with out-degree zero. Since the semiautomata induced by  $C_1$  and  $C_2$  are both reduced, there are words  $u$  and  $v$  in  $L$  that focus them:  $C_1 \cdot u = p$  and  $C_2 \cdot v = q$  where  $p \in C_1$  and  $q \in C_2$  are single states in the minimal automaton. Pick an intermediate word  $x$  such that  $uxv \in L$ . Clearly,  $uxv$  focuses both  $C_1$  and  $C_2$ , say, to states  $p' \in C_1$  and  $q' \in C_2$ , respectively. Since  $p'$  and  $q'$  have different behavior, there is some word  $w$  such that  $uxvw \in \llbracket C_1 \rrbracket - \llbracket C_2 \rrbracket$  (or vice versa), contradicting the fact that both components have behavior  $L$ .  $\square$

The theorem also provides an alternative proof of a characterization of minimal Fischer automata in terms of homomorphisms: there is a homomorphism from any Fischer automaton for a fixed language onto the minimal Fischer automaton. Since minimality is equivalent with reducedness, the homomorphism is simply the quotient map with respect to behavioral equivalence. See [2] for an alternative proof based on an analysis of 0-minimal ideals in the syntactic semigroup of the language.

There is no similar result for pre-Fischer automata; it is not difficult to construct regular, factorial and extensible languages that admit several different minimal pre-Fischer automata. Moreover, there is no obvious connection between these minimal automata and the least non-transient sub-automaton of the minimal DFA. The latter may fail in particular to be minimal, though it is of

course reduced.

For transitive languages  $L$ , there are in general several strongly connected components in the minimal automaton other than the minimal Fischer automaton. Indeed, it was observed in [24] that the minimal automaton  $M$  of  $B(\rho)$  often contains “transient parts” that feed into a “main part”. By theorem 3.1, this main part is none other than the minimal Fischer automaton. The other strongly connected components, if they exist, correspond to the minimal Fischer automata for certain transitive sublanguages of  $L$ .

At any rate, the theorem allows us to define

$$\mu_F(L) := \text{size of the minimal Fischer automaton of } L$$

for any regular, factorial, transitive language  $L$ . Since the minimal Fischer automaton is nothing but a strongly connected component of the minimal automaton, we have for all  $L \neq \Sigma^*$ ,  $\mu_F(L) \leq \mu(L) - 1$ .

The choice of  $\mathcal{L}(\rho)$ , and  $B(\rho)$  as a corresponding semiautomaton, is somewhat arbitrary in as far as it conforms to the standard convention that finite strings are scanned from left to right. If we were to scan strings from right to left, we would instead deal with the minimal automaton for  $\mathcal{L}(\rho)^r$  where  $r$  stands for reversal of strings. The sizes of these minimal automata, as well as their minimal Fischer automata differ in general. Hence, it might be more interesting to study some measure of complexity that is invariant under reversal. For example, the size of the 0-minimal ideal in the syntactic semigroup of the language might be used, see [18] and [2]. Fortuitously, for the cellular automata constructed below that exhibit exponential blow-up, the two values coincide.

### Example: Binary CAs of Width 3

To specify a CA without having to give an explicit table for the local map  $\rho : \Sigma_k^w \rightarrow \Sigma_k$ , we can order the words in  $\Sigma_k^w$  lexicographically, and represent  $\rho$  by the corresponding *label sequence*. Even more convenient is to code  $\rho$  as a non-negative integer whose  $k$ -ary expansion (padded to  $k^w$  digits), is the label sequence of the CA. For example, CA  $\langle 150, 3, 2 \rangle$  corresponds to addition of three cells modulo 2 (or, as a Boolean function, to the exclusive or of the three cells). A  $(w, k)$ -CA is a CA over alphabet  $\Sigma_k$  with width  $w$ . Hence, the code numbers of  $(w, k)$ -CAs range from 0 to  $k^{k^w} - 1$ .

Since we are interested in the size of the minimal automaton and the minimal Fischer automaton for  $L = \mathcal{L}(\rho)$  rather than the CA itself, we can exploit symmetries to reduce the number of CAs to be considered. First, exchanging labels 0 and 1 affects neither  $\mu(L)$  nor  $\mu_F(L)$ . Second, we can apply an automorphism of the underlying de Bruijn graph without changing these two parameters. For binary CAs, there is exactly one non-trivial such automorphism and, in terms of the label sequence, this automorphism corresponds to reversal. Hence, every binary local map  $\rho$  is associated with maps

$\mu(L)$	$\mu_F(L)$	(3, 2)-CA number
1	1	15, 30, 45, 51, 60, 85, 86, 89, 90, 102, 105
2	1	0
3	2	4, 12, 24, 34, 66
4	2	46
4	3	2, 3, 8, 11, 14, 17, 19, 28, 36, 42, 50, 70, 81, 126
5	3	38
5	4	1, 7, 10, 21, 29, 35, 49
6	3	18
6	5	13, 25, 33, 44, 62, 110
7	5	69
7	6	61
10	8	43, 54, 113
10	9	5, 9, 20, 65
11	7	53
11	8	58
11	9	27, 78
11	10	6
12	10	23, 57, 77
14	13	26, 74
15	14	41, 97
16	15	22, 37, 73, 94

Table 1: The sizes of the minimal automata and the minimal Fischer automata for all essential (3, 2)-CAs  $\rho$ .

$\rho^c$ ,  $\rho^r$  and  $\rho^{rc}$ . The set  $\{\rho, \rho^c, \rho^r, \rho^{rc}\}$  is the equivalence class of local map  $\rho$ , and  $\rho$  is *essential* if it is the minimal element of its class (with respect to the lexical order of label sequences). By Burnside's lemma, there are  $1/4(2^N + 2^{N/2} + 2^{N/2}) \approx 1/4 \cdot 2^N$  essential binary maps of width  $w$  where  $N := 2^w$ . For example, there are 72 essential binary CAs of width 3. Note that standard topological conjugacy is a different equivalence relation; e.g., there are 84 conjugacy classes of binary CAs of width 3.

Table 1 lists the sizes of the minimal automata and the minimal Fischer automata for all 72 essential binary CAs of width 3. As one can see from the table, there are 4 essential (3, 2)-CAs with maximum blow-up, corresponding to a total of 16 (3, 2)-CAs.

## Exponential Blow-up

We will now tackle the problem of constructing binary cellular automata  $\rho$  whose associated minimal Fischer automata have maximal size. To this end, we will study a class of semiautomata that is slightly more general than the de Bruijn automata that we are ultimately interested in. These semiautomata are characterized by being (2, 2)-regular and transitive. Here, (2, 2)-regular means

that every node in the diagram of the automaton has in-degree as well as out-degree 2. Note that all these diagrams admit permutation automata: there are labelings of the diagram such that the resulting automaton is both deterministic and codeterministic.

It is easy to determine the number of all permutation labelings. For suppose we have a  $(2, 2)$ -regular graph  $G$ . Define a *zigzag* in  $G$  to be an alternating path of the form

$$x_1 \rightarrow x_2 \leftarrow x_3 \rightarrow \dots \leftarrow x_{n-1} \rightarrow x_n \leftarrow x_1$$

where all edges are distinct (but the vertices  $x_1, \dots, x_n$  need not be distinct). It is easy to show that every edge in  $G$  belongs to exactly one zigzag. Thus, we can decompose the edge set of  $G$  into a collection of zigzags. Also note that the restriction of any permutation labeling to a zigzag is determined by its value on any one edge of the zigzag. Hence, the number of permutation labelings is  $2^\zeta$  where  $\zeta$  is the number of zigzags in  $G$ .

At any rate, for a  $(2, 2)$ -regular and transitive semiautomaton  $M$  we can define the *Hamming distance*  $\text{HD}(M)$  to the nearest permutation automaton to be the least number of changes in the labeling of the automaton that need to be made in order to turn the automaton into a permutation automaton. The Hamming distance of a cellular automaton  $\rho$  will always be understood to be  $\text{HD}(B(\rho))$ . For example, the global map of a CA of Hamming distance 0 is always surjective, and in fact open, and therefore has a trivial Fischer automaton. More interesting for us are *1-permutation automata*, which have Hamming distance 1 to the nearest permutation automaton. As we will show, a large collection of these automata exhibit full exponential blow-up.

We begin with a combinatorial lemma about a monoid of maps from the power set of a finite set to itself. Consider a finite set  $Q$  and let  $\mathbb{Q} := \text{pow}(Q)$  be the power set of  $Q$ . For the sake of brevity, we will refer to maps  $f : \mathbb{Q} \rightarrow \mathbb{Q}$  as *operators*. Thus, the set of all operators forms a monoid under composition and operators act naturally on  $\mathbb{Q}$ . The next lemma describes two submonoids of this operator monoid that will be important in our proof of exponential blow-up later. The submonoids are generated by three special operators, the *cyclic shift*  $\sigma$ , the *deletion operator*  $\kappa$ , and the *branching operator*  $\beta$ . More precisely, these operators are defined as follows. Let  $\sigma$  be a permutation of  $Q$  and let  $C \subseteq Q$  be one of its cycles; we will refer to  $C$  as the *base cycle*. Fix an arbitrary point  $q_0$  on  $C$ , the *base point*. Define  $\kappa(A) := A - \{q_0\}$ , and  $\beta(A) = \sigma(A)$  if  $q_0 \notin A$ ,  $\beta(A) = \sigma(A) \cup \{q_0\}$ , otherwise. For subsets  $X, Y$  and  $P$  of  $Q$ , we write  $X =_P Y$  to indicate that  $X$  and  $Y$  agree on  $P$ :  $X \cap P = Y \cap P$ . We write composition in diagrammatical form, so that  $f \circ g(X)$  or  $fg(X)$  is to be interpreted as  $g(f(X))$ .

**Lemma 3.2** *Let  $\mathbb{T}_0$  be the operator monoid generated by  $\sigma$  and  $\kappa$  and let  $\mathbb{T}_1$  be the monoid generated by  $\kappa$  and  $\beta$ . Then the following hold.*

- For all sets  $B \subseteq C \subseteq A$ , there is an operator  $f$  in  $\mathbb{T}_0$  such that  $f(A) =_C B$  and  $f(A) =_{Q-C} A$ .

- For all sets  $B \subseteq C$  and  $A \cap C \neq \emptyset$ , there is an operator  $g$  in  $\mathbb{T}_1$  such that  $f(A) =_C B$  and  $f(A) =_{Q-C} A$ .

*Proof.* Let  $m$  be a multiple of the order of  $\sigma$  as an element of the permutation group on  $Q$  and let  $n$  be the length of the base cycle.

First note that the second claim follows from the first. To see this, observe that the shift operator is in  $\mathbb{T}_1$ :  $\sigma(A)$  is either  $\beta(A)$  or  $\beta\kappa(A)$ . But  $C \subseteq \beta^{2m}(A)$  and  $\beta^{2m}(A) =_{Q-C} A$ , so we can compose  $\beta^{2m}$  with a map in  $\mathbb{T}_0 \subseteq \mathbb{T}_1$  to obtain the desired set  $B$  inside of the base cycle.

To simplify notation, let us write the base cycle as  $C = \{q_0, q_1, \dots, q_{n-1}\}$ . The index  $i$  in  $q_i$  is understood to be taken modulo  $n$ . Now consider an operator of the form

$$f = \sigma^{m-n} \circ \tau_0 \circ \sigma \circ \tau_{-1} \circ \dots \circ \sigma \circ \tau_{1-n} \circ \sigma$$

where  $\tau_i$  is  $\kappa$  if  $q_i \notin B$ , and the identity operator otherwise. Then  $f \in \mathbb{T}_0$  and, since  $C \subseteq A$ , it is easy to see that  $f(A) =_C B$ . Furthermore, since  $\kappa$  is the identity outside of  $C$  and the order of  $\sigma$  divides  $m$ , we have  $f(A) =_{Q-C} \sigma^m(A) =_{Q-C} A$ .  $\square$

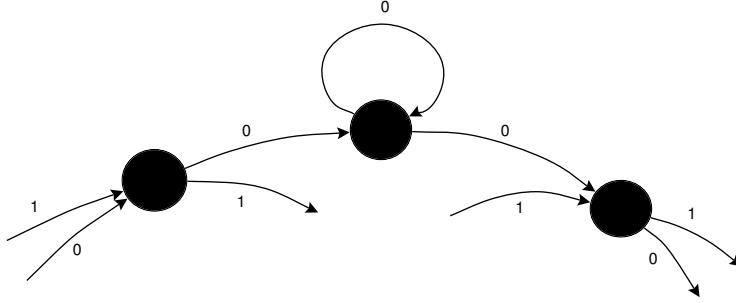
We are interested in operators  $\mathbb{Q} \rightarrow \mathbb{Q}$  that arise from the transition relation of a finite state machine on  $Q$ . More precisely, given a word  $x$ , there is the obvious operator  $[x](A) := A \cdot x$ . For example, if the finite state machine is a minimal automaton, these operators form the syntactic semigroup of the acceptance language of the machine, see, e.g., [18]. However, for our purposes we need a more general notion of operators derived from a transition relation. To this end, define an operator  $f : \mathbb{Q} \rightarrow \mathbb{Q}$  to be *representable* if for all  $A \subseteq Q$  there is a word  $x$  such that  $f(A) = A \cdot x$ . Note that we do not require the same word  $x$  to work for all  $A$ , thus a representable operator will in general not be of the form  $[x]$ . Clearly, the set of representable operators is closed under composition. Hence all operators in  $\mathbb{T}_0$  are representable whenever  $\sigma$  and  $\kappa$  are representable, and likewise for  $\mathbb{T}_1$ ,  $\kappa$  and  $\beta$ .

Now suppose  $M$  is a 1-permutation automaton. The transition that has to be relabeled in order to obtain a permutation automaton will be called the *flaw* of the automaton. The next theorem shows that if the flaw occurs at a self-loop, then the corresponding power automaton has size  $2^n$ . We will later show that this machine is also minimal.

**Theorem 3.2**

Let  $M$  be a binary, transitive, 1-permutation automaton on  $n$  states whose flaw occurs at a self-loop. Then the power automaton of  $M$  has  $2^n$  states.

*Proof.* We may safely assume that  $n \geq 2$ . Denote by  $q_0$  the source and target vertex of the flaw. By relabeling, we may assume that the zigzag containing  $q_0$  has the following form:



Decompose the state set  $Q$  of the semiautomaton  $M$  into strongly connected components of the subgraph induced by the edges labeled 0. These components partition the state set and form disjoint cycles except for one, which has an additional self-loop at  $q_0$ . The cycle through  $q_0$  will be called the *main cycle*, and  $q_0$  the *anchor point*. Likewise, the components of the subgraph induced by the edges labeled 1 form a collection of disjoint cycles, except for the trivial component  $\{q_0\}$ . We will refer to these cycles as 0-cycles and 1-cycles, respectively. Let  $m_0$  be the least common multiple of the lengths of all 0-cycles and define  $m_1$  similarly for 1-cycles.

For any state  $p \in Q$ , define the *rank* of  $p$  to be the least integer  $i \geq 0$  such that there is a word  $x$  of the form  $0^*(1+0^*)^i$  with  $q \in q_0 \cdot x$ . In other words, the rank of  $p$  is least number of blocks of consecutive 1's labeling any path from  $q_0$  to  $p$ . Since  $M$  is transitive, the rank is defined for each state; all states on the base cycle  $C_0$  have rank 0, states on any 1-cycle  $D$  intersecting  $C_0$  have rank at most 1, all states on any 0-cycle  $C \neq C_0$  intersecting  $D$  have rank exactly 1, and so forth. Let  $Q_s$  be the set of all states of rank  $s$ . We will write  $Q_{<s}$  for all states of rank less than  $s$ , and  $Q_{\leq s}$  for all states of rank at most  $s$ , and so forth.

Lastly, let us say that a subset  $P$  of  $Q$  is *controllable* if for all sets  $X, Y \subseteq Q$  such that  $Y \subseteq X \cap P$  there is a representable operator  $f$  such that

$$f(X) =_P Y \text{ and } f(X) =_{Q-P} X.$$

The crux of our argument is to show by induction on the rank  $s$  that  $Q_{\leq s}$  is controllable. The claim of the theorem then follows easily by setting  $X = P = Q$ .

Again we use the same terminology as in lemma 3.2.

*Claim 1:* The main cycle  $C_0$  is controllable.

By the first part of lemma 3.2, it suffices to show that operators  $\sigma$  and  $\kappa$  on base cycle  $C_0$  and base point  $q_0$  are representable. To see this, note that  $\kappa = [1^{m_1}]$ ,  $\sigma(X) = X \cdot 0$ , if  $q_0 \notin X$  or  $q_{-1} \in X$ , and  $\sigma(X) = X \cdot 01^{m_1}$  otherwise.

*Claim 2:* Assume that  $Q_{<s}$  is controllable and let  $D_1, \dots, D_r$  be the 1-cycles intersecting  $Q_{<s}$  that contain states of rank  $s$ . Then  $Q_{<s} \cup D_1 \cup \dots \cup D_r$  is also controllable.

Since all the 1-cycles are disjoint it suffices to establish our claim for just one such cycle, say,  $D$ . Pick a state  $q \in D$  of rank less than  $s$ . We will use  $q$  as the base point and  $D$  as the base cycle. The deletion operator  $\kappa$  is representable by our hypothesis. The shift operator here is simply represented by  $[1]$ , since  $D$  does not contain the anchor point  $q_0$ .

*Claim 3:* Assume that  $Q_{<s}$  is controllable where  $s \geq 1$ . Then  $Q_{\leq s}$  is also controllable.

All the states in  $Q_\sigma$  either lie on one of the 1-cycles dealt with in claim 2, or they lie on a 0-cycle that intersects one of these 1-cycles. Again, by disjointness, it suffices to consider a single such 0-cycle  $C$ . Pick a state  $q \in C$  that lies on a 1-cycle  $D$  that in turn intersects  $Q_{<s}$ . If we choose  $C$  as the base cycle and  $q$  as the base point, it follows from our assumption and claim 2 that the deletion operator  $\kappa$  is representable. The shift operator  $\sigma$  has the same representation as for the main cycle:  $\sigma(X) = X \cdot 0$  or  $\sigma(X) = X \cdot 01^{m_1}$ , depending on  $X$ .

It follows from claims 1 and 3 that  $Q_{\leq s}$  is controllable for all ranks  $s$ . Since  $Q$  is finite, this means that  $Q$  itself is controllable. Hence, for any  $A \subseteq Q$  there is a representable operator  $f$  such that  $f(Q) = A$  and the power automaton of  $M$  must have  $2^n$  states.  $\square$

Thus, any nondeterministic semiautomaton  $M$  satisfying the conditions of the theorem exhibits full exponential blow-up during deterministic simulation. For a set  $A \subseteq Q$  the minimal word (with respect to the product order length/lexical)  $x$  such that  $Q \cdot x = A$  is the witness for  $A$ . The upper bounds for the length of a witness derivable from the proof are significantly larger than their actual lengths. For example, for the (4, 2)-CAs listed in table 2 below, the longest witness has length 18.

### Theorem 3.3

*Let  $M$  be a binary, transitive, 1-permutation automaton whose flaw occurs at a self-loop. Then the power automaton of  $M$  is the minimal automaton.*

*Proof.* Let  $n$  be the number of states of  $M$  and  $M'$  its power automaton. By the last theorem,  $M'$  has cardinality  $2^n$  and all subsets of  $Q$ , the state set of  $M$ , occur as states in  $M'$ . Hence we have to show that for all  $A \neq B \subseteq Q$ , the behavior of  $A$  is different from the behavior of  $B$ .

*Claim:* Let  $p \in Q$ ,  $B \subseteq Q$  where  $p \notin B$ . Then for some  $x \in \Sigma^*$ :  $p \cdot x \neq \emptyset$  but  $B \cdot x = \emptyset$ .

First note that it suffices to show that for some word  $z$ :  $p \cdot z \not\subseteq B \cdot z$  and  $|B \cdot z| < |B|$ .

This is straightforward if  $B \cap C_0$  is not empty: the shift and deletion operators on the main cycle  $C_0$  were shown to be representable in the last theorem, thus we can rotate  $B$  until it contains the anchor point and then delete the anchor point in the next step. Since the cyclic shift is a permutation, the non-inclusion condition is automatically satisfied.

To deal with the general situation, define the *corank* of a state  $q \in Q$  to be the least number  $i$  such that there is a word  $x \in 0^*(1^+0^*)^i$  with  $q_0 \in q \cdot x$ . The corank of  $B$  is the minimum corank of its elements. Hence, we may assume that  $B$  has positive corank  $r$ . Note that for some suitable number  $j$  we must have  $p \cdot 0^j \cap B \cdot 0^j = \emptyset$  and  $B \cdot 0^j$  contains a state on a 1-cycle that also contains at least one state of corank less than  $r$ . Hence, for some integer  $l$ ,  $B' := B \cdot 0^j 1^l$  will contain a state of rank less than  $r$ . However, it may be the case that  $p \cdot 0^j = q_0$  and therefore  $p \cdot 0^j 1^l = \emptyset \subseteq B'$ . In this situation, we can apply operator  $f := [0^j 0^{m_0} 1^l]$ : then  $f(p) \cap f(B) = \emptyset$  but  $f(p) \neq \emptyset$ , and, of course, the corank of  $f(B)$  is less than  $r$ . Repeating this procedure, we can reduce the corank of  $B$  to 0, and we have already seen how to handle this case.  $\square$

Note that no claim was made as to the transitivity of the partial minimal automaton in the last theorem. Indeed, there are cases where this machine fails to be transitive. Consider, for example, the transitive semiautomaton  $M$  on state set  $Q = \{0, 1, \dots, 6\}$  whose cycles labeled 0 are  $(0, 1, 2, 3)$  and  $(4, 5, 6)$ , and the cycles labeled 1 are  $(1, 4)$ ,  $(2, 5)$ , and  $(3, 6)$ . The self-loop labeled 0 is at state  $q_0 = 0$ . The partial minimal automaton has two strongly connected components: one of size 22 and one of size 105, the former being the minimal Fischer automaton for  $M$ . In the special case of de Bruijn automata, however, the minimal Fischer automata have maximum size  $2^n - 1$ .

### Lemma 3.3

*Let  $M$  be a binary, 1-permutation de Bruijn automaton  $M$  whose flaw occurs at a self-loop. Then the corresponding minimal Fischer automaton has size  $2^n - 1$  where  $n = 2^{w-1}$  is the size of  $M$ .*

*Proof.* From the last theorem and lemma 3.1, it suffices to show that we have  $q_0 \cdot x = Q$  for some word  $x$ . Consider the zigzag  $Z$  that includes the self-loop at  $q_0$ . Three of the edges in  $Z$  are labeled 0, the fourth is labeled 1 and is a cord in the main cycle  $C_0$ . In the diagram of theorem 3.2, this would correspond to having an edge from  $q_{-1}$  to  $q_1$  labeled 1:

$$Z : \quad q_{-1} \xrightarrow{0} q_0 \xleftarrow{0} q_0 \xrightarrow{0} q_1 \xleftarrow{1} q_{-1}.$$

Inductively define words

$$\begin{aligned} z_0 &:= 0^{2m_0}, \\ z_{i+1} &:= ((z_i 1)^{m_1} 0)^{m_0}, \end{aligned}$$

and consider the corresponding operators  $[z_i]$ .

*Claim:* For any  $A \subseteq Q$  disjoint from  $C_0$ , and all ranks  $s$ :

$$[z_s](A) =_{Q_{\leq s}} Q_{\leq s} \quad \text{and} \quad [z_s](A) =_{Q_{> s}} A.$$

We proceed by induction on rank  $s$ . The case  $s = 0$  is obvious, so suppose  $s > 0$ . By induction hypothesis,  $Q_{\leq s} \subseteq [z_s](A)$ . But then  $A' := A \cdot z_s 1 z_s$  contains  $Q_{\leq s}$ : the cord from  $q_{-1}$  to  $q_1$  is labeled 1, so  $A \cdot z_s 1 \cap C_0 \neq \emptyset$ . Moreover, since  $[z_s]$  restricted to  $Q_{> s}$  is the identity,  $A' \cap Q_{> s}$  consists of all points  $q$  of rank  $s + 1$  that can be reached from a point of rank  $s$  by one edge labeled 1. Now let  $A'' := A \cdot (z_s 1)^{m_1}$ . Then  $A''$  consists of  $Q_{< s}$  plus all 1-cycles that intersect  $Q_{< s}$ . Also, the operator induced by  $(z_s 1)^{m_1}$  is the identity outside of  $A''$ . Similarly,  $A \cdot (z_s 1)^{m_1} 0 (z_s 1)^{m_1}$  consists of all points in  $A''$  plus those that can be reached from  $A''$  by on edge labeled 0. Repeating this extension procedure, we ultimately obtain all of  $Q_{\leq s+1}$ .

It follows that the minimal automaton consists of a large strongly connected component of size  $2^n - 1$  plus the sink. By theorem 3.1, the minimal Fischer automaton is isomorphic to this large component, and we are done.  $\square$

We note in passing that the condition of having a cord labeled 1 in the main cycle that was used in the last theorem is sufficient but by no means necessary to produce a Fischer automaton of maximal size. Consider the 1-cycles  $D_1, \dots, D_r$  that intersect the main cycle. Decompose  $D_i$  into paths  $D_{i,j}$  such that the source and target of  $D_{i,j}$  lie on the main cycle but all interior points fail to do so. Let  $d_{i,j}$  be the length of  $D_{i,j}$ . Then the argument of the last theorem can also be carried out as long as the  $d_{i,j}$  are relatively prime. Unfortunately, this condition also fails to be necessary.

## 4 The Complexity of $\mathcal{L}(\rho)$

We now turn to the languages associated with cellular automata and their de Bruijn automata. As indicated previously, we will use the same terminology for cellular automata as for finite state machines and their acceptance languages. Thus, a CA  $\rho$  is a permutation automaton if  $B(\rho)$  is a permutation automaton,  $\mu(\rho)$  stands for  $\mu(\mathcal{L}(\rho))$ , and so forth. Permutation CAs are trivially surjective and even open: their global maps are exactly  $k^{w-1}$ -to-1, see [4]. The reference also

5482	5737	6502	6757	9562	9817	10582	10837
17818	18073	18854	19109	20902	21157	21673	21994
22249	22693	22918	22958	22966	23014	23173	23201
23462	23974	25241	25753	26014	26249	26257	27094

Table 2: The 32 essential  $(4, 2)$ -CAs with complexity 256.

$\Delta$	0	1	2	4	8	16	32	64
freq.	4096	512	896	480	240	208	328	352
$\Delta$	124	128	170	256	512	1024	1052	2048
freq.	8	296	8	224	160	120	8	256

Table 3: Complexity of 1-permutation  $(5, 2)$ -CAs.

shows that a cellular automaton  $\rho$  is a permutation automaton iff there is a bilaterally transitive configuration with  $k^{w-1}$  preimages under  $\rho_\infty$ .

Because of the regularity of the de Bruijn graphs underlying the semiautomata  $B(\rho)$  it is easy to count permutation automata. First, the underlying graph for a binary cellular automaton of width  $w$  is  $B(w-1, 2)$ . There are  $N := 2^w$  edges in  $B(w-1, 2)$  and every zigzag in  $B(w-1, 2)$  is of the form

$$ax \rightarrow xb \leftarrow \bar{a}x \rightarrow \bar{b}x \leftarrow ax$$

(here  $\bar{s}$  stands for the binary complement,  $s \in \Sigma_2$ ). Thus, every zigzag in  $B(w-1, 2)$  has length 4. If we disregard the degenerate case  $B(1, 2)$ , a zigzag can either be a *diamond* (4 nodes and 4 ordinary edges), or a *triangle* (3 nodes, 3 ordinary edges and a loop). There are  $2^{N/4}$  binary permutation CAs of width  $w$ . Hence, there are  $N 2^{N/4}$  binary 1-permutation CAs of width  $w$ . A binary de Bruijn graph has two self-loops, thus there are  $2 \cdot 2^{N/4}$  1-permutation CAs with flaw at a loop. By theorem 3.3, there are at least that many binary CAs  $\rho$  of width  $w$  such that  $\mu(\rho) = 2^{2^{w-1}}$ . Experimental results show there are in fact many other 1-permutation CAs with maximum complexity. Moreover, all 1-permutation CAs exhibit some sort of exponential blow-up, though they may not reach the upper bound  $2^{2^{w-1}}$ .

Let us first consider binary CAs of width 4. There are  $2^{16} = 65536$  such automata, 16 permutation automata, and 256 1-permutation automata. Half of those have maximum complexity  $\mu(\rho) = 2^8 = 256$  and are listed in table 2. To save space, we actually only enumerate the 32 essential maps with full blow-up. The remaining ones are all conjugates  $\rho^c$ ,  $\rho^r$ , or  $\rho^{rc}$  where  $\rho$  is a map from the table. For all these 128 maps we also have  $\mu_F(\rho) = 255$ . Hence, the minimal Fischer automaton is identical with the partial minimal DFA, construed as a semiautomaton. Of course, there are only 32 1-permutation automata with flaw at a loop.

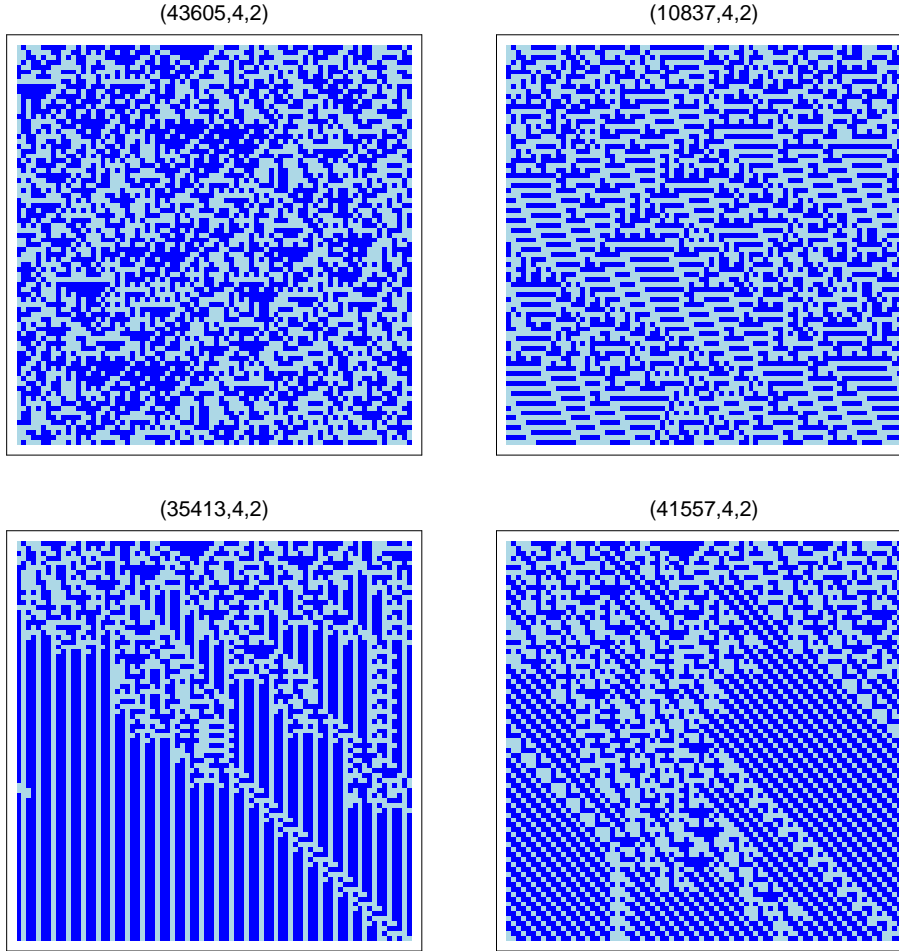


Figure 2: Evolution of a random configuration on the permutation automaton  $\langle 43605, 4, 2 \rangle$ , and three 1-permutation variants with rule numbers 10837, 35413, and 41557.

For binary CAs of width 5, there are 256 permutation automata, and correspondingly  $2^{13} = 8192$  1-permutation automata. Again, half of these have maximum complexity  $2^{16}$ . All the others still have complexity close to the maximum. Table 3 shows the frequencies of the occurring differences  $\Delta = 2^{16} - \mu(\rho)$ . As one can see, most of the differences are powers of 2 and are due to certain forbidden subsets of the form  $Q_0 \cup \text{pow}(Q_1)$  in the power automaton.

In figure 2, we plot the evolution of a random configuration on 4 binary cellular automata of width 4 (100 steps with periodic boundary conditions). The first CA is a permutation automaton, and the other three have Hamming distance 1 to the first. Their code numbers are 43605, 10837, 35413, and 41557, respectively. The corresponding  $\mu$  values are 1, 256, 255, and 255 and the  $\mu_F$  values are 1, 255, 254, and 254. As one can see, CA 43605 seems to preserve randomness of the initial configuration whereas the variants exhibit more complex behavior. The shortest excluded

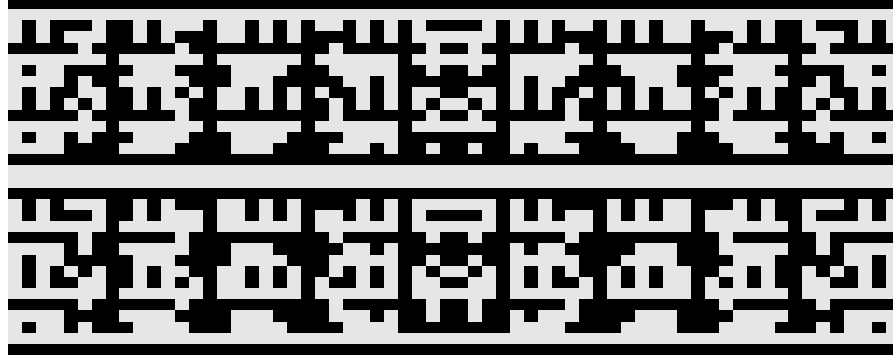


Figure 3: The complexities  $\mu(\rho)$  of  $32 \times 128$  1-permutation maps of width 5.

blocks are "100101010101001", "101011110111", and "111011110101", respectively, for the three 1-permutation automata.

The last figure 3 depicts the sizes of the minimal automata for all 1-permutation CAs. Since permutation labelings are symmetric with respect to the automorphism of the underlying de Bruijn graph, we can confine ourselves to 128 such labelings. The figure shows a 32 by 128 matrix where a filled square in position  $(i, j)$  indicates that switching the  $i$ -th edge in the  $j$ -th permutation labeling produces a machine of maximum complexity. The edges are enumerated in natural order, thus the top and bottom row correspond to the two self-loops. As one can see from the picture, there are two other edges that always produce maximum blow-up. These are the edges (0111, 1110) and (1000, 0001) and belong to the zigzags containing the loops. The blank rows correspond to the other two edges in these zigzags.

Thus the zigzags at the loops are special in that any permutation labeling can be modified to a labeling that exhibits full blow-up by switching either one of two edges in these zigzags (the loop or the edge not adjacent to the loop). A more careful analysis shows that indeed every one of the zigzags contributes 512 flawed labelings that cause full blow-up. However, for all other zigzags, the size of the corresponding minimal automaton depends not only on edge that was switched but also on the original permutation labeling.

### Iterated Maps

We now turn to the complexity of iterated automata. In [27], a table is given of the minimal automata sizes  $\mu(\rho^t)$  for all elementary cellular automata  $\rho$  and  $t \leq 5$ . For example, the first few iterates of rule 160, corresponding to logical and of the left and right neighbor, have minimal automata with sizes 9, 16, 25, 36, 49. As one can show, the partial minimal DFAs are in fact transitive, and one has  $\mu_F(\rho^t) = (t + 2)^2$ .

As pointed out in the introduction, it is easy to see that

$$\mu(\rho^t) \leq 2^{k^{t(w-1)}}$$

for all  $t \geq 1$ , and we have just seen that the bound is tight for  $t = 1$ . However, for  $t > 1$ , we will show that the iterated maps  $\rho^t$  cannot exhibit full exponential blow-up. The first step is to establish a weak converse to theorem 3.2.

**Proposition 4.1** *Suppose  $M$  is a binary  $r$ -permutation automaton of size  $n$  where  $r > 1$ . Then the power automaton of  $M$  has less than  $2^n$  states.*

*Proof.* Since  $M$  has Hamming distance larger than 1 to the nearest permutation automaton, there must be a zigzag in  $M$  with two flaws, or there have to be at least 2 zigzags with one flaw each. For the sake of brevity, we will only discuss the second case, the other is entirely analogous. First suppose both zigzags are diamonds, i.e., they contain no self-loops:

$$Z_i : s_1^i \rightarrow t_1^i \leftarrow s_2^i \rightarrow t_2^i \leftarrow s_1^i.$$

By exploiting the obvious symmetries, we may assume without loss of generality that the labeling of  $Z_1$  has its flaw at the edge  $s_1^1 \rightarrow t_1^1$  and that the first zigzag is labeled thus:

$$Z_1 : s_1^1 \xrightarrow{0} t_1^1 \xleftarrow{0} s_2^1 \xrightarrow{1} t_2^1 \xleftarrow{0} s_1^1.$$

Likewise, the second zigzag may be assumed to be labeled

$$Z_2 : s_1^2 \xrightarrow{0} t_1^2 \xleftarrow{0} s_2^2 \xrightarrow{1} t_2^2 \xleftarrow{0} s_1^2,$$

or

$$Z_2 : s_1^2 \xrightarrow{1} t_1^2 \xleftarrow{0} s_2^2 \xrightarrow{1} t_2^2 \xleftarrow{1} s_1^2.$$

In either case,  $P = \{t_2^1, t_1^2\}$  has no predecessor.

If one or both of the zigzags contain self-loops, the last argument can easily be adjusted: we can think of a self-loop as being obtained by merging two nodes in a diamond, and this merging process does not affect the existence of a predecessor for  $P$ .  $\square$

It follows that all local maps that exhibit full blow-up are in fact 1-permutation automata.

**Proposition 4.2** *Let  $\rho$  be a binary rule of width  $w$  that fails to be deterministic (or codeterministic). Then all the iterates  $\rho^t$ ,  $t \geq 2$ , are  $r$ -permutation rules where  $r \geq 2^{(t-1)(w-1)}$ .*

*Proof.* Suppose without loss of generality that  $\rho(x0) = \rho(x1)$  for some word  $x$ . Then for any word  $y$  of appropriate length we have  $\rho^t(yx0) = \rho^t(yx1)$ . Since the width of  $\rho^t$  is  $t(w-1) + 1$ ,  $y$  has length  $(t-1)(w-1)$ , and our claim follows.  $\square$

From the last two propositions it follows that full exponential blow-up cannot occur for the iterates of any binary rule.

**Lemma 4.1** *Let  $\rho$  be a binary rule. Then none of the iterates  $\rho^t$ ,  $t \geq 2$ , exhibits full exponential blow-up.*

The Fischer automata associated with the languages  $\mathcal{L}(\rho^t)$ ,  $t \geq 2$ , therefore all have size less than  $2^{2^t(w-1)} - 1$ .

## Open Problems

We conclude by stating a few open problems. It is clear from the experimental results for binary CA's of widths 3, 4, 5, that theorem 3.3 and lemma 3.3 cover only a small fraction of those local maps  $\rho$  for which  $\mu(\rho)$  is maximal. In fact, the results make it tempting to formulate the following conjecture.

### Conjecture

For all widths  $w$ , there are  $2^{w-1}2^{2^{w-2}}$  binary local maps  $\rho$  such that

$$\mu(\rho) = \mu_F(\rho) + 1 = 2^{2^{w-1}}.$$

It seems possible that the methods used when the flaw occurs at a self-loop can be modified to cover the case where the flaw is at the other critical edge in a triangle. It is not clear how to handle any of the other edges.

In general, there appears to be a much stronger correlation between  $\mu(\rho)$  and the Hamming distance of  $\rho$  to a permutation labeling: the larger the Hamming distance, the smaller the corresponding power automaton. Since the Hamming distance of an iterate  $\rho^t$  to a permutation automaton decreases exponentially, the last lemma could be strengthened considerably.

Similar comments apply to the size of the Fischer automaton. We also do not know how large a gap between  $\mu(\rho)$  and  $\mu_F(\rho)$  can occur for automata with large Hamming distances.

Lastly, there is the question of the computational cost of determining  $\mu(\rho)$  and  $\mu_F(\rho)$ . As we have seen, the brute force approach based on actual construction of the minimal automaton may require exponential space. In general, it is PSPACE-hard to compute the size of the minimal automaton of a given semiautomaton, see [22]. In fact, it is even hard to compute the size of the accessible part of the power automaton of a given semiautomaton. However, the hardness proof given in the reference does not carry over to the de Bruijn automata under consideration here. Thus, we are not aware of any lower bounds for the computation of  $\mu(\rho)$  or  $\mu_F(\rho)$ .

## Acknowledgments

It is a pleasure to acknowledge many helpful discussions with Dave Landaeta. The computational results mentioned in this paper were obtained using `automata`, a hybrid package of *Mathematica* and C++ programs written by the author. See <http://www.cs.cmu.edu/~sutner> for more information.

## References

- [1] S. Amoroso and Y. N. Patt. Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures. *Journal of Computer and Systems Sciences*, 6:448–464, 1972.
- [2] D. Beauquier. Minimal automaton for a factorial, transitive, rational language. *Theoretical Computer Science*, 67:65–73, 1989.
- [3] R. Fischer. Sofic systems and graphs. *Monatshefte für Mathematik*, 80:179–186, 1975.
- [4] G. A. Hedlund. Endomorphisms and automorphisms of the shift dynamical system. *Math. Systems Theory*, 3:320–375, 1969.
- [5] W. M. L. Holcombe. *Algebraic Automata Theory*. Cambridge University Press, 1982.
- [6] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.
- [7] L. Hurd. Formal language characterizations of cellular automata limit sets. *Complex Systems*, 1(1):69–80, 1987.
- [8] K. Culik II, L. P. Hurd, and S. Yu. Formal languages and global cellular automata behavior. *Physica D*, 45:396–403, 1990.
- [9] K. Culik II and Sheng Yu. Undecidability of CA classification schemes. *Complex Systems*, 2(2):177–190, 1988.
- [10] K. Culik II and Sheng Yu. Cellular automata,  $\omega\omega$ -regular sets, and sofic systems. *Discrete Applied Mathematics*, 32:85–101, 1991.
- [11] C. G. Langton. Studying artificial life with cellular automata. *Physica D*, 22:120–149, 1986.
- [12] C. G. Langton. Computation at the edge of chaos. *Physica D*, 42:12–37, 1990.

- [13] C. G. Langton. Life at the edge of chaos. In C. G. Langton, editor, *Artificial Life II*, pages 41–91. Santa Fe Institute, 1991.
- [14] W. Li and N. Packard. Structure of elementary cellular automata rule space. *Complex Systems*, 4(3):281–298, 1990.
- [15] M. Mitchel, J. P. Crutchfield, and P. T. Hraber. Dynamics, computation, and the “edge of chaos”: a re-examination. Technical Report 93-06-040, Santa Fe Institute, New Mexico, USA 87501, 1993.
- [16] M. Nasu. Homomorphisms of graphs and their global maps. In N. Saito and T. Nishiziki, editors, *Graph theory and algorithms*, pages 159–170. Springer Verlag, LNCS 108, 1981.
- [17] D. Perrin. Finite automata. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 2–57. Elsevier, 1990.
- [18] J. E. Pin. *Varieties of Formal Languages*. Foundations of Computer Science. Plenum Publishing Corporation, 1986.
- [19] K. Sutner. A note on Culik-Yu classes. *Complex Systems*, 3(1):107–115, 1989.
- [20] K. Sutner. Classifying circular cellular automata. *Physica D*, 45(1–3):386–395, 1990.
- [21] K. Sutner. De Bruijn graphs and linear cellular automata. *Complex Systems*, 5(1):19–30, 1991.
- [22] K. Sutner. The size of power automata. In J. Sgall, Ales Pultr, and Petr Kolman, editors, *Mathematical Foundations of Computer Science*, volume 2136 of *SLNCS*, pages 666–677, 2001.
- [23] B. Weiss. Subshifts of finite type and sofic systems. *Monatshefte für Mathematik*, 77:462–474, 1973.
- [24] S. Wolfram. Computation theory of cellular automata. *Comm. Math. Physics*, 96(1):15–57, 1984.
- [25] S. Wolfram. Universality and complexity in cellular automata. *Physica D*, 10:1–35, 1984.
- [26] S. Wolfram. Twenty problems in the theory of cellular automata. *Physica Scripta*, T9:170–183, 1985.
- [27] S. Wolfram. *Theory and Applications of Cellular Automata*. World Scientific, 1986.