Complex Systems 4 (1990) 131–138

Nonrecursive Cellular Automata Invariant Sets

Lyman P. Hurd
Lab for Plasma Research, University of Maryland,
College Park, MD 20742, USA

Abstract. Languages corresponding to two invariant subshifts, the limit set and periodic set are examined. The complement of the language produced by a cellular automaton limit set is always recursively enumerable (r.e.), and modulo intersection with a regular language and ε-limited homomorphism, all languages with r.e. complements arise this way. The language produced by the periodic set is always r.e.; the closure of the set of languages produced by all cellular automata is the set of all r.e. languages. As a corollary, a specific cellular automaton F is produced whose limit language is not r.e. (although its complement is), and a rule G whose periodic points give rise to a language which is r.e. but not recursive.

Introduction

Cellular automata can be thought of as abstract continuous maps on a compact metric space, or as a formal computational system which acts on doubly infinite strings from a finite alphabet. Applying the techniques of the theory of computation and formal languages to questions which are essentially dynamical in nature was first suggested by Wolfram in [22]. Formal definitions and proofs of several conjectures in that paper appeared in [10]. Since this time several papers have appeared with this general framework (see [3, 4]).

This paper focuses on the specific question of the complexity of the language produced by the limit set and periodic points of an arbitrary cellular automaton. More general results relating subshifts and languages can be found in [4, 6]. A survey of results applying the theory of computation to cellular automata can be found in [5].

Cellular Automata

The full shift on k symbols, $S^{\mathbf{Z}}$, is the set of functions from the integers to a set of k elements, S. Equivalently $S^{\mathbf{Z}}$ is the set of doubly infinite words in the symbols from S. The full shift is topologized by giving S the discrete topology and $S^{\mathbf{Z}}$ the product topology, or equivalently by the metric:

$$d(x,y) = \sum_{i=-\infty}^{\infty} \delta(x_i, y_i) 2^{-|i|}$$

where $\delta(a,b) = 0$ if $a = b$ and 1 otherwise.

The full shift is acted upon by a homeomorphism σ (the left shift map) defined by $\sigma(x)_i = x_{i+1}$. A closed, shift-invariant subset of $S^{\mathbf{Z}}$ is referred to as a *subshift*.

Definition 1. *A cellular automaton is a continuous function $F : S^{\mathbf{Z}} \to S^{\mathbf{Z}}$ that commutes with σ.*

Cellular automata are specified by specifying an arbitrary *local* function

$$f : S^{2r+1} \to S$$

and defining the *global* function $F : S^{\mathbf{Z}} \to S^{\mathbf{Z}}$ by:

$$F(x)_i = f(x_{i-r}, \ldots, x_{i+r})$$

The integer $r > 0$ is called the *radius* of the cellular automaton rule. All shift-invariant continuous maps arise from such an f (see [8]). It is important to observe that all cellular automata are completely described by a finite amount of information (the rule table of f).

The Limit Set

Early studies of cellular automata (see [7, 16]) focused on so-called Garden-of-Eden configurations, those configurations not in the image of F. This construction can be generalized to configurations that appear in the image of F^n but not in the image of F^{n+1}. However, from the point of view of dynamical systems it is more natural to look at the complement of these sets, configurations that can arise at any time step.

The limit set, $\Lambda(F)$, is the intersection of all forward images of $S^{\mathbf{Z}}$. Equivalently, it is the set of all $x \in S^{\mathbf{Z}}$ such that $F^{-n}(x) \neq \emptyset$ for all $n > 0$.

Definition 2.

$$\Lambda(F) = \bigcap_{i=0}^{\infty} F^i(S^{\mathbf{Z}})$$

Since $S^{\mathbf{Z}}$ is compact, and F is continuous, $\Lambda(F)$ is a nonempty subshift. Also, $\Lambda(F) = S^{\mathbf{Z}}$ if and only if F is surjective. If $Y \subseteq S^{\mathbf{Z}}$ and $F(Y) = Y$ then $Y \subseteq \Lambda(F)$. The limit set is the maximal F-invariant subset of $S^{\mathbf{Z}}$.

The limit set has the property that all configurations get closer to it uniformly. In other words:

$$d(F^n(S^{\mathbf{Z}}), \Lambda(F)) \to 0$$

as $n \to \infty$. The limit set is an attracting set in the sense of Conley (see [2]) and is maximal with respect to this property.

The Periodic Set

In analyzing the dynamics of a map, it is useful to study the set of points periodic under F. Since this set is not generally closed, one considers the set, $\Pi(F)$ that is the closure of the periodic points, which is a subshift. This set always satisfies $\Pi(F) \subseteq \Lambda(F)$[1].

Definition 3.

$$\mathrm{Per}(F) = \{x \in S^{\mathbf{Z}} \textit{ such that } F^p(x) = x \textit{ for some } p > 0\}$$

The closure of this set is denoted $\Pi(F)$.

Subshifts and Languages

Subshifts give rise to languages in a natural way. In fact there is some interest in studying the class of languages which arise from subshifts (see [1, 4, 17, 18, 19]. These languages are characterized completely in [11, 6]. For the purposes of this paper, only a few definitions will be necessary.

Definition 4. *The cylinder set,* $\mathrm{Cyl}(s)$, *associated with a string* $s = s_1 \dots s_m$ ($s_i \in S$) *is the set of* $c \in S^{\mathbf{Z}}$ *such that* $c_i = s_i$ *for* $-[(n-1)/2] \leq i \leq [n/2]$ *(where* $[x]$ *represents the greatest integer in* x*).*

These sets are not subshifts since they are not shift-invariant, but they are open and closed, and form a basis for the topology of the shift.

Definition 5. *The language associated with a subshift* $K \subseteq S^{\mathbf{Z}}$, *denoted* $\mathcal{L}(K) \subseteq S^*$ *consists of all finite strings* s *such that* $\mathrm{Cyl}(s) \cap K \neq \emptyset$.

Equivalently $\mathcal{L}(K)$ is the set of finite blocks which occur somewhere in some configuration of K. The usefulness of this definition arises from the fact that the set of finite strings completely determines the subshift.

Lemma 1. *If* $K_1, K_2 \subseteq S^{\mathbf{Z}}$ *are subshifts,* $\mathcal{L}(K_1) = \mathcal{L}(K_2)$ *implies that* $K_1 = K_2$.

Proof: This lemma was stated in [21] and proved in [10]. ∎

Statement of Results

The following theorems outline the possible complexity of limit sets and periodic points for a cellular automaton.

These theorems were motivated by a series of conjectures made by Wolfram in [22]. The construction used to prove theorem 3 was introduced in [10] and was motivated by a construction in [7].

The first result relating cellular automaton dynamics and language theory, was proved by Weiss [21], here restated in the notation of this paper.

[1]It is also the case that $\Pi(F) \subseteq \Omega(F) \subseteq \Lambda(F)$ where $\Omega(F)$ is the non-wandering set of F. Each of these inequalities may be strict (see [12]).

Theorem 1 (Weiss) *For every cellular automata $F : S^{\mathbf{Z}} \to S^{\mathbf{Z}}$, the language generated by its finite time image, $\mathcal{L}(F^i(S^{\mathbf{Z}}))$, is regular.*

The limit set can be much more complicated. A series of examples of varying complexity can be found in [13]. In all cases the complexity is bounded above by the constraint that the complement of the limit set must be recursively enumerable.

Theorem 2. *For every cellular automata $F : S^{\mathbf{Z}} \to S^{\mathbf{Z}}$, the complement of the limit language, $S^* - \mathcal{L}(\Lambda(F))$, is recursively enumerable.*

Proof: If a string s is in the complement of the limit language, it must be in the complement of $\mathcal{L}(F^n(S^{\mathbf{Z}}))$ for some $n > 0$. The strings of length m allowed by the n^{th} time-step may be computed by calculating the image under F^n of all strings of length $m + 2nr$. For each n determining whether s is in $S^* - \mathcal{L}(F^n(S^{\mathbf{Z}}))$ requires finite time. If s is in $S^* - \mathcal{L}(\Lambda(F))$ this procedure will halt. ∎

Several of the theorems in this section depend on using cellular automata to simulate Turing machines. The particular embedding used here was introduced in [10]. In that paper it is shown that this particular simulation has the important property that the cellular automaton shows the presence of a head in a given state at time n if and only if there is a valid Turing machine computation which would yield this result. Without enumerating all problems, essentially one wants to eliminate spurious transitions, and particularly the chance that the cellular automaton is simulating two interfering Turing machine heads.

A Turing Machine $M = \{Q, A, \alpha, \beta, \gamma\}$ is given by specifying a finite set of states, Q, a finite set of symbols, A, and transition functions that at any time consider the current state and the symbol on the square occupied by the head and determine a new state ($\alpha(q, a) \in Q$), a new symbol to write into the current square ($\beta(q, a) \in A$), and the direction for the head to move ($\gamma(q, a) \in \{\pm 1\}$).

Cellular automata operate on a homogeneous space, so in theory each square has the potential to hold a head. This problem is dealt with by having an alarm condition ($), and have each blank square record whether it is to the left or the right of the head. The rule corresponding to a given Turing Machine is reproduced in table 1.

Given this construction, the statement that a given configuration can occur as the arbitrary time image of the dynamics of a Turing machine, suffices to show that it is in the limit set of the cellular automaton. Furthermore if a string in the limit set includes a head state, then the configuration can occur as the arbitrary image of a Turing machine.

Theorem 3. *If $L \subseteq A^*$ is a language whose complement is recursively enumerable, there exists a cellular automaton $F : S^{\mathbf{Z}} \to S^{\mathbf{Z}}$, a regular language $R \subseteq S^*$, and an (ε-limited) homomorphism $\phi : S^* \to A^*$ such that:*

$$\phi(\mathcal{L}(\Lambda(F)) \cap R) = L$$

$S = \{r, l, q_1, ..., q_n\} \times \{a_1, ..., a_m\} \cup \{\$\}$			
	$(r, a_i)(r, a_j)(r, a_k)$	$\to$	(r, a_j)
	$(l, a_i)(l, a_j)(l, a_k)$	$\to$	(l, a_j)
$\gamma(q_n, a_i) = 1$	$(q_n, a_i)(r, a_j)(r, a_k)$	$\to$	$(\alpha(q_n, a_i), a_j)$
	$(l, a_i)(q_n, a_j)(r, a_k)$	$\to$	$(l, \beta(q_n, a_j))$
$\gamma(q_n, a_i) = -1$	$(l, a_i)(l, a_j)(q_n, a_k)$	$\to$	$(\alpha(q_n, a_k), a_j)$
	$(l, a_i)(q_n, a_j)(r, a_k)$	$\to$	$(r, \beta(q_n, a_j))$
	otherwise	$\to$	$\$$

Table 1: The Rule F_M associated with a Turing Machine M.

Proof: By assumption on L, there is a Turing machine, M, which halts precisely when a string *fails* to be in L. Define a new language $L' \subseteq (A \cup \{x\})^*$ which has the same alphabet as L and one additional symbol x, according to the following rules:

1. All strings in L' have the form $a_1 \dots a_n x \dots x$.

2. A string $a_1 \dots a_n x^m \in L'$ if and only if M started on a tape containing the string $a_1 \dots a_n$ will take at least m steps without halting.

By construction, M fails to halt precisely on the strings in L. Hence $a_1 \dots a_n \in L$ is equivalent to the statement $a_1 \dots a_n x^m \in L'$ for all $m > 0$. Furthermore, L' has been constructed so that there exists a Turing machine which will recognize strings without leaving the area of tape on which the string is written (such a machine is called a *linear bounded automaton* (see [9]). Note that if the size of the portion of the tape which the head visits does not change in forward or backward time, the Turing machine (and the cellular automaton) must be in a periodic loop.

Construct a Turing Machine M' that examines $w_l a_1 \dots a_n x \dots x w_r$. If the string between w_l and w_r is in L', M' overwrites the symbol a_n with w_r moves to the beginning of the string and starts the computation over again. This situation is the only case in which M' can re-enter the start state.

Under no circumstances does the Turing Machine leave the portion of the tape between the markers, nor does it ever move the left-most marker. Furthermore, the right-hand marker may only be moved to the left.

A simple induction shows that M' can be in the start state on a tape reading $w_l a_1 \dots a_n w_r$ after an arbitrary number of time steps, if and only if $a_1 \dots a_n \in L$. Similarly defining the cellular automaton corresponding to M' as above yields the result that the string $(w_l, l)(a_1, q_s)(a_2, r) \dots (a_n, r)(w_r, r)$ is in the limit language of F if and only if $a_1 \dots a_n \in L$. All such strings may be arrived at by intersection with a regular language, and a homomorphism maps into the alphabet A yielding the theorem. ∎

As a corollary one has the following theorem a proof of which was sketched in [10] and completed independently in [3].

Corollary 1. *There exists a cellular automaton $F_u : S^{\mathbf{Z}} \to S^{\mathbf{Z}}$ such that $\mathcal{L}(\Lambda(F_u)) \subseteq S^*$ is not a recursively enumerable language.*

Proof: One form of Gödel's Incompleteness theorem (see [15]) is the existence of a language that is recursively enumerable, but whose complement is not. The classical example is the language accepted by a Universal Turing Machine.

Choose a language L that is *not* recursively enumerable but whose complement *is*. Form the cellular automaton F_u, regular language, and homomorphism guaranteed by the previous theorem. Since the class of recursively enumerable languages is closed under the operations of homomorphism and intersection with a regular language, the fact that L is not recursively enumerable implies that $\mathcal{L}(\Lambda(F_u))$ is not recursively enumerable. ∎

Theorem 4. *If F is any cellular automaton rule, $\mathcal{L}(\Pi(F))$ is recursively enumerable.*

Proof: For any fixed period, p, blocks occurring in configurations of period p form a regular language (see [22]). Thus for each length n and period p, a finite calculation produces all strings of length n occurring in points of period p. ∎

Theorem 5. *If $L \subseteq A^*$ is any recursively enumerable language, there exists a cellular automaton rule $F : S^{\mathbf{Z}} \to S^{\mathbf{Z}}$, a regular language $R \subseteq S^*$, and a homomorphism $\phi : S^* \to A^*$ such that:*

$$\phi(\mathcal{L}(\Pi(F)) \cap R) = L$$

Proof: The proof of this result is similar to, but somewhat simpler than that of theorem 3. Let M be a Turing Machine recognizing L. Construct a new Turing Machine, M' that takes strings of the form $w_l a_1 \ldots a_n w_r$ and if M halts on $a_1 \ldots a_n$, then M' restores the tape to its initial state, and starts the computation over again. In this case the left boundary is fixed, but the right boundary is allowed to move to the right or left as required by the computation. The only constraint is that M' must always keep track of the initial state, so that it can return it. Once again, passing to a cellular automaton simulating M' gives the desired result. ∎

Once again an immediate corollary of the preceding theorem is the existence of a cellular automaton whose periodic set is recursively enumerable but not recursive. Again it should be noted that the proof of this fact can be made constructive.

Corollary 2. *There exists a cellular automaton $F : S^{\mathbf{Z}} \to S^{\mathbf{Z}}$ such that $\mathcal{L}(\Pi(F_u)) \subseteq S^*$ is a recursively enumerable language but its complement is not (i.e., it is r.e. but not recursive).*

Discussion

Corollaries 1 and 2 are proved by obtaining lower bounds on the complexity of certain cellular automaton invariant sets; this method of proof is not useful for giving a complete specification of a limit set. In [13] complementary results are given illustrating cellular automata with recursive limit languages of varying degrees of complexity.

The theorems here also concern the complexity of a single cellular automaton's invariant sets. Another type of result concerns collective properties of the class of cellular automata. A strong theorem in this direction has been proven by Kari [14] who is able to show that every non-trivial proposition about a cellular automaton limit set (including the proposition that it has a single element) is undecidable.

Acknowledgments

I am grateful for discussions with Karel Culik, Mike Hurley, Jarkko Kari, John Milnor, and Sheng Yu. This work was supported by DARPA under the ACMP program.

References

[1] L. Boasson and M. Nivat, "Adherences of languages," *Journal of Computer System Sciences*, **20** (1980) 285–309.

[2] C. Conley, *Isolated Invariant Sets and the Morse Index*, CBMS Regional Conference Series **38**, Amer. Math. Soc., Providence, RI, 1978.

[3] K. Culik II, J. Pachl, and S. Yu, "On the limit sets of cellular automata," *SIAM J. Comput.*, **18:4** (1989) 831–842.

[4] K. Culik II and S. Yu, "Cellular automata, $\omega\omega$-regular sets, and sofic systems," *Discrete Applied Mathematics*, to appear.

[5] K. Culik II, L.P. Hurd, and S. Yu, "Computation theoretic aspects of cellular automata," *Physica D*, submitted.

[6] K. Culik II, L.P. Hurd, and S. Yu, "Formal language aspects of global cellular automaton behavior," *Physica D*, submitted.

[7] U. Golze, "Differences between 1- and 2-dimensional cell spaces," in *Automata, Languages, Development*, A. Lindenmayer and G. Rozenberg, eds. (North-Holland, 1976); and "Endliche, Periodische, und Rekursive Zellulare Konfigurationen: Vorgangerberechnung und Garten-Eden-Probleme," dissertation to the Technischen Universitat Hannover.

[8] G.A. Hedlund, "Transformations commuting with the shift," in *Topological Dynamics*, J. Auslander and W.G. Gottschalk, eds. (Benjamin, 1968) 259–289; and "Endomorphisms and automorphisms of the shift dynamical system," *Math. Syst. Theor.*, **3** (1969) 320–375.

[9] J.E. Hopcroft and J.D. Ullman, *Introduction to Automata Theory, Languages, and Computation* (Addison-Wesley, 1979).

[10] L.P. Hurd, "Formal language characterizations of cellular automaton limit sets," *Complex Systems*, **1** (1987) 69–80.

[11] L.P. Hurd, "The application of formal language theory to the dynamical behavior of cellular automata," doctoral thesis, Department of Mathematics, Princeton University, June 1988.

[12] L.P. Hurd, "The non-wandering set of a CA map," *Complex Systems*, **2** (1988) 549–554.

[13] L.P. Hurd, "Recursive Cellular Automata Invariant Sets" *Complex Systems* to appear.

[14] J. Kari, "Rice's theorem for limit sets of cellular automata", in preparation.

[15] M. Minsky, *Computation: Finite and Infinite Machines* (Prentice-Hall, 1967).

[16] E.F. Moore, *Machine Models of Self-Reproduction*, Proc. Sympos. Appl. Math., **14**, Amer. Math. Soc., Providence, RI (1962) 17–33.

[17] M. Nivat, "Sur les ensembles de mots infinis engendrés par une grammaire algébrique," *RAIRO Informatique Théorique*, **12** (1978) 259–278.

[18] M. Nivat and D. Perrin, "Ensembles reconnaissables des mots biinfinis," *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing* (1982) 47–59.

[19] A. Restivo, "Finitely generated sofic systems," Research Report, Dipartimento di Matematica e Applicazioni, Universita di Palermo (1986).

[20] A.R. Smith III, "Simple computation-universal cellular spaces," *J. ACM*, **18** (1971) 339.

[21] B. Weiss, "Subshifts of finite type and sofic systems," *Monatshefte fur Mathematik*, **77** (1973) 462–474.

[22] S. Wolfram, "Computation theory of cellular automata," *Commun. Math. Phys.*, **96** (1984) 15–57.