Johann Wolfgang Goethe-Universität
Frankfurt am Main

Institut für Informatik
Fachbereich Biologie und Informatik

Descriptional Complexity of Cellular Automata and Decidability Questions

Andreas Malcher

Nr. 4/01

Frankfurter Informatik-Berichte

Institut für Informatik • Robert-Mayer-Straße 11-15 • 60054 Frankfurt am Main

Descriptional Complexity of Cellular Automata and Decidability Questions

Andreas Malcher
Institut für Informatik, Johann Wolfgang Goethe-Universität
D-60054 Frankfurt am Main, Germany
E-Mail: malcher@psc.informatik.uni-frankfurt.de

Abstract

We study the descriptional complexity of cellular automata (CA), a parallel model of computation. We show that between one of the simplest cellular models, the realtime-OCA, and "classical" models like deterministic finite automata (DFA) or pushdown automata (PDA), there will be savings concerning the size of description not bounded by any recursive function, a so-called nonrecursive trade-off. Furthermore, nonrecursive trade-offs are shown between some restricted classes of cellular automata. The set of valid computations of a Turing machine can be recognized by a realtime-OCA. This implies that many decidability questions are not even semidecidable for cellular automata. There is no pumping lemma and no minimization algorithm for cellular automata.

1 Introduction

Given a grammar or automata model, in the theory of formal languages one investigates for example the generative capacity, closure properties or decidability questions of the model. Furthermore, questions concerning the descriptional complexity arise. How succinctly can a model represent a formal language in comparison with other models? Regarding regular languages, it is known [9] that there are languages being recognized by a nondeterministic FA (NFA) with n states, such that every DFA recognizing these languages will need 2^n states. Beyond this trade-off bounded by an exponential function, Hartmanis has proved that between deterministic PDA (DPDA) and PDA there exists a trade-off not bounded by any recursive function, a so-called nonrecursive trade-off. Additional nonrecursive trade-offs are known to exist between DPDA and unambiguous PDA (UPDA), between UPDA and PDA and many other models.

The models considered so far have in common that they process their input in a sequential manner. There are also parallel computational models, among others cellular automata.

A cellular automaton consists of many identical deterministic finite automata (cells) arranged in a line. The next state of a cell depends on the current state of the cell and the current states of a bounded number of neighboring cells. The transition rule is applied synchronously to each cell at the same time. One simple model is the realtime one-way cellular automaton (realtime-OCA). Here the local transition rule depends only on the state of the cell and the neighboring cell to the right. Furthermore, the input is processed in realtime. We will define cellular automata (CA) and the recognition of formal languages by CA in the next section.

The intention of this paper is to investigate the descriptional complexity of cellular automata in comparison with classical automata models and several subclasses of cellular automata. This goal is attained rather easily due to the fact that the set of valid computations of a Turing machine can be recognized by a realtime-OCA. This allows us to use techniques as presented in the paper by Hartmanis [4]. We can show nonrecursive trade-offs between DFA and realtime-OCA, PDA and realtime-OCA and between realtime-OCA and realtime-CA. The recognition of the set of valid computations by CA has some interesting consequences: "Almost nothing" is decidable for CA, there is no pumping lemma for CA languages and there is no minimization algorithm.

2 Preliminaries and Definitions

Let Σ^* denote the set of all words over the finite alphabet Σ, $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$. By $|w|$ we denote the length of a string w, and the reversal of a word w is denoted by w^R. Let REG, LCF, CF, RE denote the families of regular, linear context-free, context-free and recursively enumerable languages. In this paper we do not distinguish whether a language L contains the empty word ϵ or not. I.e.: We identify L with $L \setminus \{\epsilon\}$. We assume that the reader is familiar with the common notions of formal language theory as presented in [5]. Let S be a set of recursively enumerable languages. Then S is said to be a property of the recursively enumerable languages. A set L has the property S, if $L \in S$. Let L_S be the set $\{< M > \mid T(M) \in S\}$ where $< M >$ is an encoding of a Turing machine M. If L_S is recursive, we say the property S is decidable; if L_S is recursively enumerable, we say the property S is semidecidable. Concerning cellular automata we largely follow the notations and definitions as introduced in [7].

Definition: A two-way cellular automaton (CA) A is a quintuple $A = (Q, \#, \Sigma, \delta, F)$, where

1. $Q \neq \emptyset$ is the finite set of cell states,
2. $\# \notin Q$ is the boundary state,
3. $\Sigma \subseteq Q$ is the input alphabet,
4. $F \subseteq Q$ the set of accepting cell states and
5. $\delta : (Q \cup \{\#\}) \times (Q \cup \{\#\}) \times (Q \cup \{\#\}) \to Q$ is the local transition function.

Restricting the flow of information only from the right to the left, we get an one-way cellular automaton (OCA) and the local transition function maps from $(Q \cup \{\#\}) \times (Q \cup \{\#\})$ to Q. To simplify matters we identify the cells by positive integers.

A configuration of a cellular automaton at some time step $t \geq 0$ is a description of its global state, formally a mapping $c_t : \{1, \dots, n\} \to Q$ for $n \in \mathbb{N}$. The initial configuration at time 0 is defined by the input word $w = x_1 \dots x_n$: $c_{0,w}(i) = x_i$, $1 \leq i \leq n$.

During a computation the O(CA) steps through a sequence of configurations whereby successor configurations are computed according to the global transition function Δ: Let c_t, $t \geq 0$, be a configuration, then its successor configuration is defined as follows:

$$\begin{aligned} c_{t+1} &= \Delta(c_t) \iff \\ & c_{t+1}(1) = \delta(\#, c_t(1), c_t(2)) \\ & c_{t+1}(i) = \delta(c_t(i-1), c_t(i), c_t(i+1)), i \in \{2, \ldots, n-1\} \\ & c_{t+1}(n) = \delta(c_t(n-1), c_t(n), \#) \end{aligned}$$

for CAs and

$$\begin{aligned} c_{t+1} &= \Delta(c_t) \iff \\ & c_{t+1}(i) = \delta(c_t(i), c_t(i+1)), i \in \{1, \ldots, n-1\} \\ & c_{t+1}(n) = \delta(c_t(n), \#) \end{aligned}$$

for OCAs. Thus, Δ is induced by δ.

An input string w is accepted by an (O)CA if at some time step i during its computation the leftmost cell enters an accepting state from the set of accepting states $F \subseteq Q$.

Definition: Let $A = (Q, \#, \Sigma, \delta, F)$ be an (O)CA.

1. A word $w \in \Sigma^+$ is accepted by A if there exists a time step $i \in \mathbb{N}$ such that $c_i(1) \in F$ holds for the configuration $c_i = \Delta^i(c_{0,w})$.
2. $T(A) = \{w \in \Sigma^+ \mid w \text{ is accepted by } A\}$ is the language accepted by A.
3. Let $t : \mathbb{N} \to \mathbb{N}$, $t(n) \geq n$, be a mapping and i_w be the minimal time step at which A accepts $w \in T(A)$. If all $w \in T(A)$ are accepted within $i_w \leq t(|w|)$ time steps, then $T(A)$ is said to be of time complexity t.
4. $\mathcal{L}_t(\text{OCA}) = \{L \subseteq \Sigma^* \mid L \text{ is accepted by an OCA with time complexity } t\}$
   $\mathcal{L}_t(\text{CA}) = \{L \subseteq \Sigma^* \mid L \text{ is accepted by a CA with time complexity } t\}$
5. If $t(n) = n$, we say these languages are accepted in realtime; if $t(n) = k \cdot n$ with a rational number $k \geq 1$, we say these languages are accepted in lineartime. The corresponding language classes are denoted by $\mathcal{L}_{rt}(\text{OCA})$, $\mathcal{L}_{rt}(\text{CA})$, $\mathcal{L}_{lt}(\text{OCA})$ and $\mathcal{L}_{lt}(\text{CA})$, the corresponding cellular devices are denoted by realtime-OCA, realtime-CA, lineartime-OCA and lineartime-CA.

It is known that REG $\subset$ LCF $\subset \mathcal{L}_{rt}(\text{OCA})$ and that CF and $\mathcal{L}_{rt}(\text{OCA})$ are incomparable [13]. $\mathcal{L}_{rt}(\text{OCA})$ is closed under union, intersection, complementation, reversal, and concatenation with regular sets [6]. $\mathcal{L}_{rt}(\text{CA})$ is closed under union, intersection, and complementation.

In the sequel we will use the set of valid computations of a Turing machine. Details are presented in [4] and [5].
The set of valid computations of a Turing machine M is denoted by VALC[M], the set of invalid computations is denoted by INVALC[M] $= \Lambda^* \setminus$ VALC[M] with respect to a coding alphabet Λ.

To show that some languages are not in $\mathcal{L}_{rt}(\text{OCA})$ we will apply the following pumping lemma for cyclic strings from [10].

Lemma 1 *For any $L \in \mathcal{L}_{rt}(OCA)$, there exists an integer n such that for any string w and any integer k, if $w^k \in L$ and $k > n^{|w|}$ then there is an integer $1 \leq m \leq n^{|w|}$ such that $w^{k+j \cdot m} \in L$ for all $j \geq 1$.*

Descriptional complexity

Concerning the notations and definitions of descriptional complexity we follow the presentation in [14]. A descriptional system K is a set of finite descriptors (e.g. automata or grammars) relating each $M \in K$ to a language $T(M)$. The language class being described by K is $T(K) = \{T(M) \mid M \in K\}$. For every language L we define $K(L) = \{M \in K \mid T(M) = L\}$. A complexity measure for K is a total function $|\cdot| : K \to \mathbb{N}$. Comparing two descriptional systems K_1 and K_2, we assume that $T(K_1) \cap T(K_2)$ is not finite. We say that a function $f : \mathbb{N} \to \mathbb{N}$, $f(n) \geq n$ is an *upper bound* for the trade-off when changing from a minimal description in K_1 for an arbitrary language to an equivalent minimal description in K_2, if for all $L \in T(K_1) \cap T(K_2)$ the following holds:

$$\min\{|M| \mid M \in K_2(L)\} \leq f(\min\{|M| \mid M \in K_1(L)\}).$$

If no recursive function is an upper bound for the trade-off between two descriptional systems K_1 and K_2, we say the trade-off is nonrecursive and write $K_1 \overset{nonrec}{\longrightarrow} K_2$.

3 Nonrecursive Trade-Offs

Theorem 1 *Let M be a Turing machine. Then two realtime-OCA A_1, A_2 can be constructed such that $T(A_1)$ = VALC[M] and $T(A_2)$ = INVALC[M].*

Proof:

In [5] it is shown that INVALC[M] is a context-free language. Taking a close look at the construction we can see that INVALC[M] is the union of regular and linear context-free languages. Therefore we can construct a linear context-free grammar G such that $L(G)$ = INVALC[M]. Given a linear context-free grammar G, Smith III has shown in [12] how to construct a realtime-OCA A such that $T(A) = L(G)$. So, we can construct a realtime-OCA A_2 such that $T(A_2)$ = INVALC[M] $\in \mathcal{L}_{rt}$(OCA). Since $\mathcal{L}_{rt}$(OCA) is effectively closed under complementation, we can construct a realtime-OCA A_1 such that $T(A_1)$ = VALC[M] $\in \mathcal{L}_{rt}(OCA)$. □

Corollary: A language L is recursively enumerable if and only if there exists a homomorphism h and a language $L' \in \mathcal{L}_{rt}$(OCA) such that $L = h(L')$.

We are now prepared to prove some nonrecursive trade-offs using the techniques presented in [4].

Theorem 2 *Let K_1 and K_2 be two descriptional systems. If for every Turing machine M a language $L_M \in T(K_1)$ can be effectively constructed such that $L_M \in T(K_2) \Leftrightarrow T(M)$ is finite, then the trade-off between K_1 and K_2 is nonrecursive.*

Proof: We assume that the trade-off is not nonrecursive. Then there exists a recursive function f as an upper bound. I.e.: Let $L = T(A)$ for $A \in K_1$ and $L \in T(K_2)$, then there exists $M \in K_2$ such that $L = T(M)$ and $|M| \leq f(|A|)$. Assuming $A \in K_1$, we can list all $M_1, M_2, \ldots, M_s$, such that $|M_i| \leq f(|A|)$ for $1 \leq i \leq s$. But this implies: $T(A) \notin T(K_2) \iff T(A) \neq T(M_i)$ for all $1 \leq i \leq s$. By checking all inputs $x \in \Sigma^*$ on each M_i, a Turing machine can be constructed which stops when $T(A) \notin T(K_2)$. Thus, the set $R = \{A \mid A \in K_1, T(A) \notin T(K_2)\}$ is recursively enumerable. Hence we can construct a Turing machine M' which gets as input an encoding of a Turing machine. We construct L_M for input $< M >$ and check whether $L_M \in R$. Therefore, M' stops when $T(M)$ is infinite. Thus the set $\{M \mid M$ is a Turing machine and $T(M)$ is infinite$\}$ is recursively enumerable which is a contradiction to Rice's theorem for recursively enumerable index sets [5]. Hence the trade-off must have been nonrecursive. □

Consequences:

- realtime-OCA $\overset{nonrec}{\longrightarrow}$ DFA using $L_M = \text{INVALC}[M]$.
- realtime-OCA $\overset{nonrec}{\longrightarrow}$ PDA using $L_M = \text{VALC}[M]$.
- realtime-CA $\overset{nonrec}{\longrightarrow}$ realtime-OCA using $L_M = \text{L}_1[M]$ where $\text{L}_1[M] \overset{\text{def}}{=} \{w^{|w|!} \mid w \in \{\#_0\}\text{VALC}[M]\{\#_1\}\}$.
- lineartime-OCA $\overset{nonrec}{\longrightarrow}$ realtime-OCA using $L_M = \text{L}_2[M]$ where $\text{L}_2[M] \overset{\text{def}}{=} \{w^{|w|!} \mid w \in \{\#_1\}\text{VALC}[M]^R\{\#_0\}\}$.

The nonrecursive trade-offs just claimed are verified by the following lemma:

Lemma 2 *Let M be a Turing machine. Then*

(1) $INVALC[M] \in REG \Leftrightarrow T(M)$ is finite
(2) $VALC[M] \in CF \Leftrightarrow T(M)$ is finite
(3) $L_1[M] \in \mathcal{L}_{rt}(OCA) \Leftrightarrow T(M)$ is finite
(4) $L_1[M] \in \mathcal{L}_{rt}(CA)$.
(5) $L_2[M] \in \mathcal{L}_{rt}(OCA) \Leftrightarrow T(M)$ is finite
(6) $L_2[M] \in \mathcal{L}_{rt}(CA)$.

Proof: *(2)* is proved in [5] and *(1)* is then easy to show. The "if" portion of *(3)* is obvious, since REG is a subset of $\mathcal{L}_{rt}(\text{OCA})$. The "only if" portion is proved by using lemma 1. We show that $\text{L}_1[M] \notin \mathcal{L}_{rt}(\text{OCA})$, if $T(M)$ is infinite: We assume that $\text{L}_1[M] \in \mathcal{L}_{rt}(\text{OCA})$. Let $n \in \mathbb{N}$ be the integer from lemma 1. Since $T(M)$ is infinite, we can choose $w \in \{\#_0\}\text{VALC}[M]\{\#_1\}$ such that $|w|! > n^{|w|}$. Hence $w^{|w|!} \in \text{L}_1[M]$ and the conditions of the lemma are fulfilled. Therefore, an integer $1 \leq m \leq n^{|w|}$ does exist such that $w^{|w|!+j\cdot m} \in \text{L}_1[M]$ for all $j \in \mathbb{N}$. Considering $j = 1$, we have $w^{|w|!+m} \in \text{L}_1[M]$. But this is a contradiction, since $|w|! + m \leq |w|! + n^{|w|} < 2|w|! < (|w| + 1) \cdot |w|! = (|w| + 1)!$ is not a factorial and hence no $w' \in \{\#_0\}\text{VALC}[M]\{\#_1\}$ does exist such that $|w'|! = |w|! + m$.
To prove *(4)* we show how to construct a realtime-CA recognizing $\text{L}_1[M]$. $\text{L}_1[M]$ is the intersection of the following three languages L_1, L_2, L_3: Let $\Delta = \Sigma \cup \{\#_0, \#_1\}$ where $\{\#_0, \#_1\} \cap \Sigma = \emptyset$.

$$L_1 \overset{\text{def}}{=} \{\#_0 w \#_1 x \mid w \in \text{VALC}[M],\ x \in \Delta^*\},$$

$$L_2 \overset{\text{def}}{=} \{w^n \mid w \in \{\#_0\}\Sigma^*\{\#_1\}, n \in \mathbb{N}\},$$

$$L_3 \stackrel{\text{def}}{=} \{\#_0 w \#_1 x \mid w \in \Sigma^*, x \in \Delta^*, |\#_0 w \#_1 x|_{\#_0} = (|w|+2)!\}.$$

Since $\mathcal{L}_{rt}(\text{CA})$ is closed under intersection, it remains for us to show that $L_i \in \mathcal{L}_{rt}(\text{CA})$ for $1 \leq i \leq 3$. $L_1 \in \mathcal{L}_{rt}(\text{CA})$ is obvious, since VALC$[M] \in \mathcal{L}_{rt}(\text{OCA})$ and $\mathcal{L}_{rt}(\text{OCA})$ is closed under concatenation with regular sets. Considering the language

$$L \stackrel{\text{def}}{=} \{x \in \Delta^* \mid x = x_1 \#_0 x_2 \#_1 \#_0 x_3 \#_1 x_4 \Rightarrow x_2 \neq x_3 \text{where } x_2, x_3 \in \Sigma^*, x_1, x_4 \in \Delta^*\},$$

we see that $L_2 = \overline{L} \cap \{\{\#_0\}\Sigma^*\{\#_1\}\}^*$. Since $L \in \text{LCF} \subset \mathcal{L}_{rt}(\text{OCA})$ and $\mathcal{L}_{rt}(\text{OCA})$ is closed under intersection and complementation, it follows that $L_2 \in \mathcal{L}_{rt}(\text{OCA}) \subset \mathcal{L}_{rt}(\text{CA})$. Now it remains for us to show that $L_3 \in \mathcal{L}_{rt}(\text{CA})$. We sketch the construction: We use a cellular automaton where each cell is split into four subcells, so we can speak of four tracks. On the first track we are collecting all occurrences of $\#_0$'s from left to right. That means for an input containing the symbol $\#_0$ m times, that at some time step the first m cells are marked with a special symbol \$. This task can be done by a realtime-CA. The second track computes the factorials according to the construction presented in [8]. We modify the construction slightly: At one step the automaton is computing the factorials, in the next step all cells are shifted one cell to the right, in the next step the automaton is computing, and so on. Therefore, after $2 \cdot n!$ steps the $n!$-th cell from the left can be marked with a special symbol. Now, the task of the third track is to cooperate with the second track and to mark the $(|w|+2)!$-th cell from the left on the fourth track. This can be done by a realtime-CA within $2 \cdot (|w|+2)!$ steps. Now we just have to compare the number of occurrences of $\#_0$'s being collected in the first track with the marked cell on the fourth track: At some time step there is a cell on the first track having as left neighbor \$ and as right neighbor the endmarker symbol. The state of the cell itself is \$. The first time that this situation does arise, we look on the fourth track if this cell is marked. If this is true, we send a signal with maximum speed to the left to accept the input, otherwise we send a signal to reject the input. Hence we can construct a realtime-CA accepting L_3 and *(4)* is proved. The proof of *(5)* and *(6)* is analogous to *(3)* and *(4)* considering that VALC$[M]^R \in \mathcal{L}_{rt}(\text{OCA})$, since $\mathcal{L}_{rt}(\text{OCA})$ is closed under reversal. □

The nonrecursive trade-off between the descriptional systems K_1 and K_2 implies that there exists no algorithm converting a descriptor $M \in K_1$ into a descriptor $M' \in K_2$. I.e.: For regular and context-free languages there is no algorithm converting a realtime-OCA into an equivalent DFA and PDA, respectively. For realtime-OCA languages there is no algorithm converting a realtime-CA and lineartime-OCA into an equivalent realtime-OCA. An exceptional case are unary languages. It is known that each unary realtime-OCA language is a regular language and Seidel shows in [11] that for unary languages a realtime-OCA can be converted into an equivalent DFA. The trade-off is quadratic.

The following easy example shows that arbitrary recursive trade-offs can be constructed.

Example: Let f be a recursive function and $n \in \mathbb{N}$. Then there exists a regular language $L(f,n)$ being recognized by a realtime-OCA having $O(n)$ states, but every DFA recognizing $L(f,n)$ will need $\Omega(f(n))$ states.

Proof: Let f be a recursive function and $n \in \mathbb{N}$ a fixed number, then there exists a

Turing machine with unary input and output M which computes $f(n)$. Thus $L(f,n) =$ VALC$[M]$ consists of one string. This string can be recognized by a DFA which needs as many states as the string is long. Hence every DFA recognizing $L(f,n)$ will need $\Omega(f(n))$ states. Consider a Turing machine M' computing $f(n)$ on every input n. According to theorem 1, we can construct a realtime-OCA A recognizing VALC$[M']$. The size of A with respect to the length of input n is a constant number. If we want to modify A to recognize $L(f,n)$ for a fixed n, we just have to count the input length n. Hence a realtime-OCA recognizing $L(f,n)$ will need $O(n)$ states. □

4 Decidability Questions

Using reductions of the Post Correspondence Problem Seidel shows in [11] that the questions of theorem 3 and theorem 4 are not decidable. In [3] it is shown that the questions of emptiness, universality and equivalence are undecidable.
Due to the fact that the set of valid computations can be recognized by realtime-OCAs, we can simply prove that many decidability questions for cellular automata are not decidable and not even semidecidable. We want to summarize the known results in theorem 3 and 4, to present short proofs, and to show that the questions are not even semidecidable.

Lemma 3 *Let M be a Turing machine. Then it is not semidecidable whether $T(M) = \emptyset$, $T(M)$ is finite, $T(M)$ is infinite, $T(M)$ is regular, $T(M)$ is context-free or $T(M) \in \mathcal{L}_{rt}(OCA)$.*

Proof: Except for the property "$T(M)$ is infinite", all the properties are violating the containment property of Rice's theorem for recursively enumerable index sets [5]. We prove this for the last property by using $L = \{a^{2^n} \mid n \leq n_0\}$ for a fixed number $n_0 \in \mathbb{N}$ and $L' = \{a^{2^n} \mid n \in \mathbb{N}\}$, respectively. Since $L \in \mathcal{L}_{rt}(\text{OCA})$, $L \subsetneq L'$ and $L' \in$ RE, Rice's theorem implies that $L' \in \mathcal{L}_{rt}(\text{OCA})$. But this is a contradiction, since all unary languages in $\mathcal{L}_{rt}(\text{OCA})$ are regular languages. Hence the containment property is violated. The non-semidecidability of the property "$T(M)$ is infinite" can be seen by showing that the second condition of Rice's theorem for recursively enumerable index sets is not fulfilled. □

Theorem 3 *It is not semidecidable for arbitrary realtime-OCA A, A' whether*

- $T(A) = \emptyset$, $T(A) = \Sigma^*$
- $T(A)$ *is finite,* $T(A)$ *is infinite*
- $T(A) = T(A')$, $T(A) \subseteq T(A')$
- $T(A) \in REG$, $T(A) \in CF$

Proof: The technique of proving each statement is quite similar. For example, we prove that the question "Is $T(A)$ infinite?" is not semidecidable. Let M be an arbitrary Turing machine. By theorem 1 we can construct a realtime-OCA A accepting VALC$[M]$. Suppose that the above question is semidecidable. Thus it would be semidecidable whether $T(M)$ is infinite. This is a contradiction to lemma 3. □

Corollary: The above questions are not semidecidable for arbitrary automata A, A' which belong to an automata class containing the realtime-OCAs.

Theorem 4 *It is not semidecidable for arbitrary realtime-CA A whether $T(A) \in \mathcal{L}_{rt}(OCA)$.*

Proof: Let M be an arbitrary Turing machine. By lemma 2(4) we can construct a realtime-CA A accepting $L_1[M]$. Suppose that the above question is semidecidable. Then by lemma 2(3) it would be semidecidable whether $T(M)$ is finite. This is a contradiction to lemma 3. □

Corollary: The above question is not semidecidable for an arbitrary automaton A which belongs to an automata class containing the realtime-CAs.

Corollary: The above question is not semidecidable for $L \in \mathcal{L}_{rt}(CA)$ and each language class containing $\mathcal{L}_{rt}(CA)$.

Theorem 5 *Let A be a realtime-OCA, h a homomorphism and h_ϵ an ϵ-free homomorphism. Then it is not semidecidable whether*

- $h(T(A)) \in REG$, $h(T(A)) \in CF$, $h(T(A)) \in \mathcal{L}_{rt}(OCA)$
- $h_\epsilon(T(A)) \in REG$, $h_\epsilon(T(A)) \in CF$, $h_\epsilon(T(A)) \in \mathcal{L}_{rt}(OCA)$

Proof: Let M be a Turing machine. By the corollary to theorem 1 there is a realtime-OCA A and a homomorphism h such that $h(T(A)) = T(M)$. If the above questions are semidecidable, it is semidecidable whether $T(M)$ is regular, context-free or $T(M) \in \mathcal{L}_{rt}$(OCA). This is a contradiction to lemmma 3.
In [1] it is shown that the closure of $\mathcal{L}_{rt}$(OCA) under ϵ-free homomorphism yields $\mathcal{L}_{rt}$(1G-OCA) where 1G-OCA denotes one guess OCAs. Let A' be a realtime-1G-OCA. By [1] there is a realtime-OCA A and an ϵ-free homomorphism h_ϵ such that $T(A') = h_\epsilon(T(A))$. The assumption that the above questions are semidecidable implies that they are semidecidable for realtime-1G-OCAs. Since $\mathcal{L}_{rt}$(OCA) $\subset \mathcal{L}_{rt}$(1G-OCA) and $\mathcal{L}_{rt}$(CA) $\subseteq \mathcal{L}_{rt}$(1G-OCA) [1], this is a contradiction to the corollaries to theorem 3 and 4. □

Corollary: The above questions are not semidecidable for an arbitrary automaton A which belongs to an automata class containing the realtime-OCAs.

Corollary: The above questions are not semidecidable for $L \in \mathcal{L}_{rt}$(OCA) and each language class containing $\mathcal{L}_{rt}$(OCA).

Example: Automata classes containing the realtime-OCAs are lineartime-OCAs, realtime-CAs, and lineartime-CAs. Language classes containing $\mathcal{L}_{rt}$(OCA) are $\mathcal{L}_{lt}$(OCA), $\mathcal{L}_{rt}$(CA), and $\mathcal{L}_{lt}$(CA).

5 Further Results

Now, the results of the previous chapter can be applied to show that there is no pumping lemma and no minimization algorithm for cellular automata.
Following [2] we say that a language class $\mathcal{L}$ possesses a pumping lemma if the class has the following property: For each language $L \in \mathcal{L}$ there exists a number $n \in \mathbb{N}$ such that for each $z \in L$ with $|z| > n$, there is a partition $z = uvw$ such that $|v| \geq 1$ and for infinite many $i \in \mathbb{N}$ holds: $u'v^iw' \in L$, where u' and w' depend on u, w and i.

Theorem 6 *$\mathcal{L}_{rt}(OCA)$ and each language class containing $\mathcal{L}_{rt}(OCA)$ does not possess a pumping lemma in the above sense.*

Proof: Let A be an arbitrary realtime-OCA. On condition that a pumping lemma does exist, we show the following claim: $T(A)$ is infinite $\Leftrightarrow \exists x \in T(A) : |x| \geq n$. Hence we can semidecide whether A accepts an infinite language. This is a contradiction to theorem 3. Now we will prove the claim: The "only if" portion is obvious. "if": Let $x \in T(A)$ such that $|x| \geq n$. Since the conditions of the pumping lemma are fulfilled, we get infinite many words in $T(A)$ by pumping. □

Theorem 7 *For realtime-OCAs there is no minimization algorithm converting an arbitrary realtime-OCA A into a realtime-OCA A' which accepts $T(A)$ and has a minimal number of states.*

Proof: Obviously, a minimal realtime-OCA $A = (Q, \#, \Sigma, \delta, F)$ recognizing $L = \emptyset$ needs $|\Sigma|$ states and has no accepting states. We suppose that a minimization algorithm does exist. Let A be an arbitrary realtime-OCA. We apply the minimization algorithm and receive a minimal realtime-OCA A'. We are now checking whether A' has no accepting states and $|Q'| = |\Sigma|$. If it is so, then $T(A') = T(A) = \emptyset$. Otherwise, if $|Q'| = |\Sigma|$ and A' has accepting states, then at least one alphabet symbol is an accepting state. But then the recognized language is not empty. Hence we can decide whether an arbitrary realtime-OCA accepts the empty set. This is a contradiction to theorem 3. □

A consequence from the characterization of RE as the homomorphic image of $\mathcal{L}_{rt}$(OCA) is a criterion for incomparability to other language classes:

Theorem 8 *$\mathcal{L}_{rt}(OCA)$ is incomparable to each language class $\mathcal{L}$ satisfying: $CF \subseteq \mathcal{L} \subset RE$ and $\mathcal{L}$ is closed under homomorphism.*

Proof: According to Terrier [13], CF is not contained in $\mathcal{L}_{rt}$(OCA) and hence CF $\setminus$ $\mathcal{L}_{rt}$(OCA) $\neq \emptyset$. Therefore $\mathcal{L} \setminus \mathcal{L}_{rt}$(OCA) $\neq \emptyset$. Now we assume that $\mathcal{L}_{rt}$(OCA) $\subseteq \mathcal{L}$. According to the corollary to theorem 1, it follows that RE $\subseteq \mathcal{L}$. This is a contradiction to the assumption that $\mathcal{L}$ is a proper subset of RE. Hence we know that $\mathcal{L}_{rt}$(OCA)$\setminus\mathcal{L} \neq \emptyset$. □

By applying this criterion we can see that $\mathcal{L}_{rt}(\text{OCA})$ is incomparable to many known and well-investigated language classes. Among others there are the language classes generated by indexed grammars, certain grammars with controlled derivations, certain contextual grammars and certain L-systems, e.g. ET0L.

6 Conclusion

We have studied the descriptional complexity of cellular automata. Nonrecursive trade-offs were shown between sequential automata like DFA and PDA and cellular devices, namely the realtime-OCA. Even within cellular automata classes, nonrecursive trade-offs were proved. The fact that the valid computations of a Turing machine can be recognized by realtime-OCAs is a strong property of $\mathcal{L}_{rt}(\text{OCA})$, since this fact leads to nonrecursive trade-offs in a straightforward manner and almost no decidability results. Therefore, it would be interesting to investigate restricted classes of cellular automata, e.g. weaker models than realtime-OCA generating language classes between REG and $\mathcal{L}_{rt}(\text{OCA})$.

References

[1] T. Buchholz, A. Klein, M. Kutrib: "One guess one-way cellular arrays", In L. Brim, J. Gruska, J. Zlatuška (eds.): Mathematical Foundations of Computer Science 1998 (MFCS'98), volume 1450 of LNCS, 807-815, Springer, Berlin, 1998

[2] G. Buntrock: "Einige Bemerkungen zum Pumpen", In M. Kutrib, T. Worsch (Hrsg.): 5. Theorietag "Automatentheorie und Formale Sprachen", Bericht 9503, Arbeitsgruppe Informatik, Universität Gießen, Gießen, 1995

[3] K. Culik II, J. Gruska, A. Salomaa: "Systolic trellis automata: Stability, decidability and complexity", Information and Control, 71: 218-230, 1986

[4] J. Hartmanis: "On the succinctness of different representations of languages", In H. Maurer (ed.): International Colloquium on Automata, Languages and Programming 1979 (ICALP'79), volume 71 of LNCS, 282-288, Springer, Berlin, 1979

[5] J.E. Hopcroft, J.D. Ullman: "Introduction to Automata Theory, Languages and Computation", Addison-Wesley, Reading MA, 1979

[6] O.H. Ibarra, S.M. Kim: "Characterizations and computational complexity of systolic trellis automata", Theoretical Computer Science, 29: 123-153, 1984

[7] M. Kutrib: "Automata arrays and context-free languages", In C. Martin-Vide, V. Mitrana (eds.): "Where Mathematics, Computer Science, Linguistics and Biology Meet", 139-148, Kluwer Academic Publishers, Dordrecht, 2001

[8] J. Mazoyer, V. Terrier: "Signals in one-dimensional cellular automata", Theoretical Computer Science, 217: 53-80, 1999

[9] A.R. Meyer, M.J. Fischer: "Economy of descriptions by automata, grammars, and formal systems", IEEE Symposium on Foundations of Computer Science, 188-191, 1971

[10] K. Nakamura: "Real-time language recognition by one-way and two-way cellular automata", In M. Kutyłowski, L. Pacholski, T. Wierzbicki (eds.): Mathematical Foundations of Computer Science 1999 (MFCS'99), volume 1672 of LNCS, 220-230, Springer, Berlin, 1999

[11] S.R. Seidel: "Language recognition and the synchronization of cellular automata", Technical Report 79-02, Department of Computer Science, University of Iowa, Iowa City, 1979

[12] A.R. Smith III: "Cellular automata and formal languages", 11th Ann. IEEE Symposium on Switching and Automata Theory, 216-224, 1970

[13] V. Terrier: "On real time one-way cellular arrays", Theoretical Computer Science, 141: 331-335, 1995

[14] D. Wotschke: "Descriptional complexity", Lecture notes, J.W. Goethe-Universität Frankfurt, 1997

Interne Berichte am Fachbereich Informatik
Johann Wolfgang Goethe-Universität Frankfurt

1/1987 Risse, Thomas:
On the number of multiplications needed to evaluate the reliability of k-out-of-n systems

2/1987 Roll, Georg [u.a.]:
Ein Assoziativprozessor auf der Basis eines modularen vollparallelen Assoziativspeicherfeldes

3/1987 Waldschmidt, Klaus ; Roll, Georg:
Entwicklung von modularen Betriebssystemkernen für das ASSKO-Multi-Mikroprozessorsystem

4/1987 Workshop über Komplexitätstheorie, effiziente Algorithmen und Datenstrukturen:
3.2.1987, Universität Frankfurt/Main

5/1987 Seidl, Helmut:
Parameter-reduction of higher level grammars

6/1987 Kemp, Rainer:
On systems of additive weights of trees

7/1987 Kemp, Rainer:
Further results on leftist trees

8/1987 Seidl, Helmut:
The construction of minimal models

9/1987 Weber, Andreas ; Seidl, Helmut:
On finitely generated monoids of matrices with entries in N

10/1987 Seidl, Helmut:
Ambiguity for finite tree automata

1/1988 Weber, Andreas:
A decomposition theorem for finite-valued transducers and an application to the equivalence problem

2/1988 Roth, Peter:
A note on word chains and regular languages

3/1988 Kemp, Rainer:
Binary search trees for d-dimensional keys

4/1988 Dal Cin, Mario:
On explicit fault-tolerant, parallel programming

5/1988 Mayr, Ernst W.:
Parallel approximation algorithms

6/1988 Mayr, Ernst W.:
Membership in polynomial ideals over Q is expotential space complete

1/1989 Lutz, Joachim [u.a.]:
Parallelisierungskonzepte für ATTEMPO-2

2/1989 Lutz, Joachim [u.a.]:
Die Erweiterung der ATTEMPO-2 Laufzeitbibliothek

3/1989 Kemp, Rainer:
A One-to-one Correspondence between Two Classes of Ordered Trees

4/1989 Mayr, Ernst W. ; Plaxton, C. Greg:
Pipelined Parallel Prefix Computations, and Sorting on a Pipelined Hypercube

5/1989 Brause, Rüdiger:
Performance and Storage Requirements of Topology-conserving Maps for Robot Manipulator Control

6/1989 Roth, Peter:
Every Binary Pattern of Length Six is Avoidable on the Two-Letter Alphabet

7/1989 Mayr, Ernst W.:
Basic Parallel Algorithms in Graph Theory

8/1989 Brauer, Johannes:
A Memory Device for Sorting

1/1990 Vollmer, Heribert:
Subpolynomial Degrees in P and Minimal Pairs for L

2/1990 Lenz, Katja:
The Complexity of Boolean Functions in Bound Depth Circuits over Basis $\{\wedge, \oplus\}$

3/1990 Becker, Bernd ; Hahn R. ; Krieger, R. ; Sparmann, U.:
Structure Based Methods for Parallel Pattern Fault Simulation in Combinational Circuits

4/1990 Goldstine, J. ; Kintala, C.M.R. ; Wotschke D.:
On Measuring Nondeterminism in Regular Languages

5/1990 Goldstein, J. ; Leung, H. ; Wotschke, D.:
On the Relation between Ambiguity and Nondeterminism in Finite Automata

1/1991 Brause, Rüdiger:
Approximator Networks and the Principles of Optimal Information Distribution

2/1991 Brauer, Johannes ; Stuchly, Jürgen:
HyperEDIF: Ein Hypertext-System für VLSI Entwurfsdaten

3/1991 Brauer, Johannes:
Repräsentation von Entwurfsdaten als symbolische Ausdrücke

4/1991 Trier, Uwe:
Additive Weights of a Special Class of Nonuniformly Distributed Backtrack Trees

5/1991 Dömel, P. [u.a.]:
Concepts for the Reuse of Communication Software

6/1991 Heistermann, Jochen:
Zur Theorie genetischer Algorithmen

7/1991 Wang, Alexander [u.a.]:
Embedding complete binary trees in faulty hypercubes

1/1992 Brause, Rüdiger:
The Minimum Entropy Network

2/1992 Trier, Uwe:
Additive Weights Under the Balanced Probability Model

3/1992 Trier, Uwe:
(Un)expected path lengths of asymetric binary search trees

4/1992 Coen Alberto ; Lavazza, Luigi ; Zicari, Roberto:
Assuring type-safety of object oriented languages

5/1992 Coen, Alberto ; Lavazza, Luigi ; Zicari, Roberto:
Static type checking of an object-oriented database schema

6/1992 Coen, Alberto ; Lavazza, Luigi ; Zicari, Roberto:
Overview and progress report of the ESSE project : Supporting object-oriented database schema analysis and evolution

7/1992 Schmidt-Schauß, Manfred:
Some results for unification in distributive equational theories

8/1992 Mayr, Ernst W. ; Werchner, Ralph:
Divide-and-conquer algorithms on the hypercube

1/1993 Becker, Bernd ; Drechsler, Rolf ; Hengster, Harry:
Local circuit transformations preserving robust path-delay-fault testability

2/1993 Krieger, Rolf ; Becker, Bernd ; Sinković, Robert:
A BDD-based algorithmen for computation of exact fault detection probabilities

3/1993 Mayr, Ernst W. ; Werchner, Ralph:
Optimal routing of parentheses on the hypercube

4/1993 Drechsler, Rolf ; Becker, Bernd:
Rapid prototyping of fully testable multi-level AND/EXOR networks

5/1993 Becker, Bernd ; Drechsler, Rolf:
On the computational power of functional decision diagrams

6/1993 Berghoff, P. ; Dömel, P. ; Drobnik, O. [u.a.]:
Development and management of communication software systems

7/1993 Krieger, Rolf ; Hahn, Ralf ; Becker Bernd:
test_circ : Ein abstrakter Datentyp zur Repräsentation von hierarchischen Schaltkreisen (Benutzeranleitung)

8/1993 Krieger, Rolf ; Becker, Bernd ; Hengster, Harry:
lgc++ : Ein Werkzeug zur Implementierung von Logiken als abstrakte Datentypen in C++ (Benutzeranleitung)

9/1993 Becker, Bernd ; Drechsler, Rolf ; Meinel, Christoph:
On the testability of circuits derived from binary decision diagrams

10/1993 Liu, Ling ; Zicari, Roberto ; Liebherr, Karl ; Hürsch, Walter:
Polymorphic reuse mechanism for object-oriented database specifications

11/1993 Ferrandina, Fabrizio ; Zicari, Roberto:
Object-oriented database schema evolution: are lazy updates always equivalent to immediate updates ?

12/1993 Becker, Bernd ; Drechsler, Rolf ; Werchner, Ralph:
On the Relation Between BDDs and FDDs

13/1993 Becker, Bernd ; Drechsler, Rolf:
Testability of circuits derived from functional decision diagrams

14/1993 Drechsler, R. ; Sarabi, A. ; Theobald, M. ; Becker, B. ; Perkowski, M.A.:
Efficient repersentation and manipulation of switching functions based on ordered Kronecker functional decision diagrams

15/1993 Drechsler, Rolf ; Theobald, Michael ; Becker, Bernd:
Fast FDD based Minimization of Generalized Reed-Muller Forms

1/1994 Ferrandina, Fabrizio ; Meyer, Thorsten ; Zicari, Roberto:
Implementing lazy database updates for an object database system

2/1994 Liu, Ling ; Zicari, Roberto ; Hürsch, Walter ; Liebherr, Karl:
The Role of Polymorhic Reuse mechanism in Schema Evolution in an Object-oriented Database System

3/1994 Becker, Bernd ; Drechsler, Rolf ; Theobald, Michael:
Minimization of 2-level AND/XOR Expressions using Ordered Kronecker Functional Decision Diagrams

4/1994 Drechsler, R. ; Becker, B. ; Theobald, M. ; Sarabi, A. ; Perkowski, M.A.:
On the computational power of Ordered Kronecker Functional Decision Diagrams

5/1994 Even, Susan ; Sakkinen, Marku:
The safe use of polymorphism in the O2C database language

6/1994 GI/ITG-Workshop:
Anwendungen formaler Methoden im Systementwurf : 21. und 22. März 1994

7/1994 Zimmermann, M. ; Mönch, Ch. [u.a.]:
Die Telematik-Klassenbibliothek zur Programmierung verteilter Anwendungen in C++

8/1994 Zimmermann, M. ; Krause, G.:
Eine konstruktive Beschreibungsmethodik für verteilte Anwendungen

9/1994 Becker, Bernd ; Drechsler, Rolf:
How many Decomposition Types do we need ?

10/1994 Becker, Bernd ; Drechsler, Rolf:
Sympathy: Fast Exact Minimization of Fixed Polarity Reed-Muller Expression for Symmetric Functions

11/1994 Drechsler, Rolf ; Becker, Bernd ; Jahnke, Andrea:
On Variable Ordering and Decompostion Type Choice in OKFDDs

12/1994 Schmidt-Schauß:
Unification of Stratified Second-Order Terms

13/1994 Schmidt-Schauß:
An Algorithmen for Distributive Unification

14/1994 Becker, Bernd ; Drechsler, Rolf:
Synthesis for Testability: Circuit Derived from ordered Kronecker Functional Decision Diagrams

15/1994 Bär, Brigitte:
Konformität von Objekten in offenen verteilten Systemen

16/1994 Seidel, T. ; Puder, A. ; Geihs, K. ; Gründer, H.:
Global object space: Modell and Implementation

17/1994 Drechsler, Rolf ; Esbensen, Henrik ; Becker, Bernd:
Genetic algorithms in computer aided design of integrated circuits

1/1995 Schütz, Marko:
The $G^{\#}$-Machine: efficient strictness analysis in Haskell

2/1995 Henning, Susanne ; Becker, Bernd:
GAFAP: A Linear Time Scheduling Approach for High-Level-Synthesis

3/1995 Drechsler, Rolf ; Becker, Bernd ; Göckel, Nicole:
A Genetic Algorithm for variable Ordering of OBDDs

4/1995 Nebel, Markus E.:
Exchange Trees, eine Klasse Binärer Suchbäume mit Worst Case Höhe von $\log(n)$

5/1995 Drechsler, Rolf ; Becker, Bernd:
Dynamic Minimization of OKFDDs

6/1995 Breché, Philippe ; Ferrandina, Fabrizio ; Kuklok, Martin:
Simulation of Schema and Database Modification using Views

7/1995 Breché, Philippe ; Wörner, Martin:
Schema Update Primitives for ODB Design

8/1995 Schmidt-Schauß, Manfred:
On the Sematics and Interpretation of Rule Based Programs with Static Global Variables

9/1995 Rußmann, Arnd:
Adding Dynamic Actions to LL(k) Parsers

10/1995 Rußmann, Arnd:
Dynamic LL(k) Parsing

11/1995 Leyendecker, Thomas ; Oehler, Peter ; Waldschmidt, Klaus:
Spezifikation hybrider Systeme

12/1995 Cerone, Antonio ; Maggiolo-Schettini, Andrea:
Time-based Expressivity of Times Petri Nets

1/1996 Schütz, Marko ; Schmidt-Schauß, Manfred:
A Constructive Calculus Using Abstract Reduction for Context Analysis (nicht erschienen)

2/1996 Schmidt-Schauß, Manfred:
CPE: A Calculus for Proving Equivalence of Expressions in a Nonstrict Functional Language

1/1997 Kemp, Rainer:
On the Expected Number of Nodes at Level k in 0-balanced Trees

2/1997 Nebel, Markus:
New Results on the Stack Ramification of Binary Trees

3/1997 Nebel, Markus:
On the Average Complexity of the Membership Problem for a Generalized Dyck Language

4/1997 Liebehenschel, Jens:
Ranking and Unranking of Lexicographically Ordered Words: An Average-Case Analysis

5/1997 Kappes, Martin:
On the Generative Capacity of Bracketed Contextual Grammars

1/1998 Arlt, B. ; Brause, R.:
The Principal Independent Components of Images. *Elektronisch publiziert unter URL http://www.informatik.uni-frankfurt.de/fbreports/fbreport1-98.ps.gz*

2/1998 Miltrup, Matthias ; Schnitger, Georg:
Large Deviation Results for Quadratic Forms

3/1998 Miltrup, Matthias ; Schnitger, Georg:
Neural Networks and Efficient Associative Memory

4/1998 Kappes, Martin:
Multi-Bracketed Contextual Grammars

5/1998 Liebehenschel, Jens:
Lexicographical Generation of a Generalized Dyck Language

6/1998 Kemp, Rainer:
On the Joint Distribution of the Nodes in Uniform Multidimensional Binary Trees

7/1998 Liebehenschel, Jens:
Ranking and Unranking of a Generalized Dyck Language

8/1998 Grimm, Christoph ; Waldschmidt, Klaus:
Hybride Datenflußgraphen

9/1998 Kappes, Martin:
Multi-Bracketed Contextual Rewriting Grammars

1/1999 Kemp, Rainer:
On Leftist Simply Generated Trees

2/1999 Kemp, Rainer:
A One-to-one Correspondence Between a Class of Leftist Trees and Binary Trees

3/1999 Kappes, Martin:
Combining Contextual Grammars and Tree Adjoining Grammars

4/1999 Kappes, Martin:
Descriptional Complexity of Deterministic Finite Automata with Multiple Initial States

5/1999 Nebel, Markus E.:
New Knowledge on AVL-Trees

6/1999 Manfred Schmidt-Schauß, Marko Schütz (editors):
13th International Workshop on Unification

7/1999 Brause, R.; Langsdorf, T.; Hepp, M.:
Credit Card Fraud Detection by Adaptive Neural Data Mining. *Elektronisch publiziert unter URL http://www.informatik.uni-frankfurt.de/fbreports/fbreport7-99.ps.gz*

8/1999 Kappes, Martin:
External Multi-Bracketed Contextual Grammars

9/1999 Priese, Claus P.:
A Flexible Type-Extensible Object-Relational DataBase Wrapper-Architecture

10/1999 Liebehenschel, Jens:
The Connection between Lexicographical Generation and Ranking

11/1999 Brause, R.; Arlt, B.; Tratar, E.:
A Scale-Invariant Object Recognition System for Content-based Queries in Image Databases. *Elektronisch publiziert unter URL http://www.informatik.uni-frankfurt.de/fbreports/fbreport11-99.ps.gz*

12/1999 Kappes, M.; Klemm, R. P.; Kintala, C. M. R.:
Determining Component-based Software System Reliability is Inherently Impossible

13/1999 Kappes, Martin:
Multi-Bracketed Contextual Rewriting Grammars With Obligatory Rewriting

14/1999 Kemp, Rainer:
On the Expected Number of Leftist Nodes in Simply Generated Trees

1/2000 Kemp, Rainer:
On the Average Shape of Dynamically Growing Trees

2/2000 Arlt, B.; Brause, R.; Tratar, E.:
MASCOT: A Mechanism for Attention-based Scale-invariant Object Recognition in Images. *Elektronisch publiziert unter URL http://www.cs.uni-frankfurt.de/fbreports/fbreport2-00.pdf*

3/2000 Heuschen, Frank; Waldschmidt, Klaus:
Bewertung analoger und digitaler Schaltungen der Signalverarbeitung

4/2000 Hamker, Fred H.; Paetz, Jürgen; Thöne, Sven; Brause, Rüdiger; Hanisch, Ernst:
Erkennung kritischer Zustände von Patienten mit der Diagnose „Septischer Schock" mit einem RBF-Netz. *Elektronisch publiziert unter URL http://www.cs.uni-frankfurt.de/fbreports/fbreport04-00.pdf*

1/2001 Nebel, Markus E.:
A Unified Approach to the Analysis of Horton-Strahler Parameters of Binary Tree Structures

2/2001 Nebel, Markus E.:
Combinatorial Properties of RNA Secondary Structures

3/2001 Nebel, Markus E.:
Investigation of the Bernoulli-Model for RNA Secondary Structures

4/2001 Malcher, Andreas:
Descriptional Complexity of Cellular Automata and Decidability Questions