Arbeitsgruppe Informatik
Universität Giessen
Arndtstr. 2, D-35392 Giessen, Germany

On the power of one-way bounded cellular time computers

Thomas Buchholz Martin Kutrib

Report 9604 December 1996

JUSTUS-LIEBIG-
UNIVERSITÄT
GIESSEN

On the power of one-way bounded cellular time computers

Thomas Buchholz and Martin Kutrib

AG Informatik, University of Giessen
Arndtstr. 2, D-35392 Giessen, Germany
{buchholz,kutrib}@informatik.uni-giessen.de

December 1996

Abstract

Comparisons of different cellular devices and the investigation of their computing power can be made in terms of their capabilities to time-construct and time-compute functions. Time-construction means that a distinguished cell has to enter distinguished states exactly at the time steps $f(1), f(2), \ldots$, whereas time-computation requires the distinguished cell to enter a distinguished state firstly at time step $f(n)$, where n is the length of the input. Here the family of functions which are time-constructible by a two-way unbounded cellular space ($\mathscr{F}(\mathrm{CS})$) is characterized in terms of functions which are time-computable by one of the simplest cellular devices, a one-way bounded cellular automaton ($\mathscr{C}(\mathrm{OCA})$). Conceptually, time-constructible functions have to be strictly increasing. Regarding that restriction the reverse characterization is shown, too.

Some results concerning the structure of $\mathscr{F}(\mathrm{CS})$ and $\mathscr{C}(\mathrm{OCA})$ and their relation to formal language recognition are established.

1 Introduction

In the field of computational complexity theory there is a particular interest in infinite dense hierarchies of complexity classes defined by bounding some resources by a function f. Most of the corresponding proofs require "well-behaved" complexity functions. The notion "well-behaved" is usually concretized in terms of the constructibility of functions with respect to the device investigated.

If we investigate the constructibility of functions in parallel polyautomata we are strongly concerned with the concept of signals. In a lot of works specific pulses (or signals) are an useful tool to construct algorithms, for example prime number generators [5] or a solution to the famous firing squad synchronization problem [1, 11]. A general investigation of signals of its own in polyautomata was started by Mazoyer and Terrier [8, 9]. They considered signals and the constructibility of functions in two-way unbounded cellular automata (cellular spaces).

Since signals can encode and propagate information through the automaton their realizability can show the computation power and the limitations of the model. Moreover, we can regard signals as a higher programming concept which allows modularization techniques at algorithm design.

In the field of polyautomata theory a problem arises with the end of a computation. From the usual definition it follows that the machines will never halt. A common way of defining final configurations is to define a predicate these configurations have to fulfill. In case of language recognition this predicate mostly requires a border cell to be in a designated final state. But what afterwards a final configuration is reached? One can additionally require that final configurations are stable in some sense, i.e. a final configuration leads always to a final configuration. If this property is not required and a certain complexity class is under consideration, i.e. the resulting configuration is taken by the outside world at a time step $f(n)$, then the ability of recognizing time step $f(n)$ is an ability of the outside world and not necessarily of the model. But both cases coincide if the time step $f(n)$ is recognizable by the automaton itself, i.e. if the automaton can time-compute f.

The object of the present paper is to establish some results concerning the time-constructibility in two-way bounded and unbounded cellular arrays and the time-computability in one-way and two-way bounded cellular arrays. Some of the proofs are done in the field of formal language recognition. As far as not available the corresponding bridges are built.

We focus our interest on relations between the functions which are time-constructible in cellular spaces and functions which are time-computable in one-way cellular automata. On one side there are the computational universal cellular spaces whereas on the other side we have quite simple devices, the one-way bounded cellular automata. But nevertheless we can fully characterize one family by the other and vice versa.

2 Definitions

We denote the integers by $\mathbb{Z}$, the positive natural numbers $\{1, 2, \ldots\}$ by $\mathbb{N}$, the set $\mathbb{N} \cup \{0\}$ by $\mathbb{N}_0$.

2.1 Cellular devices

A *cellular space* is a linear array of deterministic finite automata which is infinite to one end. For convenience we identify the single nodes, sometimes called cells, by natural numbers.

A *cellular automaton* is a (space) bounded cellular space where the number of cells depends on the size of the input.

The state transition function is applied to all cells synchronously at discrete time steps. It depends on the state of the cell itself and on the states of the nodes the cell is connected to, sometimes called its neighbors.

We distinguish two different interconnection patterns that are related to one-way and two-way information flow through the network. In the first case each cell is connected to its right nearest neighbor only, thus transmission is from right to left. In the second case each cell is connected to its both nearest neighbors.

For simplicity we assume that the border node(s) are initialized especially such that their states indicate their distinguished position.

Definition 1 A *cellular space* (CS) is a system $(S, \sigma, \#, q)$, where

a) S is the finite, nonempty set of *states*,
b) $\# \in S$ is the *boundary state*,
c) $q \in S$ is the *quiescent state*,
d) $\sigma : S^3 \to S$ is the *local transition function* satisfying
$\forall\, s_1, s_2, s_3 \in S : \sigma(s_1, s_2, s_3) = \# \iff s_2 = \#$ and $\sigma(q, q, q) = q$.

The local transition function induces a global transition $\mathcal{T} : S^+ \longrightarrow S^+$ according to the following:

Let $n \in \mathbb{N}$ be an arbitrary natural number and $s_1, \ldots, s_n \in S$

$$\begin{aligned} \mathcal{T}(s_1) &:= \sigma(\#, s_1, q)\sigma(s_1, q, q) \\ \mathcal{T}(s_1 \cdots s_n) &:= \sigma(\#, s_1, s_2)\sigma(s_1, s_2, s_3) \cdots \sigma(s_{n-1}, s_n, q)\sigma(s_n, q, q) \end{aligned}$$

In cellular spaces we are always considering two-way information flow.

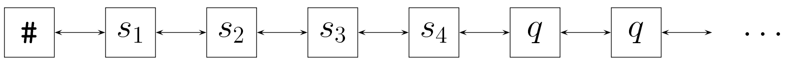

Figure 1: A cellular space.

If the number of cells is bounded, the resulting device is a two-way *cellular automaton* (CA for short). Now we have the local transition function length preserving:

$$\begin{aligned} \mathcal{T}(s_1) &:= \sigma(\#, s_1, \#) \\ \mathcal{T}(s_1 \cdots s_n) &:= \sigma(\#, s_1, s_2)\sigma(s_1, s_2, s_3) \cdots \sigma(s_{n-1}, s_n, \#) \end{aligned}$$

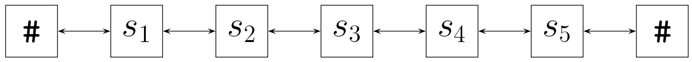

Figure 2: A two-way cellular automaton.

If we restrict the flow of information to one-way, we obtain a quite simple device, a *one-way cellular automaton* (OCA for short).

The local transition function now maps from S^2 to S and satisfies $\sigma(s_1, s_2) = \# \iff s_2 = \#$ and $\sigma(q,q) = q$. It induces the following global transition:

$$\begin{array}{rcl} \mathcal{T}(s_1) & := & \sigma(s_1, \#) \\ \mathcal{T}(s_1 \cdots s_n) & := & \sigma(s_1, s_2) \cdots \sigma(s_n, \#) \end{array}$$

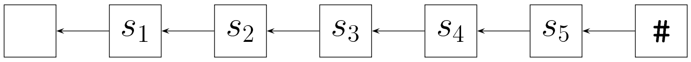

Figure 3: A one-way cellular automaton.

2.2 Computations

The following sections are devoted to the investigation and comparison of three types of computations in the previously described models.

A function f which maps the natural numbers to the natural numbers is said to be time-computed by cellular automata if the input of n identical items leads to a computation such that the configuration at time step $f(n)$ is distinguished. The distinction is done by means of a set of final states into which the leftmost cell of the network has to switch. The input item is the quiescent state q. $\pi_i(s_1 \cdots s_n) := s_i$ selects the ith component of $s_1 \cdots s_n$ and $\pi_{i,j}$ is an abbreviation for $\pi_i(\pi_j)$.

Definition 2 A function $f : \mathbb{N} \longrightarrow \mathbb{N}$ is *(O)CA-time-computable* iff there exist an (O)CA $M = (S, \sigma, \#, q)$ and a set of final states $F \subseteq S \setminus \{q\}$ such that $f(n)$ is the smallest natural number t for which $\pi_1(\mathcal{T}^t(q^n)) \in F$.

We denote the family of OCA-time-computable resp. CA-time-computable functions by $\mathscr{C}$(OCA) resp. $\mathscr{C}$(CA).

For time-computation an array has to compute the time step which corresponds to the value of the function on the length of input. Another computation, the time-constructibility of functions, requires the array to compute all values of the function up to the input length. For that we need to restrict to strictly increasing functions.

Definition 3 A strictly increasing function $f : \mathbb{N} \longrightarrow \mathbb{N}$ is *time-constructible* by a cellular space (CS-time-constructible) resp. by a cellular automaton (CA-time-constructible) iff there exist a CS resp. CA $M = (S, \sigma, \#, q)$ and a set of final states $F \subseteq S$ such that for all $t \in \mathbb{N}$ resp. $t \in \{1, \dots, f(n)\}$:

$$\pi_1(\mathcal{T}^t(q)) \in F \iff \exists\, i \in \mathbb{N} : t = f(i) \text{ resp.}$$
$$\pi_1(\mathcal{T}^t(q^n)) \in F \iff t \in \bigcup_{i=1}^{n} \{f(i)\}.$$

The family of CS-time-constructible resp. CA-time-constructible functions is denoted by $\mathscr{F}$(CS) resp. $\mathscr{F}$(CA).

Our third type of computation is the most classical one when comparisons between different models are made: the formal language processing. Although in principle the family of languages acceptable by a specific array may be considered of its own it is usual to bound some resource e.g. the time.

Definition 4

a) A word $w = w_1 \cdots w_n \in A^+$ is accepted by a(n) (O)CA with state set S iff $A \subseteq S$ and there exists a set of final states $F \subseteq S$ with the property that a cell once entered a final state remains in a final state and $n_0 \in \mathbb{N}$ such that n_0 is the smallest integer for which $\pi_1(\mathcal{T}^{n_0}(w_1 \cdots w_n)) \in F$ holds. The word is said to be accepted in n_0 time steps.

b) A formal language L is accepted by a(n) (O)CA M with time complexity $t : \mathbb{N} \longrightarrow \mathbb{N} \iff L = \{w \mid w \text{ is accepted by } M \text{ in at most } t(|w|) \text{ time steps}\}$.

The family of all languages which can be accepted by a CA resp. OCA with time complexity $t(n)$ is denoted by $\mathscr{L}_{t(n)}(\mathrm{CA})$ resp. $\mathscr{L}_{t(n)}(\mathrm{OCA})$. In this connection the identity id is denoted real-time and aliased to rt.

3 One-way bounded time-computability versus two-way unbounded time-constructibility

In the sequel f always denotes a strictly increasing mapping from $\mathbb{N}$ to $\mathbb{N}$.

Our first result in this section deals with the direct comparison of $\mathscr{C}(\mathrm{OCA})$ and $\mathscr{F}(\mathrm{CS})$.

On one hand a function seems to be easier time-computed than time-constructed, but on the other hand the construction is to be done in a two-way unbounded array against the computation in a one-way bounded array.

In the following we will incorporate some results from the fundamental work of Mazoyer and Terrier [8]. They have shown that the family $\mathscr{F}(\mathrm{CS})$ is closed under addition and in some sense under multiplication with positive rational coefficients ($f \in \mathscr{F}(\mathrm{CS}) \iff c \cdot f \in \mathscr{F}(\mathrm{CS})$, $c \in \mathbb{N}$). At first a useful lemma concerning the substraction of id from functions belonging to $\mathscr{F}(\mathrm{CS})$ is established.

Lemma 5 $f + id \in \mathscr{F}(\mathrm{CS}) \Longrightarrow f \in \mathscr{F}(\mathrm{CS})$

Proof. In [8] it has been shown that if g and h are belonging to $\mathscr{F}(\mathrm{CS})$ and $g \geq h$ and for a $b \in \mathbb{N}_0$ the function $(b+1)g - bh$ is strictly increasing, then it belongs to $\mathscr{F}(\mathrm{CS})$, too. Obviously, the identity is CS-time-constructible.

Let $g := f + id$ and $h := 2id$. From the closure under multiplication with rational coefficients it follows $h \in \mathscr{F}(\mathrm{CS})$. Since f is strictly increasing it holds $f \geq id$ and, hence, $g = f + id \geq id + id \geq 2id = h$.

Choose $b = 1$, then $(b+1)g - bh = 2g - h = 2f + 2id - 2id = 2f$ is strictly increasing and thus belongs to $\mathscr{F}(\mathrm{CS})$. Again from the closure under multiplication with rational coefficients we obtain $f \in \mathscr{F}(\mathrm{CS})$.

□

Now we are prepared to prove that every strictly increasing OCA-time-computable function is also CS-time-constructible.

Theorem 6 $f \in \mathscr{C}(\mathrm{OCA}) \Longrightarrow f \in \mathscr{F}(\mathrm{CS})$

Proof. Due to lemma 5 it suffices to show $f \in \mathscr{C}(\mathrm{OCA}) \Longrightarrow f + id \in \mathscr{F}(\mathrm{CS})$.

Let M be an OCA that time-computes f. By definition the flow of information in OCAs is from right to left. By reversing the arguments of the local transition we obtain an OCA M' that has an information flow from left to right. Obviously, in M' cell i becomes final at time step $f(i)$ at the first.

A cellular space which time-constructs f has to perform two tasks in parallel: One is to simulate M'. Additionally, every cell i sends a signal to the left at exactly that time step it becomes final for the first time, i.e. $f(i)$. This signal will arrive at the leftmost cell at time step $f(i) + i$ which is the ith value of $f + id$. Since f is strictly increasing the signals will not interfere. □

The converse of the theorem is not true: There are superexponential functions in $\mathscr{F}(\mathrm{CS})$ [8] which cannot belong to $\mathscr{C}(\mathrm{OCA})$ [2]. On the other hand it should be mentioned that there are functions in $\mathscr{C}(\mathrm{OCA})$ that are not strictly increasing [2] and therefore are not CS-time-constructible.

The aim of the present section is to characterize the family $\mathscr{F}(\mathrm{CS})$ in terms of OCA-time-computability. To step around the problem concerning the superexponential functions we walk on another string. The idea is to use the corresponding complement functions.

Definition 7 Let $f : \mathbb{N} \to \mathbb{N}$ be a strictly increasing function. The complement function of f is defined by

$$f^{-1} : \mathbb{N} \to \mathbb{N}_0,\ f^{-1}(m) = \max(\{n \in \mathbb{N} \mid f(n) \leq m\} \cup \{0\}).$$

The following lemma gives one direction of the characterization.

Lemma 8 $f \in \mathscr{F}(\mathrm{CS}) \Longrightarrow f^{-1} + 2id \in \mathscr{C}(\mathrm{OCA})$

Proof. In [8] it has been shown that the unary language $L_f := \{\mathtt{a}^{f(n)} \mid n \in \mathbb{N}\}$ belongs to $\mathscr{L}_{rt}(\mathrm{CA})$ iff f belongs to $\mathscr{F}(\mathrm{CS})$. In [4] it has been shown that L belongs to $\mathscr{L}_{rt}(\mathrm{CA})$ iff L^R belongs to $\mathscr{L}_{2id}(\mathrm{OCA})$. ($L^R$ denotes the mirror image of L.)

Since for unary languages we have $L = L^R$ and by the assertion $f \in \mathscr{F}(\mathrm{CS})$ the language L_f belongs to $\mathscr{L}_{2id}(\mathrm{OCA})$. We are going to construct an OCA M'

that time-computes $f^{-1} + 2id$ from a given OCA M that accepts L with time complexity $2id$.

Due to one-way information flow a cell is not influenced by cells located to its left. Therefore and since $2id \in \mathscr{C}(\mathrm{OCA})$ we may assume that each cell i of M becomes final at time $2(n-i+1)$ at the earliest.

M' has to perform two tasks in parallel. The first one is to simulate M. The second one is to generate and delay a signal s as follows. s is generated at the right border at the beginning of the computation. It moves with speed $1/2$ to the left.

When it enters a new cell i (i.e. at time $2(n-i+1)$ or later) this one has finished its decision whether the word $\mathtt{a}^{n-i+1}$ to its right belongs to L_f (i.e. whether there exists a m such that $f(m) = n-i+1$). If it belongs to, then the signal is delayed for one time step.

Altogether the signal is delayed for $|\{m \in \mathbb{N} \mid f(m) \leq n\}|$ time steps and arrives at time $|\{m \in \mathbb{N} \mid f(m) \leq n\}| + 2n$ at the left border. Since f is strictly increasing $|\{m \in \mathbb{N} \mid f(m) \leq n\}| = f^{-1}(n)$ holds. Thus, s may cause the left border to enter a final state and M' time-computes $f^{-1} + 2id$. □

In order to proof the converse of the lemma, which would complete our characterization, we will show a relationship to formal language recognition.

Lemma 9 Let $c \in \mathbb{N}$ be a natural number.

$$f^{-1} + c \cdot id \in \mathscr{C}(\mathrm{OCA}) \Longrightarrow L_f = \{\mathtt{a}^{f(n)} \mid n \in \mathbb{N}\} \in \mathscr{L}_{(c+1)id}(\mathrm{OCA})$$

Proof. Let M be an OCA that time-computes $f^{-1} + c \cdot id$. We are going to construct a language acceptor M' for L_f. M' performs two tasks in parallel.

The first one is to simulate M on its whole input of length n.

The second one is to simulate M on an input of length $n-1$. The second simulation has to start c time steps delayed. This can easily be achieved by initializing cell n as pseudo border cell during the first time step. Delaying the start by c time steps can be done by each cell of its own.

The tasks would cause the leftmost cell to become final at time step $f^{-1}(n)+cn$ resp. $f^{-1}(n-1)+c(n-1)+c = f^{-1}(n-1)+cn$. Dependend on whether there exists a $m \in \mathbb{N}$ such that $f(m) = n$ or not, $f^{-1}(n) = f^{-1}(n-1) + 1$ or $f^{-1}(n) = f^{-1}(n-1)$.

Thus, if both tasks would cause the leftmost cell to become final exactly at the same time step, then $f^{-1}(n) = f^{-1}(n-1)$ and $\mathtt{a}^n$ does not belong to L. Otherwise it does.

Since $f^{-1} \leq id$ the time complexity of the acceptor is at most $(c+1)id$. □

Now we are prepared to prove our main result.

Theorem 10 $f^{-1} + 2id \in \mathscr{C}(\text{OCA}) \iff f \in \mathscr{F}(\text{CS})$

Proof. By lemma 8 it suffices to show $f^{-1} + 2id \in \mathscr{C}(\text{OCA}) \Longrightarrow f \in \mathscr{F}(\text{CS})$. If we choose $c = 2$ then from lemma 9 it follows that $L_f := \{\mathtt{a}^{f(n)} \mid n \in \mathbb{N}\}$ belongs to $\mathscr{L}_{3id}(\text{OCA})$. In [6] it has been shown that it then belongs to $\mathscr{L}_{2id}(\text{OCA})$ and by a result in [4] L_f^R belongs to $\mathscr{L}_{rt}(\text{CA})$. Since $L_f = L_f^R$ we obtain $L_f \in \mathscr{L}_{rt}(\text{CA})$ and by the result in [8] cited above $f \in \mathscr{F}(\text{CS})$. $\square$

Example 11 The function $\exp^* : \mathbb{N} \to \mathbb{N}$ is defined according to $\exp^*(1) = 2^1$, $\exp^*(n+1) = 2^{\exp^*(n)}$. Its complement function is denoted by $\log^*$. It holds for $\log(0) = 0$: $\log^*(m) = \min\{n \in \mathbb{N}_0 \mid \log^n(m) = 1\}$.

It is known that $L = \{\mathtt{a}^{\exp^*(n)} \mid n \in \mathbb{N}\}$ belongs to $\mathscr{L}_{rt}(\text{CA})$ [7]. Thus $\exp^*$ is CS-time-constructible and by theorem 10 the function ${\exp^*}^{-1} + 2id = \log^* + 2id$ is OCA-time-computable.

Since in $\mathscr{C}(\text{OCA})$ there is a gap between id and $id+\log$ [2] that example becomes important to the search for other gaps in $\mathscr{C}(\text{OCA})$.

4 Two-way bounded time-computability versus two-way unbounded time-constructibility

In the present section we are interested in obtaining similar results for two-way bounded time computers as for OCAs in the previous section.

Unfortunately, we can only prove one direction. The other one is directly related to the longstanding open problem whether $\mathscr{L}_{rt}(\text{CA})$ is identical to $\mathscr{L}_{lt}(\text{CA})$ or not.

Fortunately, we can prove a stronger version of that one direction.

Once the following theorem is proved we can derive another one with $\mathscr{C}(\text{CA})$ at the right side since in [3] has been shown that the family $\mathscr{F}(\text{CA})$ is properly contained in $\mathscr{C}(\text{CA})$.

Theorem 12 $f \in \mathscr{F}(\text{CS}) \Longrightarrow f^{-1} + id \in \mathscr{F}(\text{CA})$

Proof. Since f belongs to $\mathscr{F}(\text{CS})$ it does $2f$ [8]. We are going to prove that $2(f^{-1} + id)$ is CS-time-constructible. Applying the result in [8] again shows the CS-time-constructibility of $f^{-1} + id$. Since $f^{-1} + id \leq 2id$ the function $f^{-1} + id$ is linearly bounded and, thus, can be constructed by a (space) bounded CA.

It remains to show $2f \in \mathscr{F}(\text{CS}) \Longrightarrow 2(f^{-1} + id) \in \mathscr{F}(\text{CS})$.

The rest of the proof is divided in two sections. At first a bundle of signals is introduced which set up a CS time constructor M' for $2(f^{-1}+id)$. Subsequently we will verify that M' does what it should do. For the following cf. figure 4.

Let M be a CS time constructor for $2f$.

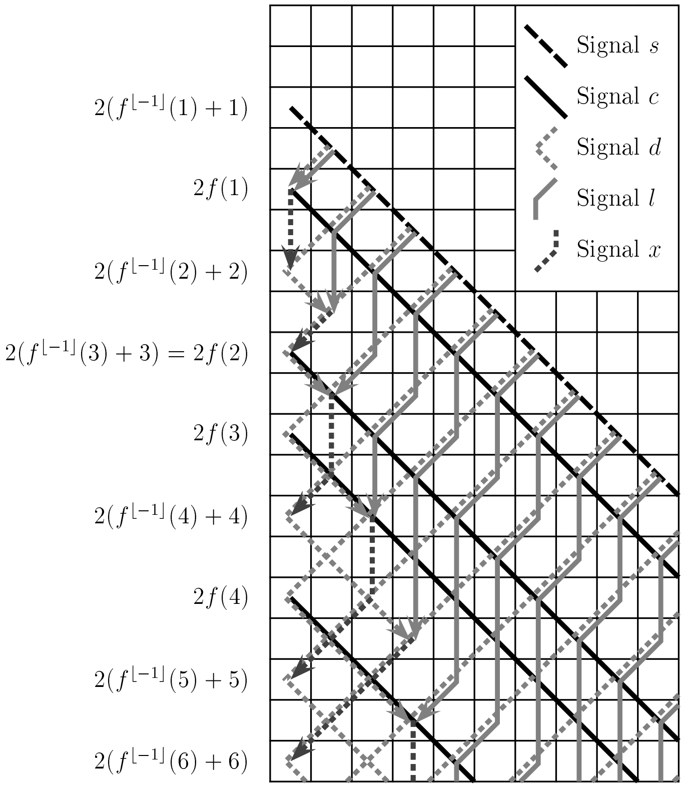

Figure 4: Example to theorem 12.

Basically, M' simulates M. At every time step $2f(n)$, $n \in \mathbb{N}$, the leftmost cell gets marked by the simulated constructor M. At these time steps right moving signals c with speed 1 are generated by the leftmost cell.

Additionally, at time step 2 a right moving signal s with speed 1 is generated. At each cell passed through by s two left moving signals l and d with speed 1 are transmitted. A signal l is delayed for two time steps when it collides with a signal c.

The signals d are moving without any delay to the leftmost cell and are bounced there.

If two signals d and l meet again, they are dropped and a left moving signal x with speed 1 is transmitted instead. If the meeting takes place together with a meeting of a signal c then the signal x is delayed for two time steps before it is transmitted to the left.

Each arrival of a signal x in the leftmost cell happens at a time step $2(f^{-1}(m)+m)$, $m \in \mathbb{N}$, and vice versa.

Now we turn to the question why the signals x do their job well.

Let us consider the signals d and l which are transmitted at time step $m+1$, $m \geq 1$, by the signal s. d arrives at cell 1 at time $m+1+m-1 = 2m$. Subsequently, it is bounced and determines the area in space-time in which l

may be delayed. But l is delayed only if there exist some signals c that have to be transmitted at time steps $2f(n)$, where $2f(n) \leq 2m$, thus $f(n) \leq m$.

On the other hand we have $\{n \in \mathbb{N} \mid f(n) \leq m\} = \{1, 2, \ldots, f^{-1}(n)\}$ and therefore $f^{-1}(m) = |\{n \in \mathbb{N} \mid f(n) \leq m\}|$. It follows that the signal l is delayed for $2f^{-1}(m)$ time steps. Since the signal x transmitted at the meeting of d and l moves leftward with speed 1 this delay corresponds exactly to the time between the arrival of the signals d and x at the leftmost cell. Hence, x arrives in cell 1 exactly at time $2f^{-1}(m) + 2m = 2(f^{-1}(m) + m)$.

It remains to clarify that the cells can check whether meeting d and l signals are partners, i.e. they are generated at the same time. But it can easily be verified that the first (bounced) signal d that meets l is its partner, which proves the theorem. □

Corollary 13 $f \in \mathscr{F}(\mathrm{CS}) \Longrightarrow f^{-1} + id \in \mathscr{C}(\mathrm{CA})$

The proof of the following lemma is analogous to the proof of lemma 9. If at least for unary languages it would hold $\mathscr{L}_{lt}(\mathrm{CA}) = \mathscr{L}_{rt}(\mathrm{CA})$ then it would prove the converse of theorem 12.

Lemma 14 Let $c \in \mathbb{N}$ be a natural number.

$$f^{-1} + c \cdot id \in \mathscr{C}(\mathrm{CA}) \Longrightarrow \{\mathsf{a}^{f(n)} \mid n \in \mathbb{N}\} \in \mathscr{L}_{(c+1)id}(\mathrm{CA})$$

References

[1] Balzer, R. M. *An 8-state minimal time solution to the firing squad synchronization problem*. Information and Control 10 (1967), 22–42.

[2] Buchholz, Th. and Kutrib, M. *On time computability of functions in one-way cellular automata*. Report 9502, Arbeitsgruppe Informatik, Universität Gießen, Gießen, 1995.

[3] Buchholz, Th. and Kutrib, M. *Some relations between massively parallel arrays*. Parallel Computing (1997), to appear.

[4] Choffrut, C. and Čulik II, K. *On real-time cellular automata and trellis automata*. Acta Informatica 21 (1984), 393–407.

[5] Fischer, P. C. *Generation of primes by a one-dimensional real-time iterative array*. Journal of the ACM 12 (1965), 388–394.

[6] Ibarra, O. H., Kim, S. M., and Moran, S. *Sequential machine characterizations of trellis and cellular automata and applications*. SIAM Journal on Computing 14 (1985), 426–447.

[7] Kutrib, M. *A note on cellular real-time recognizability of languages definable by a family of functions*. Unpublished manuscript, 1995.

[8] Mazoyer, J. and Terrier, V. *Signals in one dimensional cellular automata*. Research Report RR 94-50, Ecole Normale Supérieure de Lyon, Lyon, 1994.

[9] Terrier, V. *Signals in linear cellular automata*. Proceedings of a Workshop on Cellular Automata, 1991.

[10] Vollmar, R. *Algorithmen in Zellularautomaten*. Teubner, Stuttgart, 1979.

[11] Waksman, A. *An optimum solution to the firing squad synchronization problem*. Information and Control 9 (1966), 66–78.