Johann Wolfgang Goethe-Universität
Frankfurt am Main

Institut für Informatik
Fachbereich Biologie und Informatik

On Two-Way Communication in Cellular Automata with a Fixed Number of Cells

Andreas Malcher

Nr. 2/03

Frankfurter Informatik-Berichte

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

16-9107

On Two-Way Communication in Cellular Automata with a Fixed Number of Cells

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

Abstract

The effect of adding two-way communication to k cells one-way cellular automata (kC-OCAs) on their size of description is studied. kC-OCAs are a parallel model for the regular languages that consists of an array of k identical deterministic finite automata (DFAs), called cells, operating in parallel. Each cell gets information from its right neighbor only. In this paper, two models with different amounts of two-way communication are investigated. Both models always achieve quadratic savings when compared to DFAs. When compared to a one-way cellular model, the result is that minimum two-way communication can achieve at most quadratic savings whereas maximum two-way communication may provide savings bounded by a polynomial of degree k.

1 Introduction

The descriptional complexity of abstract machines is a field of theoretical computer science in which the size of description of certain objects is studied. One main question is how the size of description varies when an object is described by several descriptional systems. One early and basic result is from Meyer and Fischer in [9] who proved that there exists an infinite sequence of regular languages $(L_n)_{n\geq 1}$ such that each L_n is recognized by an n-state nondeterministic finite automaton (NFA) and each equivalent deterministic finite automaton (DFA) needs at least 2^n states. Since an NFA can be converted to a DFA with at most 2^n states by the subset construction, their result shows that there is a tight exponential trade-off between NFAs and DFAs. In [9] it is additionally proven that the trade-off between two descriptional systems may not be bounded by any recursive function. They showed such a non-recursive trade-off between context-free grammars and DFAs.

In preceding papers some research on the descriptional complexity of cellular automata was started. In [5] it is proven that there are non-recursive trade-offs between different models of unrestricted cellular automata. A cellular automaton can be described as a set of many identical DFAs, called cells, which are arranged in a line. The next state of each 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. It is known [5] that for unrestricted cellular automata almost all decidability questions are undecidable and not even semidecidable and that there exist neither pumping lemmas nor minimization algorithms for these automata. Thus, it is obvious to restrict the general model. This is done in [6] where a one-way cellular automaton with a fixed number k of cells (kC-OCA) is studied whereas an unrestricted OCA is provided with as many cells as the input is long. The generative capacity of the restricted model is then reduced to the set of regular languages. The trade-off between kC-OCAs and DFAs is bounded by a polynomial of degree k. The upper bound when converting an n-state DFA to a kC-OCA is $n+1$ and this upper bound is known to be tight [7]. That is, there are regular languages which are "inherently sequential," since both a sequential and a parallel model need almost the same number of states.

In this paper, we want to continue the study of kC-OCAs and we look at the effect of adding two-way communication on the size of description. We investigate two generalized models. In the one model only the rightmost cell is allowed to communicate with its left neighbor (kC-OCA$_t$) and in the other model all cells are allowed to communicate with their left neighbors (kC-CA). The main results are as follows. Even one two-way communication cell is sufficient so that every n-state DFA can be equivalently converted to a kC-OCA$_t$ with $O(\sqrt{n})$ states. Hence, "two-way is always better than one-way". The "inherently sequential" languages mentioned above can be accepted by a kC-CA with n states whereas any DFA or kC-OCA needs at least $O(n^k)$ states. Thus, two-way communication always provides quadratical savings when compared with DFAs and may provide polynomial savings of degree k when compared with kC-OCAs.

2 Preliminaries and Definitions

Let Σ^* denote the set of all strings over the finite alphabet Σ, ϵ the empty string, and $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$. By $|w|$ we denote the length of a string w and by $|M|$ the number of states of a DFA M. Let REG denote the family of regular languages. In this paper we do not distinguish whether a language L contains the empty string ϵ or not. In other words, 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 [3]. We say that two automata are equivalent if both accept the same language. Concerning the notations and definitions for kC-OCAs, kC-OCA$_t$s, and kC-CAs we adapt the notations of the unrestricted model as introduced in [4] to our needs. More detailed information on unrestricted cellular automata can be found in [4], more detailed information on kC-OCAs may be found in [6].

Definition: A k cells one-way cellular automaton (kC-OCA) A is a tuple $A = (Q, \Sigma, \sqcup, \triangledown, k, \delta_r, \delta, F)$, where

1. $Q \neq \emptyset$ is the finite set of cell states,
2. Σ is the input alphabet,
3. $\sqcup \notin Q \cup \Sigma$ is the quiescent state,

4. $\triangledown \notin Q \cup \Sigma$ is the end-of-input symbol,
5. k is the number of cells,
6. $F \subseteq Q$ the set of accepting cell states and
7. $\delta_r : (Q \cup \{\sqcup\}) \times (\Sigma \cup \{\triangledown\}) \to Q \cup \{\sqcup\}$ is the local transition function for the rightmost cell. We require that only the pair $(\sqcup, \triangledown)$ is mapped to $\sqcup$.
8. $\delta : (Q \cup \{\sqcup\}) \times (Q \cup \{\sqcup\}) \to Q \cup \{\sqcup\}$ is the local transition function for the other cells. We require that only the pair $(\sqcup, \sqcup)$ is mapped to $\sqcup$.

A k cells one-way cellular automaton with two-way communication cell (kC-OCA$_t$) A is identical to a kC-OCA except that 7. is redefined as follows.

7'. $\delta_r : (Q \cup \{\sqcup\}) \times (Q \cup \{\sqcup\}) \times (\Sigma \cup \{\triangledown\}) \to Q \cup \{\sqcup\}$ is the local transition function for the rightmost cell. We require that only the tuple $(\sqcup, \sqcup, \triangledown)$ is mapped to $\sqcup$.

A k cells two-way cellular automaton (kC-CA) A is identical to a kC-OCA except that 7. and 8. are redefined. Since the leftmost cell has no left neighbor, an additional boundary state $\# \notin Q \cup \Sigma$ is needed.

7". $\delta_r : (Q \cup \{\sqcup, \#\}) \times (Q \cup \{\sqcup\}) \times (\Sigma \cup \{\triangledown\}) \to Q \cup \{\sqcup\}$ is the local transition function for the rightmost cell. We require that only the tuples $(\sqcup, \sqcup, \triangledown)$ and $(\#, \sqcup, \triangledown)$ are mapped to $\sqcup$.

8". $\delta : (Q \cup \{\sqcup, \#\}) \times (Q \cup \{\sqcup\}) \times (Q \cup \{\sqcup\}) \to Q \cup \{\sqcup\}$ is the local transition function for the other cells. We require that only the tuples $(\sqcup, \sqcup, \sqcup)$ and $(\#, \sqcup, \sqcup)$ are mapped to $\sqcup$.

The restricted models work similar to the unrestricted model. The next state of each cell depends on the current state of the cell itself and its right neighbor. The next state of the rightmost cell in kC-OCA$_t$s and all cells in kC-CAs additionally depend on the state of the left neighboring cell. The transition rule is applied synchronously to each cell at the same time. In contrast to unrestricted cellular automata the input is processed as follows. In the beginning all cells are in the quiescent state. The rightmost cell is the communicating cell to the input. At every time step one input symbol is processed by the rightmost cell. All other cells behave as described. The input is accepted, if the leftmost cell enters an accepting state. Since the minimal time to read the input and to send all information from the rightmost cell to the leftmost cell is the length of the input plus k, we input a special end-of-input symbol $\triangledown$ to the rightmost cell after reading the input. The size of an automaton $A = (Q, \Sigma, \sqcup, \triangledown, k, \delta_r, \delta, F)$ is defined as the number of states in Q, i.e. $|A| = |Q|$. To simplify matters we identify the cells by positive integers.

A configuration of a kC-OCA (kC-OCA$_t$, kC-CA) at some time step $t \geq 0$ is a pair (c_t, w_t) where $w_t \in \Sigma^*$ denotes the remaining input and c_t is a description of the k cell states, formally a mapping $c_t : \{1, \ldots, k\} \to Q \cup \{\sqcup\}$. We consider the input string $u = u_1 \ldots u_n$: The initial configuration at time 0 is defined by $c_0(i) = \sqcup$, $1 \leq i \leq k$ and $w_0 = u$.

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

For kC-OCAs:

$$
\begin{aligned}
(c_{t+1}, w_{t+1}) \;=\; & \Delta(c_t, w_t) \iff \\
& c_{t+1}(i) = \delta(c_t(i), c_t(i+1)), i \in \{1, \dots, k-1\} \\
& c_{t+1}(k) = \delta_r(c_t(k), x)
\end{aligned}
$$

For kC-OCA$_t$s:

$$
\begin{aligned}
(c_{t+1}, w_{t+1}) \;=\; & \Delta(c_t, w_t) \iff \\
& c_{t+1}(i) = \delta(c_t(i), c_t(i+1)), i \in \{1, \dots, k-1\} \\
& c_{t+1}(k) = \delta_r(c_t(k-1), c_t(k), x)
\end{aligned}
$$

For kC-CAs:

$$
\begin{aligned}
(c_{t+1}, w_{t+1}) \;=\; & \Delta(c_t, w_t) \iff \\
& c_{t+1}(1) = \delta(\#, c_t(1), c_t(2)) \\
& c_{t+1}(i) = \delta(c_t(i-1), c_t(i), c_t(i+1)), i \in \{2, \dots, k-1\} \\
& c_{t+1}(k) = \delta_r(c_t(k-1), c_t(k), x)
\end{aligned}
$$

where $x = \nabla$ and $w_{t+1} = \epsilon$ if $w_t = \epsilon$, and $x = x_1$ and $w_{t+1} = x_2 \dots x_n$ if $w_t = x_1 x_2 \dots x_n$. Thus, Δ is induced by δ_r and δ.

An input string u is accepted by a kC-OCA (kC-OCA$_t$, kC-CA) if at some time step during its computation the leftmost cell enters an accepting state from the set of accepting states $F \subseteq Q$.

Definition: Let $A = (Q, \Sigma, \sqcup, \nabla, k, \delta_r, \delta, F)$ be a kC-OCA (kC-OCA$_t$, kC-CA).

1. A string $u \in \Sigma^+$ 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 $(c_i, w_i) = \Delta^i((c_0, u))$.
2. $T(A) = \{u \in \Sigma^+ \mid u \text{ is accepted by } A\}$ is the language accepted by A.
3. If all $u \in T(A)$ are accepted within $|u|+k$ time steps, we say that A accepts in realtime. $\mathcal{L}(k\text{C-OCA}) = \{L \mid L \text{ is accepted by a realtime-}k\text{C-OCA}\}$. $\mathcal{L}(k\text{C-OCA}_t)$ and $\mathcal{L}(k\text{C-CA})$ are defined analogously.

We investigate in this paper the descriptional systems DFA, kC-OCA, kC-OCA$_t$, and kC-CA. As descriptional complexity measure for these automata we count the number of states. Since a kC-OCA (kC-OCA$_t$, kC-CA) is composed of k identical cells, this measure is reasonable. The definitions of upper and lower bounds follow the presentation in [1].

We say that a function $f : \mathbb{N} \to \mathbb{N}$, $f(n) \geq n$ is an *upper bound* for the blow-up in complexity when changing from one descriptional system D_1 to another system D_2, if every description $M \in D_1$ of size n has an equivalent description $M' \in D_2$ of size at most $f(n)$.

We say that a function $g : \mathbb{N} \to \mathbb{N}$, $g(n) \geq n$ is a *lower bound* for the trade-off between two descriptional systems D_1 and D_2, if there is an infinite sequence $(L_i)_{i \in \mathbb{N}}$ of pairwise distinct languages L_i such that for all $i \in \mathbb{N}$ there is a description $M \in D_1$ for L_i of size n and every description $M' \in D_2$ for L_i is at least of size $g(n)$.

3 Comparing kC-CAs and kC-OCA$_t$ with DFAs

In [6], Lemma 2 it is shown that every n-state kC-OCA can be converted to an equivalent DFA having $O(n^k)$ states. The idea of the construction is to build the Cartesian product of k cells and to define the accepting states suitably. It can be easily observed that the same construction can be used for kC-CAs and kC-OCA$_t$s. The only difference is to define the transition function of the DFA according to the new transition functions δ and δ_r. However, this can be realized without increasing the number of states. Lemma 4 in [6] shows that every kC-OCA accepting $L_{n,k} = \{a^m \mid m \geq n^k + n^{k-1} + 1\}$ needs at least $n+1$ states. The essential observation is that $n^k + n^{k-1} + 1$ distinct configurations on k cells have to be distinguished which requires $n+1$ states. The same reasoning holds for kC-CAs and kC-OCA$_t$, because the ability of cells to see its left neighbor obviously does not reduce the number of configurations which have to be distinguished. Hence we obtain the following two lemmas:

Lemma 1 *Every n-state kC-CA (kC-OCA$_t$) can be converted to an equivalent DFA having $O(n^k)$ states.*

Lemma 2 *Every kC-CA (kC-OCA$_t$) accepting $L_{n,k} = \{a^m \mid m \geq n^k + n^{k-1} + 1\}$ needs at least $n+1$ states.*

It is known [6] that an n-state DFA can be converted to an equivalent $(n+1)$-state kC-OCA. This bound is known to be tight, i.e., there are languages where the parallelism provided in terms of additional cells does not help to reduce the size of description. This situation changes in case of kC-CAs and kC-OCA$_t$s. Here, these models can always achieve savings in size when compared to DFAs.

Lemma 3 *Every n-state DFA M can be converted to a kC-OCA$_t$ (kC-CA) A such that $T(A) = T(M)$ and $|A| \leq \lceil\sqrt{n}\rceil(|\Sigma|+1) + 2|\Sigma| + 1 = O(\sqrt{n})$ where $T(M) \subseteq \Sigma^*$.*

Proof: Let M be an n-state DFA accepting a language over the alphabet Σ. Let Q denote the set of states, $F \subseteq Q$ the set of accepting states, 0 the initial state, and δ the transition function. We construct a kC-OCA$_t$ by simulating M in the last two cells from the right. In detail, the state set Q is encoded by two bits of a $\lceil\sqrt{n}\rceil$-ary alphabet. This encoding is then used to compute the first bit of M's actual state in the last but one cell and the second bit in the rightmost cell, respectively. After reading the input, we check whether an accepting state of M has been computed in the last two cells and we then send an accepting state with maximum speed to the left. Otherwise, the computation is blocked.

Let $Q = \{0, 1, 2, \ldots, n-1\}$ and $\ell = \lceil\sqrt{n}\rceil - 1$. Let

$$Q_c = \{00, 01, 02, \ldots, 0\ell, 10, 11, 12, \ldots, 1\ell, \ldots, \ell 0, \ell 1, \ell 2, \ldots, \ell\ell\}$$

If $|Q_c| > |Q|$, we delete some endmost elements of Q_c so that $|Q_c| = |Q|$. Let $\phi : Q \to Q_c$ be any, but fixed bijection between Q and Q_c such that $\phi(0) = 00$. Let

$[\cdot]_1, [\cdot]_2 : Q \to \{0, 1, 2, \ldots, \ell\}$ define projections from states $q \in Q$ on the first and second bits of their encodings, respectively. E.g.: $q = 1, \phi(q) = 01, [q]_1 = 0, [q]_2 = 1$.

Now, $Q' = (\{0, 1, 2, \ldots, \ell\} \times (\Sigma \cup \{h\})) \cup \{g\} \cup \Sigma \cup \Sigma'$ where Σ' is a "primed" version of Σ and $\{g, h\} \cap (\{0, 1, 2, \ldots, \ell\} \cup \Sigma \cup \Sigma') = \emptyset$. Let $\psi : \Sigma \to \Sigma'$ be the bijection defined by $\psi(\sigma) = \sigma'$ for all $\sigma \in \Sigma$.

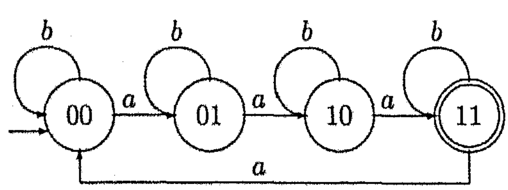
Let $A = (Q', \Sigma, \sqcup, \triangledown, k, \delta'_r, \delta', \{g\})$ and for $i, j \in \{0, 1, 2, \ldots, \ell\}$, $\sigma, \tau, \tau_l, \tau_r \in \Sigma$, and $\tau', \tau'_r \in \Sigma'$:

$$\begin{aligned}
\delta'_r(\sqcup, \sqcup, \sigma) &= \sigma && (1)\\
\delta'_r(\sqcup, \tau, \sigma) &= \psi(\sigma) && (2)\\
\delta'_r((i, \tau_l), \tau', \sigma) &= ([\delta(0, \tau_l)]_2, \sigma) && (3)\\
\delta'_r((j, \tau_l), (i, \tau), \sigma) &= ([\delta(\phi^{-1}(j, i), \tau_l)]_2, \sigma) && (4)\\
\delta'_r(\sqcup, \tau, \triangledown) &= \begin{cases} g & : \ \delta(0, \tau) \in F \\ \tau & : \ \text{otherwise} \end{cases} && (5)\\
\delta'_r((i, \tau_l), \tau', \triangledown) &= ([\delta(\phi^{-1}(i, 0), \tau_l)]_2, h) && (6)\\
\delta'_r((j, \tau_l), (i, \tau), \triangledown) &= ([\delta(\phi^{-1}(j, i), \tau_l)]_2, h) && (7)\\
\delta'(\sqcup, \tau) &= (0, \tau) && (8)\\
\delta'((0, \tau), \tau'_r) &= ([\delta(0, \tau)]_1, \psi^{-1}(\tau'_r)) && (9)\\
\delta'((i, \tau), (j, \tau_r)) &= ([\delta(\phi^{-1}(i, j), \tau)]_1, \tau_r) && (10)\\
\delta'((i, \tau), (j, h)) &= \begin{cases} g & : \ \delta(\phi^{-1}(i, j), \tau) \in F \\ (i, \tau) & : \ \text{otherwise} \end{cases} && (11)\\
\delta'(q, g)) &= g; \ q \in Q' \cup \{\sqcup\} && (12)\\
\delta'(\sqcup, (i, \tau_r)) &= (i, \tau_r) && (13)
\end{aligned}$$

The remaining transitions are undefined. Rather than to give a formal proof of $T(A) = T(M)$, we refer to the following Example 1. The number of states in Q' may be estimated as: $|Q'| \leq \lceil \sqrt{n} \rceil (|\Sigma| + 1) + 2|\Sigma| + 1 = O(\sqrt{n})$.

The construction for kC-CAs is essentially the same by ignoring any information from the left in all cells except the rightmost cell. $\square$

Example 1: Consider the following DFA M:



A 4C-OCA$_t$ works as follows. The last two cells are initialized in the first two time steps. Then the binary encoding of M's states can be found in the first component of the last two cells. After reading the end-of-input symbol for the first time the rightmost cell is marked with a special symbol h. In the next time step the last but one cell processes the last input symbol stored in its second component and sends an accepting state g to the left if an accepting state of M has been computed; otherwise it remains in its state and the computation is blocked. The processing of $aaab$ and $aababa$ may be found in the following tables.

$\sqcup$	$\sqcup$	$\sqcup$	$\sqcup$	$aaab$
$\sqcup$	$\sqcup$	$\sqcup$	a	aab
$\sqcup$	$\sqcup$	$(0,a)$	a'	ab
$\sqcup$	$(0,a)$	$(0,a)$	$(1,a)$	b
$(0,a)$	$(0,a)$	$(1,a)$	$(0,b)$	ϵ
$(0,a)$	$(1,a)$	$(1,b)$	$(1,h)$	ϵ
$(1,a)$	$(0,b)$	g	$(1,h)$	ϵ
$(1,b)$	g	g	$(1,h)$	ϵ
g	g	g	$(1,h)$	ϵ

$\sqcup$	$\sqcup$	$\sqcup$	$\sqcup$	$aababa$
$\sqcup$	$\sqcup$	$\sqcup$	a	$ababa$
$\sqcup$	$\sqcup$	$(0,a)$	a'	$baba$
$\sqcup$	$(0,a)$	$(0,a)$	$(1,b)$	aba
$(0,a)$	$(0,a)$	$(1,b)$	$(0,a)$	ba
$(0,a)$	$(1,b)$	$(1,a)$	$(0,b)$	a
$(1,b)$	$(1,a)$	$(1,b)$	$(1,a)$	ϵ
$(1,a)$	$(0,b)$	$(1,a)$	$(1,h)$	ϵ
$(1,b)$	$(0,a)$	$(1,a)$	$(1,h)$	ϵ
$(1,a)$	$(1,a)$	$(1,a)$	$(1,h)$	ϵ
$(0,a)$	$(0,a)$	$(1,a)$	$(1,h)$	ϵ

The next lemma says that the construction given in Lemma 3 is in a way optimal for kC-OCA$_t$s, since the upper bound is proven to be nearly tight. The same reasoning provides a lower bound for kC-CAs. Surprisingly, the lower bound does not depend on the number of cells. That is, whatever number of cells is provided, there are regular languages where at most savings of $O(\sqrt[3]{n/\log n})$ can be achieved. It is an open question whether the construction of Lemma 3 can be improved to $O(\sqrt[3]{n})$ or, alternatively, whether the proof for kC-CAs can be refined to show a lower bound of $O(\sqrt{n})$.

Lemma 4 *There is an infinite sequence of languages $(L_n)_{n \in \mathbb{N}}$ such that each L_n is accepted by an $(n+1)$-state DFA. Every kC-OCA$_t$ accepting L_n needs at least $\Omega(\sqrt{n/\log n})$ states and every kC-CA accepting L_n needs at least $\Omega(\sqrt[3]{n/\log n})$ states.*

Proof: We show that there is a constant c depending on k such that for every $n \in \mathbb{N}$ with $n \geq c$ holds: there is a singleton language $L_n = \{x\}$ with $|x| = n$ that is accepted by a DFA having $n+1$ states and every kC-OCA$_t$ accepting L_n needs at least m states with $m + 1 \geq \sqrt{\frac{n}{(\log n)(|\Sigma|+1)}}$.

In the following we are using an incompressibility argument. More general information on Kolmogorov complexity and the incompressibility method may be found in [8]. Let L_n be a singleton of length n and A a kC-OCA$_t$ accepting L_n. Then $C(L_n|n)$ denotes the minimal size of a program describing L_n and knowing the length n. It is easy to see that the size of this minimal description is lower than or equal to the size of a certain encoding *cod*(A) of A and the size $|P|$ of a program P which describes how a kC-OCA$_t$ is encoded and how a kC-OCA$_t$ describes L_n. Obviously, $|P|$ does not depend on L_n, A, and n. An encoding *cod*(A) of A consists of encodings *cod*(δ) and *cod*(δ_r) of its transition functions, an encoding *cod*(F) of the accepting states, and an encoding *cod*(k) of k.

We choose $n \in \mathbb{N}$ such that $|P| \leq (1/8)n$ and $\log k \leq (1/8)n$. According to Theorem 2.3. of [8] we know that there exists a string x of length n such that $C(x|n) \geq n$.

We set $L_n = \{x\}$ and assume by way of contradiction that there exists a kC-OCA$_t$ A accepting L_n with m states and $m+1 < \sqrt{\frac{n}{(\log n)(|\Sigma|+1)}}$.

Since $|\Sigma| \geq 2$, we can conclude the following two inequalities which will be needed below.

$$(m+1)^2 \log m \leq (1/3)(m+1)^2(|\Sigma|+1)\log m,$$
$$(m+1)\log m \leq (1/6)(m+1)^2(|\Sigma|+1)\log m$$

$$\begin{aligned}
|cod(A)| + |P| &\leq \underbrace{\log m^{(m+1)m}}_{|cod(\delta)|} + \underbrace{\log m^{(m+1)^2(|\Sigma|+1)}}_{|cod(\delta_r)|} + \underbrace{m \log m}_{|cod(F)|} + \underbrace{\log k}_{|cod(k)|} + |P| \\
&\leq (m+1)m\log m + (m+1)^2(|\Sigma|+1)\log m + m\log m + \frac{1}{4}n \\
&\leq (m+1)^2\log m + (m+1)^2(|\Sigma|+1)\log m + (m+1)\log m + \frac{1}{4}n \\
&\leq (\frac{1}{3} + 1 + \frac{1}{6})(m+1)^2(|\Sigma|+1)\log m + \frac{1}{4}n \\
&= \frac{3}{2}(m+1)^2(|\Sigma|+1)\log m + \frac{1}{4}n \\
&< \frac{3}{2}\frac{n}{(\log n)(|\Sigma|+1)}(|\Sigma|+1)\log\left(\left(\frac{n}{(\log n)(|\Sigma|+1)}\right)^{1/2}\right) + \frac{1}{4}n \\
&= \frac{3}{4}\frac{n}{\log n}\log\left(\frac{n}{(\log n)(|\Sigma|+1)}\right) + \frac{1}{4}n \\
&= \frac{3}{4}\frac{n}{\log n}(\log n - \log\log n - \log(|\Sigma|+1)) + \frac{1}{4}n \\
&= \frac{3}{4}n\left(1 - \frac{\log\log n}{\log n} - \frac{\log(|\Sigma|+1)}{\log n}\right) + \frac{1}{4}n \\
&\leq \frac{3}{4}n + \frac{1}{4}n = n
\end{aligned}$$

Hence, $C(L_n|n) \leq |cod(A)| + |P| < n$ which is a contradiction to $C(L_n|n) \geq n$.

The reasoning for kC-CAs is quite the same by considering $m+1 < \sqrt[3]{\frac{n}{(\log n)(|\Sigma|+1)}}$, estimating $|cod(\delta)| \leq \log m^{(m+1)^2 m}$, and replacing $(m+1)^2$ by $(m+1)^3$. □

As consequence of Lemma 1 and Lemma 3 we obtain:

Theorem 1 $\mathcal{L}(kC\text{-}OCA_t) = \mathcal{L}(kC\text{-}CA) = REG$.

4 Comparing kC-OCAs, kC-OCA$_t$s and kC-CAs

4.1 Embedding

Lemma 5 *Every n-state kC-OCA can be converted to an equivalent n-state kC-CA or kC-OCA$_t$. Every n-state kC-OCA$_t$ can be converted to an equivalent n-state kC-CA. All bounds are tight.*

Proof: Obviously, a kC-CA or kC-OCA$_t$ can simulate a kC-OCA without increasing the number of states. By the same token, a kC-CA can simulate a kC-OCA$_t$. In Lemma 2 it is shown that any kC-OCA accepting $L_{n,k}$ needs at least $n+1$ states, since $n^k + n^{k-1} + 1$ states have to be distinguished with k cells. The same reasoning holds for kC-OCA$_t$s and kC-CAs, respectively. Hence, every kC-OCA$_t$ and kC-CA accepting $L_{n,k}$ needs at least $n+1$ states. □

4.2 Upper Bounds

Lemma 6

(a) Every n-state kC-CA can be converted to a kC-OCA with at most $O(n^k)$ states.

(b) Every n-state kC-OCA$_t$ can be converted to a kC-OCA with at most n^2+n states.

(c) Every n-state kC-CA can be converted to a kC-OCA$_t$ with at most $O(n^{\lceil k/2 \rceil})$ states.

Proof: (a) An n-state kC-CA can be converted to a DFA having $O(n^k)$ states due to Lemma 1. This DFA can be converted to a kC-OCA having $O(n^k)$ states applying Lemma 1 from [6]. (b) Let A be an n-state kC-OCA$_t$. We construct a kC-OCA A' which is essentially the same as A except that the rightmost cell has to simulate the last but one cell. Therefore, we consider the Cartesian product of two cells in the rightmost cell. A' works the same way as A except that the last but one cell is additionally simulated in the first component of the rightmost cell. It is easy to see that $T(A') = T(A)$ and $|A'| \leq n^2 + n$. (c) Let A be an n-state kC-CA. To construct a kC-OCA$_t$ A', we have to simulate k cells in the last two cells. Therefore, we consider the Cartesian product of $\lceil k/2 \rceil$ cells in every cell. In the rightmost cell of A', the last $\lceil k/2 \rceil$ cells are simulated and the first $k - \lceil k/2 \rceil$ cells in the last but one cell. Now, A' works the same way as A until the end-of-input symbol is read the first time by the rightmost cell. At this moment, A' checks whether the actual configuration of A, which is encoded in the last two cells, leads to an accepting state in the first cell within the next k time steps when computed in A. If so, an accepting state is sent with maximum speed to the left, otherwise the computation is blocked. It is easy to verify that $T(A') = T(A)$ and $|A'| \leq (n+1)^{\lceil k/2 \rceil} + 2 = O(n^{\lceil k/2 \rceil})$. □

4.3 Lower Bounds

In this section we consider the languages $L_p = \{a^n \mid n \equiv 0 \mod p\}$ where p is a prime number. It is shown in [7] that every kC-OCA accepting $L_p \cdot \{a\}$ needs at least $p+1$ states. The first part of the proof can be easily adapted to show that every kC-OCA accepting L_p needs at least p states.

Lemma 7 *Every kC-OCA accepting L_p needs at least p states.*

The following result from the theory of numbers may be found in [2]:

Theorem 2 (Bertrand's Postulate) *If $n \geq 1$, there is at least one prime p such that $n < p \leq 2n$.*

Let $(n_m)_{m\in\mathbb{N}}$ be an infinite sequence of natural numbers such that $2n_i < n_{i+1}$ for all $i \geq 1$. This implies $n_i^k < 2n_i^k < n_{i+1}^k$ for all $i \geq 1$. Due to Theorem 2, there exists a prime number p_i such that $n_i^k < p_i < 2n_i^k < n_{i+1}^k$ for all $i \geq 1$. Thus, there exists an infinite sequence of prime numbers $(p_m)_{m\in\mathbb{N}}$ such that $n_i^k < p_i < 2n_i^k < n_{i+1}^k$ for all $i \geq 1$ and $k \geq 2$.

Lemma 8 *Every language L_{p_i} can be accepted by a kC-CA having $O(n_i)$ states.*

Proof: We know that $n_i^k < p_i < 2n_i^k$ due to the above considerations. Let $p_i = n_i^k + r$ with $1 \leq r < n_i^k$. The rough idea is as follows. We construct an n_i-ary counter. After $n_i^k + k - 1$ time steps, the leftmost cell gets a carry-over. Then, at every time step, the leftmost cell starts a signal from left to right that checks whether r has been counted. If so, the counter is reset and the next counting of $n_i^k + r$ starts; otherwise the signal is canceled. If the input is read and $n_i^k + r$ is counted, the input is accepted, otherwise the input is rejected. To be more precise, let A be an n_i-ary counter. A construction may be found in [6]. Let $c_t(j)$ denote the state of the j-th cell after reading a^t. We now construct a kC-CA A' where each cell is split into two subcells, so we can speak of two tracks. On the first track we install an n_i-ary counter. Let $c^1(j)$ denote the state of the first track of the j-th cell. We have to distinguish two cases. At first, we consider the case $r \geq 2k-1$: After $n_i^k + k - 1$ steps the first cell gets a carry-over; from this time the second track is used to check whether r has been counted. In detail, a signal is started which successively checks whether $c^1(1) = c_{n_i^k+r-(k-1)}(1)$, $c^1(2) = c_{n_i^k+r-(k-1)+1}(2)$, $c^1(3) = c_{n_i^k+r-(k-1)+2}(3)$, $\ldots, c^1(k) = c_{n_i^k+r-(k-1)+(k-1)}(k) = c_{n_i^k+r}(k)$. As soon as one of the above equations does not hold, the signal is stopped. If all equations hold, then the signal has arrived at the rightmost cell and r has been counted. If the next input is $\triangledown$, then an accepting state is sent with maximum speed to the left, otherwise we send a signal with maximum speed to the left which resets the counter on the first track and stops the emitting of signals from the leftmost cell. Then, the automaton works as described and starts the next counting of $n_i^k + r$.

The case $r < 2k - 1$ is more complicated. The construction is identical to the above construction until the leftmost cell gets a carry-over. Then a signal R from left to right is started at every time step successively checking whether $c^1(1) = c_{n_i^k}(1)$, $c^1(2) = c_{n_i^k+1}(2)$, $\ldots, c^1(k) = c_{n_i^k+2k-1}(k)$. If R arrives at the rightmost cell and the next input is $\triangledown$, then an accepting state is sent with maximum speed to the left, otherwise we send a signal I with maximum speed to the left which initializes the counter with $2k - r$ and starts the next computation. One special case remains to be treated. If the rightmost cell reads the first end-of-input symbol at some time t before R has been arriving at the rightmost cell, then a signal L from right to left is initialized successively checking the following equations: $c_t^1(k) = c_{n_i^k+r}(k)$, $c_{t+1}^1(k-1) = c_{n_i^k+r+1}(k-1)$, $\ldots, c_{t+k}^1(1) = c_{n_i^k+r+k-1}(1)$. L is canceled as soon as one of these equations does not hold. If L meets R, then an accepting state is sent with maximum speed to the left.

We now have to sum up the number of states used in the construction. The counting can be realized with $n_i + 1$ states and the signals R, L, and I need at most $k + 1$ states each. Thus, the size of the kC-CA is at most $(n_i+1)(2k+2)+k+2 = O(n_i)$. □

Lemma 9

(a) Every kC-OCA accepting L_{p_i} needs at least $\Omega(n_i^k)$ states.

(b) Every kC-OCA$_t$ accepting L_{p_i} needs at least $\Omega(n_i^{k/2})$ states.

Proof: Due to Lemma 7 we know that every kC-OCA accepting L_{p_i} needs at least $p_i = n_i^k + r$ states with $1 \leq r \leq n_i^k$. Hence, $p_i = \Omega(n_i^k)$. L_{p_i} is accepted by a kC-CA with $O(n_i)$ states. Assume by way of contradiction that L_{p_i} is accepted by a kC-OCA$_t$ with $n = o(n_i^{k/2})$ states. Due to the construction of Lemma 6(b), then there exists a kC-OCA accepting L_{p_i} with $n^2 + n = o(n_i^k)$ states which is a contradiction to (a). □

Theorem 3 *The following table holds. An entry in column A and row B describes the upper and lower bounds when converting type-A automata to type-B automata.*

	DFA	*kC-OCA*	*kC-OCA$_t$*	*kC-CA*
DFA	—	$O(n^k)$ (a) $\Omega(n^k)$	$O(n^k)$ (a) $\Omega(n^k)$	$O(n^k)$ (a) $\Omega(n^k)$
kC-OCA	$\leq n+1$ (b) $\geq n+1$	—	n^2+n (c) $\Omega(n^2)$	$O(n^k)$ (d) $\Omega(n^k)$
kC-OCA$_t$	$O(\sqrt{n})$ (e) $\Omega(\sqrt{n/\log n})$	$\leq n$ (f) $\geq n$	—	$O(n^{\lceil k/2\rceil})$ (d) $\Omega(n^{k/2})$
kC-CA	$O(\sqrt{n})$ (e) $\Omega(\sqrt[3]{n/\log n})$	$\leq n$ (f) $\geq n$	$\leq n$ (f) $\geq n$	—

Proof: (*a*) follows from Lemma 1 and Lemma 2. (*b*) follows from Lemma 1 in [7] and Lemma 7. (*d*) follows from Lemma 6(a), Lemma 8(a), Lemma 9(a). (*e*) follows from Lemma 3 and Lemma 4. (*f*) follows from Lemma 5. The upper bound in (*c*) follows from Lemma 6(b). Combining Lemma 8 and Lemma 3, we can construct a kC-OCA$_t$ accepting L_{p_i} with $n = O(n_i^{k/2})$ states. An equivalent kC-OCA needs $p_i = \Omega(n^2)$ states. □

5 Conclusion

We studied the descriptional complexity of cellular automata with a fixed number of cells and several amounts of two-way communication ranging from no two-way communication (kC-OCAs) to maximum two-way communication (kC-CAs). kC-OCA$_t$s are an intermediate model with minimum two-way communication. All models describe the regular languages. We showed that the conversion to DFAs implies a polynomial blow-up of degree k and this upper bound was shown to be tight. On the other hand, the conversion of DFAs to cellular automata may provide no savings in case of one-way communication, but always provides quadratic savings in case of two-way communication. The latter bound was additionally shown to be nearly tight for kC-OCA$_t$s. Furthermore, we showed bounds which are tight in order of magnitude when converting kC-OCA$_t$s or kC-CAs to kC-OCAs. In the latter case, maximum two-way communication may provide polynomial savings of degree k in contrast to one-way communication. Some open problems result from our considerations. Lemma 4 showed that there are languages such that kC-CAs permit at most cubic savings. It should be investigated whether the upper bound can be improved such that kC-CAs always achieve cubic savings. Since we have studied here two models with a minimum and a maximum amount of two-way communication, it could be interesting to investigate how the size of description of a language varies when gradually more and more cells are provided with two-way communication.

References

[1] J. Goldstine, M. Kappes, C.M.R. Kintala, H. Leung, A. Malcher, D. Wotschke: "Descriptional complexity of machines with limited resources," Journal of Universal Computer Science, 8(2): 193-234, 2002

[2] G.H. Hardy, E.M. Wright: "An Introduction to the Theory of Numbers," Clarendon Press, Oxford, 1988

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

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

[5] A. Malcher: "Descriptional complexity of cellular automata and decidability questions," Journal of Automata, Languages and Combinatorics, 7(4): 549-560, 2002

[6] A. Malcher: "On one-way cellular automata with a fixed number of cells," In J. Dassow, M. Hoeberechts, H. Jürgensen, D. Wotschke (Eds.): "Fourth International Workshop on Descriptional Complexity of Formal Systems (DCFS 2002)," Report No. 586, Department of Computer Science, The University of Western Ontario, London, Ontario, Canada, 2002

[7] A. Malcher: "On one-way cellular automata with a fixed number of cells," submitted for journal publication, 2003

[8] M. Li, P. Vitányi: "An Introduction to Kolmogorov Complexity and Its Applications," Springer-Verlag, New York, 1993

[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

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

1/2002 Paetz, Jürgen:
Durchschnittsbasierte Generalisierungsregeln; Teil I: Grundlagen

2/2002 Paetz, Jürgen; Brause, Rüdiger:
Durchschnittsbasierte Generalisierungsregeln Teil II: Analyse von Daten septischer Schock-Patienten

3/2002 Nießner, Frank:
Decomposition of Deterministic ω-regular Liveness Properties and Reduction of Corresponding Automata

4/2002 Kim, Pok-Son:
Das $\mathcal{RSV}$-Problem ist $\mathcal{NP}$-vollständig

5/2002 Nebel, Markus E.:
On a Statistical Filter for RNA Secondary Structures

6/2002 Malcher, Andreas:
Minimizing Finite Automata is Computationally Hard

1/2003 Malcher, Andreas:
On One-Way Cellular Automata with a Fixed Number of Cells

2/2003 Malcher, Andreas:
On Two-Way Communication in Cellular Automata with a Fixed Number of Cells