Johann Wolfgang Goethe-Universität
Frankfurt am Main

Institut für Informatik
Fachbereich Biologie und Informatik

On One-Way Cellular Automata with a Fixed Number of Cells

Andreas Malcher

Nr. 1/03

Frankfurter Informatik-Berichte

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

16–9107

On One-Way 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

We investigate a restricted one-way cellular automaton (OCA) model where the number of cells is bounded by a constant number k, so-called kC-OCAs. In contrast to the general model, the generative capacity of the restricted model is reduced to the set of regular languages. A kC-OCA can be algorithmically converted to a deterministic finite automaton (DFA). The blow-up in the number of states is bounded by a polynomial of degree k. We can exhibit a family of unary languages which shows that this upper bound is tight in order of magnitude. We then study upper and lower bounds for the trade-off when converting DFAs to kC-OCAs. We show that there are regular languages where the use of kC-OCAs cannot reduce the number of states when compared to DFAs. We then investigate trade-offs between kC-OCAs with different numbers of cells and finally treat the problem of minimizing a given kC-OCA.

1 Introduction

The descriptional complexity of abstract machines is a field of theoretical computer science which has attracted the attention of many researchers in the last thirty years. The central question is: How succinctly can a model represent a formal language in comparison with other models? Regarding regular languages, it is known that each nondeterministic finite automaton (NFA) having n states can be converted by the subset construction to an equivalent deterministic finite automaton (DFA) with at most 2^n states. In [7] is shown that this upper bound is tight, since there exists an infinite sequence of regular languages $(L_n)_{n\geq 1}$ such that each L_n is recognized by an n-state NFA and each equivalent DFA needs at least 2^n states. In [1] a survey of results on the descriptional complexity of machines from the vantage point of limited resources is given.

In a preceding paper [5] some research was started on the descriptional complexity of cellular automata which are a parallel model of computation. 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 itself 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 itself and the neighboring cell to the right. Furthermore, the available time to process the input is bounded by the length of the input. If the available time is a constant multiple of the length of the input, we say that the automaton works in linear time.

Apart from exponential trade-offs between descriptional systems, e.g., the above-mentioned exponential blow-up between NFAs and DFAs, or, more generally, trade-offs which are bounded by a recursive function, it is known that there are trade-offs between descriptional systems that are not bounded by any recursive function, so-called non-recursive trade-offs. They were first studied in [7] on the basis of the trade-off between context-free grammars and DFAs. In [5] it was possible to prove such non-recursive trade-offs between realtime-OCAs and sequential models like DFAs or PDAs. Furthermore, non-recursive trade-offs are shown to exist between realtime-CAs and realtime-OCAs as well as between lineartime-OCAs and realtime-OCAs. The proofs benefit from the fact that the set of valid computations of a Turing machine can be recognized by a realtime-OCA. In addition, this fact has some interesting consequences. For cellular language classes almost all decidability questions as, for example, emptiness, finiteness, inclusion, equivalence, and regularity are undecidable and not even semidecidable. Moreover, it can be shown that for cellular language classes neither exist pumping lemmas nor minimization algorithms.

Thus, the general model turns out to be rather unwieldy and hence we are motivated to look for appropriate restrictions. To accept a formal language by cellular automata, it is required to provide as many cells as the input is long. This is not very realistic from a practical perspective. It is therefore an obvious restriction to limit the number of cells. In this paper, we are going to investigate cellular automata with only a fixed number $k \geq 2$ of cells, so-called kC-OCAs. This limitation has grave consequences on the generative capacity of the restricted model which is reduced to the regular languages (REG). So, kC-OCAs are a parallel model for REG and we investigate the ramifications to their descriptional complexity. We can show that the blow-up in the number of states, when converting a kC-OCA to a DFA, is bounded by a polynomial of degree k. By exhibiting an infinite sequence of unary languages we can show that this upper bound is tight in order of magnitude and we obtain a tight hierarchy concerning the number of states. We then investigate upper and lower bounds when converting DFAs to kC-OCAs and trade-offs between kC-OCAs with different numbers of cells. Finally, we want to address the problem of minimizing a given kC-OCA.

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. 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 [3]. We say that two DFAs or kC-OCAs are equivalent if both accept the same language. Concerning the notations and definitions for kC-OCAs we adapt the notations of the unrestricted model as introduced in [4] to our needs. More detailed information about unrestricted cellular automata may be found in [4].

Definition: A k cells one-way cellular automaton (kC-OCA) is defined as a tuple $A = (Q, \Sigma, \sqcup, \nabla, k, \delta_r, \delta, F)$ 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. $\nabla \notin Q \cup \Sigma$ is the end-of-input symbol,
5. k is the number of cells,
6. $F \subseteq Q$ is the set of accepting cell states and
7. $\delta_r : (Q \cup \{\sqcup\}) \times (\Sigma \cup \{\nabla\}) \to Q \cup \{\sqcup\}$ is the local transition function for the rightmost cell. We require that only the pair $(\sqcup, \nabla)$ 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 kC-OCA works 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 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 ∇ to the rightmost cell after reading the input. To avoid an implicit use of the quiescent state as additional state, it is required that only the pairs $(\sqcup, \sqcup)$ and $(\sqcup, \nabla)$ are mapped to $\sqcup$ by δ_r and δ. Hence the quiescent state can be the state of a cell only within the first k time steps. The size of a kC-OCA $A = (Q, \Sigma, \sqcup, \nabla, k, \delta_r, \delta, F)$ is defined as the number of states in Q, i.e. $|A| = |Q|$. To simplify matters we identify the cells by positive integers.

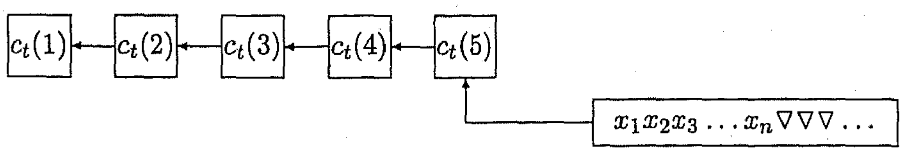

Figure 1: A 5 cells one-way cellular automaton (5C-OCA)

A configuration of a kC-OCA 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 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:

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

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

An input string u is accepted by a kC-OCA 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.

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 is a realtime-kC-OCA. $\mathcal{L}_{rt}(k\text{C-OCA}) = \{L \mid L \text{ is accepted by a realtime-}k\text{C-OCA}\}$.
   $\mathcal{L}_{rt}(k\text{C-OCA}_n)$ is the set of all languages accepted by realtime-kC-OCAs which have at most n states.

In this paper, we consider solely kC-OCAs operating in realtime; thus the terms "realtime-kC-OCA" and "kC-OCA" are used as synonyms.

Example 1: As an example we consider the language

$$L(n, k) = \{a^m \mid m \geq n^k\}$$

and present the construction for $n = 2$ and $k = 4$. The idea is to construct an n-ary counter on k cells where the state $+$ represents a carry-over. If the leftmost cells enters the accepting state $+$, at least n^k input symbols are read and the input is accepted. Let $A = (\{0, 1, +\}, \{a\}, \sqcup, \nabla, 4, \delta_r, \delta, \{+\})$ where

δ	$\sqcup$	0	1	+
$\sqcup$	$\sqcup$	0	0	·
0	·	0	0	1
1	·	1	1	+
+	·	0	0	·

and

δ_r	a	∇
$\sqcup$	1	$\sqcup$
0	·	·
1	+	1
+	1	1

A $\cdot$ indicates that the transition needs not to be defined, since such a situation can never occur on every input. The functionality of the automaton is illustrated with two examples.

1. Input $u = a^{20}$:

$\sqcup$	$\sqcup$	$\sqcup$	$\sqcup$	a^{20}
$\sqcup$	$\sqcup$	$\sqcup$	1	a^{19}
$\sqcup$	$\sqcup$	0	+	a^{18}
$\sqcup$	0	1	1	a^{17}
0	0	1	+	a^{16}
0	0	+	1	a^{15}
0	1	0	+	a^{14}

0	1	1	1	a^{13}
0	1	1	+	a^{12}
0	1	+	1	a^{11}
0	+	0	+	a^{10}
1	0	1	1	a^{9}
1	0	1	+	a^{8}
1	0	+	1	a^{7}

1	1	0	+	a^{6}
1	1	1	1	a^{5}
1	1	1	+	a^{4}
1	1	+	1	a^{3}
1	+	0	+	a^{2}
+	0	1	1	a

After $19 \leq |u| + k = 24$ time steps the first cell enters the accepting state + and the input is accepted.

2. Input $u = a^8$:

$\sqcup$	$\sqcup$	$\sqcup$	$\sqcup$	a^{8}
$\sqcup$	$\sqcup$	$\sqcup$	1	a^{7}
$\sqcup$	$\sqcup$	0	+	a^{6}
$\sqcup$	0	1	1	a^{5}
0	0	1	+	a^{4}
0	0	+	1	a^{3}
0	1	0	+	a^{2}

0	1	1	1	a
0	1	1	+	ϵ
0	1	+	1	ϵ
0	+	0	1	ϵ
1	0	0	1	ϵ
1	0	0	1	ϵ

Here the first cell can never enter the accepting state +; we say that the computation is blocked.

We investigate in this paper the descriptional systems DFA and kC-OCA. As descriptional complexity measure for DFAs and kC-OCAs we count the number of states. Since a kC-OCA 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 $g(n)$. We write:

$$\begin{array}{ccc} D_1 & \longrightarrow & D_2 \\ n & & \leq f(n) \\ n & & \geq g(n) \end{array}$$

3 Generative Capacity of kC-OCAs

Limiting the number of cells to some constant number reduces the generative capacity of kC-OCAs to REG.

Lemma 1 *Every n-state DFA M can be converted to a kC-OCA A such that $T(A) = T(M)$ and $|A| = n+1$.*

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, q_0 the initial state, and δ the transition function. We now construct a kC-OCA by simulating M in the rightmost cell. After reading the input u, an accepting state is sent with maximum speed to the left if $u \in T(M)$, otherwise the computation is blocked.
Formally, let $g \notin Q$ and $Q' = Q \cup \{g\}$. We define $A = (Q', \Sigma, \sqcup, \nabla, k, \delta'_r, \delta', \{g\})$ such that $\delta'_r(\sqcup, \sigma) = \delta(q_0, \sigma)$, $\delta'_r(q, \sigma) = \delta(q, \sigma)$, $\delta'_r(f, \nabla) = g$ and $\delta'_r(p, \nabla) = p$ for $\sigma \in \Sigma$, $q \in Q$, $f \in F$ and $p \in Q \setminus F$, and $\delta'(p, q) = q$ for $p \in Q' \cup \{\sqcup\}$ and $q \in Q'$.
An induction on i shows: $\delta(q_0, u_1u_2 \ldots u_i) = q \Leftrightarrow c_i(k) = q$ and $w_i = \epsilon$.
Hence we can conclude: $u \in T(M) \Leftrightarrow \delta(q_0, u) \in F \Leftrightarrow c_{|u|}(k) \in F$ and $w_{|u|} = \epsilon \Leftrightarrow c_{|u|+1}(k) = g \Leftrightarrow c_{|u|+k}(1) = g \Leftrightarrow u \in T(A)$. □

Lemma 2 *Every n-state kC-OCA A can be converted to a DFA M such that $T(M) = T(A)$ and, if $|\Sigma| > 1$, $|M| \leq n^k + \frac{|\Sigma|^k - 1}{|\Sigma| - 1}$, otherwise $|M| \leq n^k + k$.*

Proof: A DFA accepts an input w if an accepting state is entered after exactly $|w|$ time steps. By definition, an input w is accepted by a kC-OCA if the first cell enters an accepting state. This may happen at some time $t < |w|$ or $|w| \leq t \leq |w| + k$. Hence we have to cope with these two cases when constructing a DFA from a given kC-OCA. The construction can be outlined as follows. At first we construct the Cartesian product of the k cells and we obtain a DFA which accepts a prefix of $w\nabla^k$ if w is accepted by the kC-OCA. Next we modify this DFA so that, if $t < |w|$, the input ends up in an accepting loop. And, if $|w| < t \leq |w| + k$, the set of accepting states is suitably enlarged to accept w.

Let $A = (Q, \Sigma, \sqcup, \nabla, k, \delta_r, \delta, F)$ be a kC-OCA. We define a DFA $M' = (Q', \Sigma', \delta', q'_0, F')$ as follows: $Q' = (Q \cup \{\sqcup\})^k, \Sigma' = \Sigma \cup \{\nabla\}, q'_0 = (\sqcup, \sqcup, \ldots, \sqcup)$ and $F' = F \times Q^{k-1}$.
Let $q_i, q'_i \in Q \cup \{\sqcup\}$ $(1 \leq i \leq k)$ and $\sigma \in \Sigma'$: $\delta'((q_1, q_2, \ldots, q_k), \sigma) = (q'_1, q'_2, \ldots, q'_k)$ such that $q'_1 = \delta(q_1, q_2), q'_2 = \delta(q_2, q_3), \ldots, q'_{k-1} = \delta(q_{k-1}, q_k)$ and $q'_k = \delta_r(q_k, \sigma)$.
Let $w = w_1w_2 \ldots w_n$ and $w\nabla^k = w_1w_2 \ldots w_nw_{n+1} \ldots w_{n+k}$ with $w_{n+l} = \nabla$ $(1 \leq l \leq k)$.
We claim that for $1 \leq i \leq n+k$ and $1 \leq j \leq k$ the following holds:

$$c_i(j) = q \Leftrightarrow \delta'(q'_0, w_1w_2 \ldots w_i) = (q_1, q_2, \ldots, q_k) \text{ such that } q_j = q.$$

This claim can be shown by an induction on i differentiating the two cases $j < k$ and $j = k$.

$$\begin{aligned} w \in T(A) &\Leftrightarrow \exists i \leq |w| + k \text{ such that } c_i(1) \in F \\ &\Leftrightarrow \delta'(q'_0, \overline{w}) = (q_1, q_2, \ldots, q_k) \text{ such that } q_1 \in F \text{ and } \overline{w} \text{ is a prefix of } w\nabla^k \\ &\Leftrightarrow \overline{w} \in T(M') \text{ and } \overline{w} \text{ is a prefix of } w\nabla^k \end{aligned}$$

We now define another DFA $M'' = (Q', \Sigma, \delta'', q'_0, F'')$ having the following properties:

(i) $\delta''(q, \sigma) = \delta'(q, \sigma)$ for $q \in Q' \setminus F'$ and $\sigma \in \Sigma$

(ii) $\delta''(q, \sigma) = q$ for $q \in F'$ and $\sigma \in \Sigma$

(iii) $F'' = F' \cup \overline{F'}$ with $\overline{F'} = \{q \in (Q \setminus F) \times Q^{k-1} \mid \exists 1 \leq l \leq k : \delta'(q, \nabla^l) \in F'\}$

We need the following claim which can be shown by an induction on $|w|$.

Claim: Let $q \in Q'$ and $w \in \Sigma^*$. If $\delta'(q, w') \notin F'$ for all proper prefixes w' of w, then $\delta''(q, w) = \delta'(q, w)$. If $\delta''(q, w') \notin F'$ for all proper prefixes w' of w, then $\delta'(q, w) = \delta''(q, w)$.

We now want to show that $\overline{w} \in T(M')$ and $\overline{w}$ is a prefix of $w\nabla^k \Leftrightarrow w \in T(M'')$.
"$\Rightarrow$": We know that $\overline{w} \in T(M')$ and $\overline{w}$ is a prefix of $w\nabla^k$. W.l.o.g. we may assume that $\overline{w}$ is the shortest prefix of $w\nabla^k$ such that $\overline{w} \in T(M')$. We have to consider two cases:

1. $\overline{w} = w_1 \ldots w_i$ with $i \leq n \Rightarrow \delta'(q'_0, \overline{w}) \in F'$
$\Rightarrow \delta''(q'_0, \overline{w}) \in F'$ (due to Claim 3)
$\Rightarrow \delta''(q'_0, w) \in F'$ (due to property (ii))
$\Rightarrow w \in T(M'')$

2. $\overline{w} = w\nabla^l$ with $1 \leq l \leq k \Rightarrow \delta'(q'_0, w) = q \notin F'$ and $\delta'(q, \nabla^l) \in F'$
$\Rightarrow \delta''(q'_0, w) = q \in \overline{F'}$ (due to Claim 3 and (iii))
$\Rightarrow w \in T(M'')$

"$\Leftarrow$": $w \in T(M'') \Rightarrow \delta''(q'_0, w) \in F'$ or $\delta''(q'_0, w) \in \overline{F'}$.

1. $\delta''(q'_0, w) \in F' \Rightarrow$ there is a shortest prefix $\overline{w}$ of w such that $\delta''(q'_0, \overline{w}) = q \in F'$
$\Rightarrow \delta'(q'_0, \overline{w}) \in F'$ (due to Claim 3)
$\Rightarrow \overline{w} \in T(M')$ and $\overline{w}$ is a prefix of w and therefore of $w\nabla^k$

2. $\delta''(q'_0, w) \in \overline{F'} \Rightarrow \exists q \notin F', l \leq k : \delta''(q'_0, w) = q$ and $\delta''(q, \nabla^l) \in F'$ (l is minimal)
$\Rightarrow \delta'(q'_0, w) = q$ and $\delta'(q, \nabla^l) \in F'$ (due to Claim 3)
$\Rightarrow \delta'(q'_0, w\nabla^l) = \delta'(q'_0, \overline{w}) \in F'$ ($\overline{w} = w\nabla^l$)
$\Rightarrow \overline{w} \in T(M')$ and $\overline{w}$ is a prefix of $w\nabla^k$

This shows that $T(M'') = T(A)$. We now want to compute the number m of reachable states of M''. Due to our definition only the pairs $(\sqcup, \sqcup)$ and $(\sqcup, \nabla)$ are mapped to the quiescent state $\sqcup$ by δ and δ_r, respectively. Therefore, if a cell has entered a state $q \neq \sqcup$, then it will never enter $\sqcup$ again. This fact enables us to count the number of reachable states of Q' where the first l ($1 \leq l \leq k$) components are $\sqcup$. Since there are $|\Sigma|^{k-l}$ different inputs of length $k - l$, there are at most $|\Sigma|^{k-l}$ different states in Q' where it is required that the first l components are $\sqcup$. Let $n = |Q|$. To compute m

we have to sum up all possible states where the first l cells ($1 \leq l \leq k$) are $\sqcup$ and all possible states where each cell is in Q. Hence we have:

$$m \leq \sum_{l=1}^{k} |\Sigma|^{k-l} + n^k = \sum_{l=0}^{k-1} |\Sigma|^l + n^k = \frac{|\Sigma|^k - 1}{|\Sigma| - 1} + n^k$$

We observe that in case of unary alphabets the upper bound is $n^k + k$, since there are only k different inputs of size $k - l$ with $1 \leq l \leq k$. This completes the proof. $\square$

Remark: To obtain an upper bound which does not depend on the size of Σ, we can argue as follows. Since only $(\sqcup, \sqcup)$ and $(\sqcup, \triangledown)$ are mapped to $\sqcup$ and since a cell can never reenter $\sqcup$, for every reachable state $(q_1, q_2, \dots, q_k) \in Q'$ and $1 \leq i \leq k$ holds: $q_i \neq \sqcup \Rightarrow q_j \neq \sqcup$ for all $j > i$. So we can identify the set $\{\sqcup\}^l \times Q^m$, where $l + m = k$, with the set Q^m and have a decomposition of Q' into $Q' = \{q_0'\} \cup Q \cup Q^2 \cup \dots \cup Q^k$. Let $|Q| = n$, so we have $|Q'| = 1 + |Q| + |Q|^2 + \dots + |Q|^k = 1 + n + n^2 + \dots + n^k = \frac{n^{k+1}-1}{n-1} \leq \frac{n}{n-1} n^k$.

The next theorem summarizes the above two lemmas.

Theorem 1 $\mathcal{L}_{rt}(k\,C\text{-}OCA) = REG$

4 A Lower Bound for the Trade-Off

In this section we are going to investigate the family $L_{n,k}$ of unary languages which enables us to show that the upper bound proven in Lemma 2 is tight in order of magnitude. For $n \geq 2$ and $k \geq 2$ let

$$L_{n,k} = \{a^m \,|\, m \geq n^k + n^{k-1}\}$$

Lemma 3 *Each DFA recognizing $L_{n,k}$ needs at least $n^k + n^{k-1} + 1$ states.*

Proof: We use the Nerode equivalence relation $\equiv_{L_{n,k}}$ on $L_{n,k}$ and show that the index of $\equiv_{L_{n,k}}$ exceeds $n^k + n^{k-1} + 1$. For $x, y \in \Sigma^*$, $\equiv_{L_{n,k}}$ is defined as:

$$x \equiv_{L_{n,k}} y :\Longleftrightarrow xz \in L_{n,k} \Leftrightarrow yz \in L_{n,k} \text{ for all } z \in \Sigma^*$$

Let i, j be two integers such that $0 \leq i < j \leq n^k + n^{k-1}$. $a^i a^{n^k+n^{k-1}-i-1} = a^{n^k+n^{k-1}-1} \notin L_{n,k}$ and $a^j a^{n^k+n^{k-1}-i-1} = a^{n^k+n^{k-1}+j-i-1} \in L_{n,k}$, since $j - i - 1 \geq 0$. Hence it follows that $a^i \not\equiv_{L_{n,k}} a^j$ and so we have at least $n^k + n^{k-1} + 1$ pairwise distinct equivalence classes and therefore index$(\equiv_{L_{n,k}}) \geq n^k + n^{k-1} + 1$. $\square$

Lemma 4 *Each kC-OCA recognizing $L_{n,k}$ needs at least $n+1$ states.*

Proof: First of all, we show that there exists a kC-OCA accepting $L_{n,k}$ which has $n+1$ states. Taking a look at the construction of the binary counter in Example 1, which can be generalized to an n-ary counter, we can see that in the rightmost cell a period of length n is counted and that the state 0 is never entered. We modify the construction such that in the rightmost cell a period of length $n+1$ is counted by using the state 0. The transition function δ of Example 1 remains the same and δ_r is modified such that $\delta_r(0,a) = +$ and $\delta_r(1,a) = 0$. It is easy to verify that the modified automaton accepts $L_{n,k}$ and has $n+1$ states. We now want to show that every kC-OCA accepting $L_{n,k}$ needs at least $n+1$ states. Each automaton A must enter $n^k + n^{k-1} + 1$ distinct configurations (including the start configuration $(\sqcup, \ldots, \sqcup)$) within the first $n^k + n^{k-1}$ time steps. Since A has k cells, the assumption that every cell has n states implies that A can enter only $n^k + k$ different configurations according to the considerations in the proof of Lemma 2. This is a contradiction, since $n^k + k \geq n^k + n^{k-1} + 1$ implies $n = 1$. Hence each cell has to be equipped with $n+1$ states, so that at least $n^k + n^{k-1} + 1 \leq (n+1)^k$ distinct configurations can be entered. Therefore we have: $|A| \geq n+1$. □

We summarize our results:

$$
\begin{array}{lll}
k\text{C-OCA} & \longrightarrow & \text{DFA} \\
n & \leq & \begin{cases} n^k + \frac{|\Sigma|^k - 1}{|\Sigma| - 1} = n^k + O(|\Sigma|^{k-1}) & |\Sigma| > 1 \\ n^k + k & |\Sigma| = 1 \end{cases} \\
 & & \\
n & \leq & \frac{n}{n-1} n^k \leq 2n^k = O(n^k) \\
n & \geq & (n-1)^k + (n-1)^{k-1} + 1 = \Omega(n^k)
\end{array}
$$

Although the upper bound is tight only in order of magnitude, we can show the following hierarchy concerning the number of states. Each language recognized by an n-state kC-OCA is trivially recognized by an $(n+1)$-state kC-OCA. But there is a sequence of languages L_n being recognized by an n-state kC-OCA such that no kC-OCA having less than n states can recognize L_n.

Theorem 2

(i) For $k \geq 2$: $\mathcal{L}_{rt}(k\text{C-OCA}_1) = \{\Sigma^*, \emptyset\}$

(ii) For $n \geq 1$ and $k \geq 2$: $\mathcal{L}_{rt}(k\text{C-OCA}_n) \subset \mathcal{L}_{rt}(k\text{C-OCA}_{n+1})$

Proof: Let A be a kC-OCA which has only one state q. If $q \notin F$ then $T(A) = \emptyset$, since the leftmost cell never enters an accepting state. If $q \in F$ then $T(A) = \Sigma^*$, since q is an accepting state and the first cell enters this state after k time steps. This implies (i). For $n \geq 2$ we can conclude from Lemma 4: $L_{n,k} \in \mathcal{L}_{rt}(k\text{C-OCA}_{n+1})$ and $L_{n,k} \notin \mathcal{L}_{rt}(k\text{C-OCA}_n)$. For the remaining case $n = 1$ we show that there is a language which is accepted by a two state kC-OCA, but not by any one state kC-OCA. Hence $\mathcal{L}_{rt}(k\text{C-OCA}_1) \subset \mathcal{L}_{rt}(k\text{C-OCA}_2)$. Let $A = (\{p,q\}, \{a\}, \sqcup, \triangledown, k, \delta_r, \delta, \{q\})$

such that $\delta_r(\sqcup, a) = p$, $\delta_r(p, a) = q$, $\delta_r(p, \nabla) = p$, $\delta_r(q, a) = q$, $\delta_r(q, \nabla) = q$, $\delta(\sqcup, p) = p$, $\delta(p, p) = p$, $\delta(p, q) = q$ and $\delta(q, q) = q$. The remaining transitions are undefined. It is easy to see that $T(A) = \{a^m \mid m \geq 2\}$. Since $T(A) \neq \emptyset$ and $T(A) \neq \{a\}^*$, $T(A)$ is not accepted by any one state kC-OCA due to (i). □

5 Bounds when Converting DFAs to kC-OCAs

It has been shown in Lemma 1 that every n-state DFA can be converted to an $(n+1)$-state kC-OCA. In this section we shall investigate the tightness of this upper bound. Let $p \geq 2$ be a fixed prime number and $L_p = \{a^n \mid n = m \cdot p + 1, m \geq 0\}$.

Lemma 5 *Every kC-OCA accepting L_p needs at least $p+1$ states.*

Proof: Let $A = (Q, \Sigma, \sqcup, \nabla, k, \delta_r, \delta, F)$ be a kC-OCA such that $T(A) = L_p$ and $|A| = n$. For $1 \leq i \leq k$, let $\pi_i : (Q \cup \{\sqcup\})^k \times \Sigma^* \to (Q \cup \{\sqcup\})^{k-i+1}$ define projections as $\pi_i((q_1, q_2, \ldots, q_k), w) = (q_i, q_{i+1}, \ldots, q_k)$. Since the input is unary and A is one-way, it is easy to see that the sequences $s_i = (\pi_i(\Delta^t(c_0, a^t)))_{t \geq 0}$ will become periodical. In detail, s_i will have two identical elements within the first $n^{k-i+1} + k + 1$ elements, because $|A| = n$. Let l_i denote the length of the period between the first occurrence of two identical elements in s_i. We set $p_k = l_k$. Obviously, $l_k = p_k \leq n$. Since $|A| = n$, it follows that $l_{k-1} = p_{k-1}p_k$ with $1 \leq p_{k-1} \leq n$. By the same argument, we have that $l_{k-2} = p_{k-2}p_{k-1}p_k$ with $1 \leq p_{k-2} \leq n$ and, generally, $l_i = p_i p_{i+1} \cdots p_k$ with $1 \leq p_j \leq n$ for $1 \leq i \leq k$ and $i \leq j \leq k$. Then, $l_1 = p_1 p_2 \ldots p_{k-1} p_k$ is the length of the "period of A", because $\Delta^{l_1}(c, a^m) = (c, a^{m-l_1})$ for any configuration (c, a^m) such that $m \geq l_1$ and $c(i) \neq \sqcup$ for $1 \leq i \leq k$.

We now assume that $n < p$. It follows that p does not divide $l_1 = p_1 p_2 \ldots p_k$, since $p_i \leq n < p$ for $1 \leq i \leq k$ and p is prime. We next choose an integer t' such that $t'p+1 > n^k+k+1$. Because $a^{t'p+1} \in L_p$, $\Delta^{t'p+1}(c_0, a^{t'p+1}\nabla^k) = (c', \nabla^k)$ and $\Delta^{k'}(c', \nabla^{k'}) = (c'', \epsilon)$ with $c''(1) \in F$ and $1 \leq k' \leq k$. Let $w = a^{t'p+1+l_1}$. Then, $\Delta^{t'p+1}(c_0, w\nabla^k) = (c', a^{l_1}\nabla^k)$, $\Delta^{l_1}(c', a^{l_1}\nabla^k) = (c', \nabla^k)$, and $\Delta^{k'}(c', \nabla^{k'}) = (c'', \epsilon)$. Hence we know that $w \in L_p$ and therefore is $t'p + 1 + l_1 = t''p + 1$ with $t'' \geq 1$. Thus, $t'p + l_1$ is a multiple of p. This implies that p divides $l_1 = p_1 p_2 \ldots p_k$ which is a contradiction.

We now assume that $n = p$. We observe that there is at least one cell j which enters all p states given a^m $(m \geq n^k + k + 1)$ as input. Otherwise, $p_i < n = p$ for $1 \leq i \leq k$ and it follows that p does not divide $l_1 = p_1 p_2 \ldots p_k$. As above we can conclude that p then divides $l_1 = p_1 p_2 \ldots p_k$ and get a contradiction. It is easy to see that the first cell can enter an accepting state, given $a^m \nabla^k$ $(m \geq 1)$ as input, not before time step $m+k$. Let $a^m \in L_p$ $(m \geq n^k + k + 1)$. After reading ∇ for the first time, the information that the whole input is read must be sent to the leftmost cell and passes cell j at time $m+k-j+1$. Since the information is propagated in terms of a state, let $q \in Q$ denote that state which j enters at time $m + k - j + 1$. Hence, $\Delta^{m+k-j+1}(c_0, a^m\nabla^k) = (c, \nabla^{j-1})$ with $c(j) = q$ and $\Delta^{j-1}(c, \nabla^{j-1}) = (c', \epsilon)$ with $c'(1) \in F$. Let $\pi : (Q \cup \{\sqcup\})^k \to (Q \cup \{\sqcup\})^j$ be the projection defined by $\pi(q_1, q_2, \ldots, q_k) = (q_1, q_2, \ldots, q_{j-1}, q_j)$. We observe that the state q in the cell j leads to an accepting state in the first cell after $j-1$ time steps regardless of the rest of the remaining input. It follows that every $d \in (Q \cup \{\sqcup\})^k$ with $\pi(d) = \pi(c)$ leads to an accepting state in the first cell after $j-1$ time steps regardless of the states of the cells $j+1, \ldots, k$ and the remaining input. Since s_1 is periodical, A is one-way, and the cell j assumes all states in Q, it follows that there is an integer $m' \leq n^k + k$ such that $\Delta^{m'}(c_0, a^{m'}) = (d, \epsilon)$ with $\pi(d) = \pi(c)$. Now, let $m'' \geq j$ be an integer such that $m' + m'' - 1$ is not a multiple of p. Then, $\Delta^{m'}(c_0, a^{m'+m''}) = (d, a^{m''})$ and $\Delta^{j-1}(d, a^{m''}) = (d', a^{m''-j+1})$ with $d'(1) \in F$. Hence, $a^{m'+m''} \in L_p$. This implies that $m' + m'' - 1$ is a multiple of p which is a contradiction. □

Lemma 6 *Every DFA accepting L_p needs at least p states.*

Proof: As in Lemma 3 we will use the Nerode equivalence relation $\equiv_{L_p}$ on L_p. Let i, j be two integers such that $0 \leq i < j \leq p-1$. $a^i a^{p-i+1} = a^{p+1} \in L_p$ and $a^j a^{p-i+1} = a^{p+1+j-i} \notin L_p$, since $0 < j - i < p$. Hence $a^i \not\equiv_{L_p} a^j$ and we have $\text{index}(\equiv_{L_p}) \geq p$. □

Since there are infinitely many prime numbers, we obtain that $g(n) = n$ is a lower bound for the trade-off between DFAs and kC-OCAs. Hence we have:

DFA	$\longrightarrow$	kC-OCA
n		$\leq n+1$
n		$\geq n+1$

This demonstrates that there are languages where the use of a parallel computational model does not help to reduce the size of description in comparison with a sequential model. It should be mentioned that this result does not depend on the particular number of cells k of the kC-OCA. Therefore, these languages are hard to parallelize in the kC-OCA sense, since any "amount of parallelism" employed in terms of additional cells cannot reduce the number of states. The construction in Lemma 1 introduced an additional state g which manages whether the whole input is read or not. The lower bound shows that there are cases in which this additional state is necessary. Thus, some effort in terms of additional states is needed in order to administrate the array of DFAs in contrast to a single DFA.

6 Investigating the Number of Cells

It is very natural to investigate a possible trade-off between kC-OCAs and k'C-OCAs where $k' > k$. How much succinctness can we gain, if the automaton is equipped with more cells? If we enlarge our computational resources, here the number of cells, will there be savings concerning the number of states? And, if so, can these savings be quantified in terms of upper and lower bounds. Comparing kC-OCAs, which only can accept regular languages, with unrestricted OCAs, it is known [5] that in this case the trade-off is not recursively bounded. Unfortunately, we do currently not know whether an n-state kC-OCA can be embedded into an n-state $(k+1)$C-OCA or not. Hence we can give only a partial answer to the above questions.

An obvious try to embed kC-OCAs into $(k+1)$C-OCAs and to preserve the number of states would be to take the old transition function and then to propagate accepting states to the first cell. Unfortunately, this try fails. We take a look at the construction of $L(n,k)$ in Example 1. We observe that a kC-OCA A accepting $L(n,k)$ and a $(k+1)$C-OCA accepting $L(n,k+1)$ have the same transition functions δ_r and δ. Now we want to accept $L(n,k)$ by an n-state $(k+1)$C-OCA A'. If we define A''s transition functions to be those of A, $T(A') \neq L_{n,k}$.

Nevertheless, although we are not able to show whether or not $\mathcal{L}_{rt}((k-1)\text{C-OCA}_n)$ is a proper subset of $\mathcal{L}_{rt}(k\text{C-OCA}_n)$, we can prove $\mathcal{L}_{rt}(k\text{C-OCA}_n) \setminus \mathcal{L}_{rt}((k-1)\text{C-OCA}_n) \neq \emptyset$ provided that $n \geq 4$ and $k \leq n$. In other words, there are some languages such that k is the minimal number of cells which enables an n-state kC-OCA to accept them.

Lemma 7 *For $n \geq 3$ and $2 \leq k \leq n$ holds: $(n+1)^k \leq n^{k+1}$ and $(n+1)^{k-i} \leq n^{k+1-i}$ for $0 \leq i \leq k$.*

Proof: The first claim is proven by induction on n:
Basis: $n=3$, then, $k=2$: $(3+1)^2 = 16 \leq 3^{2+1} = 27$, $k=3$: $(3+1)^3 = 64 \leq 3^{3+1} = 81$.
Induction step: We have to show $(n+2)^k \leq (n+1)^{k+1}$. Due to the binomial formula $(x+y)^k = \sum_{i=0}^{k} \binom{k}{i} x^{k-i} y^i$ we write $(n+2)^k = ((n+1)+1)^k$ and $(n+1)^{k+1}$ as follows:

$$(n+2)^k = (n+1)^k + k(n+1)^{k-1} + \binom{k}{2}(n+1)^{k-2} + \ldots + \binom{k}{k-1}(n+1) + 1 + 0$$

$$(n+1)^{k+1} = n^{k+1} + (k+1)n^k + \binom{k+1}{2}n^{k-1} + \ldots + \binom{k+1}{k-1}n^2 + (k+1)n + 1$$

Since $\binom{k}{i} \leq \binom{k+1}{i}$ for $0 \leq i \leq k$ and by the induction hypothesis, every addend of the upper equation is lower or equal to the equivalent addend of the lower equation. Hence we conclude that $(n+2)^k \leq (n+1)^{k+1}$ and the first inequality is proven. To show the second one we observe that $(n+1)^k \leq n^{k+1} \Leftrightarrow (n+1)^{k-i}(n+1)^i \leq n^i n^{k+1-i} \Leftrightarrow (n+1)^{k-i} \leq \left(\frac{n}{n+1}\right)^i n^{k+1-i} \leq n^{k+1-i}$, since $\frac{n}{n+1} \leq 1$ implies $\left(\frac{n}{n+1}\right)^i \leq 1$. □

Theorem 3 *For $n \geq 4$ and $k \leq n$ there is a language $L(n,k) \in \mathcal{L}_{rt}(k\text{C-OCA}_n)$, but $L(n,k) \notin \mathcal{L}_{rt}(l\text{C-OCA}_n)$ for $l < k$.*

Proof: Let $m = n-1$ and $L(n,k) = L_{n-1,k} = L_{m,k}$. Due to Lemma 4 we know that $L(n,k) \in \mathcal{L}_{rt}(k\text{C-OCA}_n) = \mathcal{L}_{rt}(k\text{C-OCA}_{m+1})$. Since $l < k$, we have $l+1+i = k \Leftrightarrow l = k-i-1$ with $0 \leq i \leq k-2$. We next assume that $L(n,k)$ is accepted by an $(m+1)$-state lC-OCA A. Due to Lemma 2, A can be converted to a DFA M having p cells and p can be estimated as follows.

$$\begin{aligned} p &\leq (m+1)^l + l = (m+1)^{k-i-1} + k - i - 1 < (m+1)^{k-i-1} + k - i \\ &\leq m^{k+1-i-1} + k - i = m^{k-i} + k - i \leq m^k + k \end{aligned}$$

Since $k \leq m^{k-1}$ for $k \geq 2$ and $m \geq 2$, we have a contradiction to Lemma 3 which states that $p \geq m^k + m^{k-1} + 1$. □

7 Minimizing kC-OCAs

In this section we treat the problem of converting an arbitrary kC-OCA to an equivalent kC-OCA which has a minimal number of states. Seidel [8] proves that many decidability questions are undecidable for unrestricted OCAs. The undecidability of the minimization problem for unrestricted OCAs then results from the undecidability of emptiness as is shown in [5]. On the other hand, the minimization problem is solvable in time $O(n \log n)$ for DFAs [2]. Finding a minimization algorithm for kC-OCAs and, if possible, an efficient one, is of particular interest, since this would provide an algorithmic tool to parallelize a given regular language in an optimal way. We refer to the discussion of Open Problem 61 in [6]. We obtain here an intermediate result: kC-OCAs can be algorithmically minimized, but up to now we do not know whether there exists an efficient, i.e. polynomial time minimization algorithm. At first we show that a minimal kC-OCA is, in contrast to DFAs, not necessarily unique.

Theorem 4 *A minimal kC-OCA is not necessarily unique.*

Proof: In Lemma 5 is shown that every kC-OCA accepting L_p needs at least $p+1$ states. We exhibit now two 3-state kC-OCAs with non-isomorphic transition functions both accepting L_2. The generalization to primes $p \geq 3$ is straightforward.

1. We are counting modulo 2 in the rightmost cell. If the input is read and the actual modulus is 1, an accepting state g is sent with maximum speed to the left, otherwise the computation is blocked.
   $A_1 = (\{0,1,g\},\{a\},\sqcup,\nabla,k,\delta_r,\delta,\{g\})$ where

δ	$\sqcup$	0	1	g
$\sqcup$	$\sqcup$	0	1	g
0	·	0	1	g
1	·	0	1	g
g	·	·	·	g

and

δ_r	a	∇
$\sqcup$	1	$\sqcup$
0	1	0
1	0	g
g	·	g

2. The input is shifted into the rightmost cell where a corresponds to 0 and ∇ corresponds to 1. The last but one cell is now counting modulo 2 and acts as the rightmost cell in A_1.
   $A_2 = (\{0,1,g\},\{a\},\sqcup,\nabla,k,\delta_r,\delta,\{g\})$ where

δ	$\sqcup$	0	1	g
$\sqcup$	$\sqcup$	1	·	·
0	·	1	0	g
1	·	0	g	g
g	·	·	·	g

and

δ_r	a	∇
$\sqcup$	0	$\sqcup$
0	0	1
1	·	1
g	·	·

□

Theorem 5 *There exists an algorithm which converts a given kC-OCA A to an equivalent kC-OCA A' such that A' has a minimal number of states.*

Proof: We describe a brute force algorithm. First of all, A is converted to a DFA M according to Lemma 2. Then we list all kC-OCAs $A_1, \ldots, A_m$ such that $|A_i| < |A|$. Now, for each $i \in \{1, \ldots, m\}$, A_i is converted to a DFA M_i and the equality of $T(M)$ and $T(M_i)$ is tested. If there exists no $i \in \{1, \ldots, m\}$ such that $T(M_i) = T(M)$, then A must have been of minimal size already and we return A. Otherwise we have found a finite set $\mathcal{M}$ of equivalent kC-OCAs A_i of smaller size than A. We then choose an automaton $A' \in \mathcal{M}$ of minimal size and return A'. □

8 Conclusion

In this paper, we have put a natural restriction on realtime-OCAs. The generative capacity of the restricted model is reduced to the set of regular languages. We have investigated upper and lower bounds when converting kC-OCAs to DFAs and vice versa. It has been shown that the use of kC-OCAs can lead to polynomial savings of degree k in comparison with DFAs. On the other hand, there are languages which are "inherently sequential" in the kC-OCA sense, since any number of cells employed cannot help to reduce the number of states in comparison with DFAs. We then have studied trade-offs between kC-OCAs with different numbers of cells and finally could state a minimization algorithm for kC-OCAs. The time complexity of the minimization problem is currently unknown. Since a minimal kC-OCA does not have to be necessarily unique, minimization is likely to be a hard computational problem.

One topic of further research could be a more thorough examination of the time complexity of the minimization problem, since an efficient algorithm would be of great practical relevance. Otherwise, if minimization turns out to be computationally hard, suitable restrictions should be studied permitting the design of efficient minimization algorithms. Furthermore, since we have studied here only restrictions on realtime one-way cellular automata, it could be interesting to investigate descriptional complexity aspects of similar restrictions on two-way cellular automata as well as on cellular automata working in linear time.

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] J.E. Hopcroft: "An $n \log n$ algorithm for minimizing states in a finite automaton," In Z. Kohavi (ed.): "Theory of machines and computations," 189–196, Academic Press, New York, 1971

[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] M. Delorme, E. Formenti, J. Mazoyer: "Open problems on cellular automata," Technical Report 2000-25, École Normale Supérieure de Lyon, Lyon, 2000

[7] 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

[8] 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

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