Cellular Automata Workshop 1996

Schloß Rauischholzhausen

March 25–27, 1996

Martin Kutrib, Thomas Worsch (eds.)

Preproceedings

JUSTUS-LIEBIG-
UNIVERSITÄT
GIESSEN

UNIVERSITÄT
KARLSRUHE (TH)

Contents

Program

Monday, 25th March 1996

07:30–09:00 Breakfast

09:00–09:10 Opening Address

09:10–09:45 Rudolf Freund (Technische Universität Wien), *Cellular automata and attributed parallel array grammars*

09:45–10:20 Katsunobu Imai (Hiroshima University), *Firing Squad Synchronization Problem in One and Two Dimensional Reversible Cellular Automata*

10:20–10:55 Gianpiero Cattaneo (Universitá degli Studi di Milano), *Dynamical Classification of CA: Topological Chaos or Lyapunov exponent*

10:55–11:20 Coffee Break

11:20–11:55 André Barbé (Katholieke Universiteit Leuven), *Shift-add correlation patterns of linear cellular automata are pseudo cellular automata*

11:55–12:30 Fritz von Haeseler (Universität Bremen), *(Non)-Automaticity of Orbits of Linear Cellular Automata*

12:30–14:00 Lunch

14:00–14:35 Rolf Hoffmann (Technische Hochschule Darmstadt), *Architectures for Cellular Processing*

14:35–15:10 Ralf Schneider (Technische Hochschule Darmstadt), *ACL – Building Cellular Processing Kernels based on Attributes*

15:10–15:45 Olivier Heen (Université Paris 7), *Constant speed-up on one dimensional calculators*

15:45–16:10 Coffee Break

16:10–16:45 Bruno Durand (Ecole Normale Supérieure de Lyon), *Growing Patterns in 1D Cellular Automata*

16:45–17:20 Thomas Worsch (Universität Karlsruhe), *On parallel Turing machines with multi-head control units*

18:00– Dinner

Tuesday, 26th March 1996

07:30–09:00 Breakfast

09:00–09:35 Hiroshi Umeo (Osaka Electro-Communication University), *A Note on Firing Squad Synchronization Algorithms*

09:35–10:10 Maurice Margenstern (Université de Metz), *Towards a new approach to universality about Turing machines*

10:10–10:45 Thomas Buchholz (Universität Gießen), *Time computability of functions in (bounded) one-way cellular automata*

10:45–11:15 Coffee Break

11:15–11:50 Ivan Korec (Slovak Academy of Sciences), *Algebraical constructions of reversible generalized Pascal triangles and one-dimensional cellular automata*

11:50–12:25 Guentcho Skordev (Universität Bremen), *Linear Cellular Automata over Fields and Factorization of Polynomials*

12:30–14:00 Lunch

14:00–14:35 Giancarlo Mauri (Universitá degli Studi di Milano), *Transformations on the 1D Cellular Automata Rule Space*

14:35–15:10 Ken Miura (Kobe University), *The Application of C.A. to Economics: Simulation of Economies with Snobs and Conformists*

15:10–15:45 Jérôme Olivier Durand-Lose (Université Bordeaux I), *Sand Dripping in Linear Space*

15:45–16:10 Coffee Break

16:10–16:35 Frédéric Geurts (Université catholique de Louvain), *CA-like error propagation in fuzzy cellular automata*

16:35–17:45 Mario Markus (Max-Planck-Institut für molekulare Physiologie), *a) Waves in the BZ-reaction: experiments and CA simulations*
b) CA simulations and observations of biological and geological processes. Is there evidence for class 4 in nature?

18:00–19:30 Dinner

Wednesday, 27th March 1996

07:30–09:00 Breakfast

09:00–09:35 Max Garzon (The University of Memphis), *Bilinear Cellular Automata*

09:35–10:10 Danuta Makowiec (Gdansk University), *Chaos in cellular automata with majority rule*

10:10–10:45 Bruno Martin (Université de Nice Sophia Antipolis), *Spatial Machines and Cellular Automata*

10:45–11:15 Coffee Break

11:15–11:50 Jörg Richard Weimar (Technische Universität Braunschweig), *Cellular Automata for Reaction-Diffusion Systems*

11:50–12:25 Tim Boykett (Universität Linz), *Algebraic Aspects of One Dimensional Reversible Cellular Automata*

12:30–14:00 Lunch

14:00–14:35 Peter Hartmann (Technische Hochschule Darmstadt), *Cristal meshes and symmetry relationships: An approach based on cellular hypergraphs*

14:35–15:10 Jean-Paul Allouche (Université Paris Sud), *Cellular automata, finite automata and the Schur property for polynomials*

15:10–15:30 Coffee Break

15:30–16:05 Antje Petersen (Universität Bremen), *Automaticity of rational functions*

16:05–16:40 Christian Hochberger (Technische Hochschule Darmstadt), *CDL – A Language for Cellular Processing*

16:40–16:45 Closing

18:00– Dinner

Cellular automata, finite automata and the Schur property for polynomials

J.-P. Allouche and G. Skordev

Although they have similar names, finite automata and cellular automata are quite different. It is nevertheless interesting to ask whether patterns generated by the time evolution of a one-dimensional cellular automaton can be also generated by two-dimensional finite automata. This question has been addressed in the case of (one-dimensional) linear cellular automata by Korec [4] and by F. von Haeseler, H.-O. Peitgen, A. Petersen and the authors [1, 2]. One useful tool is the "m-Fermat property": a polynomial on a ring R is called m-Fermat if

$$P^m(X) = (P(X))^m.$$

In particular if $R = \mathbb{Z}/p\mathbb{Z}$ with p prime, every polynomial is p-Fermat and, as a consequence,

$$\forall t \geq 0,\ \forall r \in [0, p-1],\ P^{pt+r}(X) = P^t(X^p)P^r(X).$$

The generalization we study here consists in replacing the sequence $(P^t(X))_{t\geq 0}$ which represents the time evolution of a one-dimensional linear cellular automaton by a sequence of polynomials $(Q_t(X))_{t\geq 0}$ verifying what we call "the p-Schur property":

$$\forall t \geq 0,\ \forall r \in [0, p-1],\ Q_{pt+r} = Q_t(X^p)Q_r(X).$$

If $Q_t(X) = \sum a_{t,n}X^n$ and $\deg Q_t \leq t$, this is equivalent to saying that:

$$(*)\ \ \forall t, n \geq 0,\ \forall i, j \in [0, p-1],\ a_{pt+i,pn+j} = a_{t,n}a_{i,j}.$$

Of course the sequence of polynomials defined by $Q_t(X) = (1 + X)^t$ satisfies the p-Schur property on the ring $\mathbb{Z}/p\mathbb{Z}$, the corresponding sequence $(a(t, n))_{t,n}$ is nothing but the sequence $(\binom{t}{n})_{t,n}$ and the property $(*)$ is the Lucas theorem. Analogously the sequence of polynomials $Q_t(X) = B^t(X)$

where B is a fixed polynomial has the p-Schur property on $\mathbb{Z}/p\mathbb{Z}$. We can hence generalize (one-dimensional) linear cellular automata by taking a sequence of polynomials having the p-Schur propery instead of the sequence $B^t(X)$. Another (less evident) example of a p-Schur sequence of polynomials is given by the Legendre polynomials: *the Legendre polynomials have the p-Schur property for any odd prime p*, this has been proved by Schur (according to [3, 6]). We also note that sequences $(a(t,n))_{t,n}$ having the property $(*)$ have been studied by McIntosh [5].

We have obtained results on the Legendre, Hermite and Tshebishev polynomials using *inter alia* results of Carlitz. For example, for the Legendre polynomials, using the p-Schur property and the relation (see [7] for example)

$$\sum_{t\geq 0} P_t(X)Y^t = (1-2XY+Y^2)^{-\frac{1}{2}}$$

we can prove the following. *Define the sequence $(a(t,n))_{t,n}$ by $P_t(X) = \sum a_{t,n}X^n$ where P_t is the t-th Legendre polynomial. Then the sequence $(a(t,n))_{t,n} \bmod \mathrm{d}$ is p-automatic for $d = p^s$ where p is an odd prime and not k-automatic if d has two different odd prime divisors.*

References

[1] J.-P. Allouche, F. v. Haeseler, H.-O. Peitgen, G. Skordev, *Linear cellular automata, finite automata and Pascal's triangle*, Discrete Appl. Math. (to appear).

[2] J.-P. Allouche, F. v. Haeseler, H.-O. Peitgen, A. Petersen, G. Skordev, *Automaticity of double sequences generated by one-dimensional linear cellular automata*, Inst. Dyn. Syst., Univ. Bremen, Rep. 346 (1995).

[3] L. Carlitz, *Congruence properties of polynomials of Hermite, Laguerre and Legendre*, Math. Zeitschr., **59** (1954), 474–483.

[4] I. Korec, *Pascal triangles modulo n and modular trellises*, Computers and Artificial Intelligence **9** (1990), 105–113.

[5] R. J. McIntosh, *A generalization of a congruential property of Lucas*, Amer. Math. Monthly, **99** (1992), 231–238.

[6] J. Wahab, *New cases of irreducibility for Legendre polynomials*, Duke Math. J., **19** (1952), 165–176.

[7] E. T. Whittaker, G. N. Watson, *A course of modern analysis*, Cambridge Univ. Press, 1965, p. 302.

Jean-Paul Allouche
CNRS, LRI, Bâtiment 490
F-91405 Orsay Cedex
France
allouche@lri.lri.fr

Gencho Skordev
Center for Complex Systems and Visualization
University of Bremen
D-28334 Bremen
Germany
skordev@mathematik.uni-bremen.de

Shift-add correlation patterns of linear cellular automata are pseudo cellular automata

André Barbé

Katholieke Universiteit Leuven
Dept. of Electrical Engineering
Kard. Mercierlaan, 94
B-3001 Leuven
Belgium
Email:Andre.Barbe@esat.kuleuven.ac.be

We consider linear cellular automata (CA) defined over a finite commutative ring $\mathcal{R}$ with multiplicative unity. *Shift-add correlation patterns* were introduced in [1],[2] as a means to detect hidden structures in coarse-graining invariant orbits of cellular automata with a linear rule in $\mathbb{Z}_p$.

Dual arrays of shift-add correlation patterns

Let $\bar{a}(r,c) \stackrel{\mathsf{def}}{=} (a(r,c))_{(r,c)\in\mathbb{N}\times\mathbb{Z}}$ be a two-dimensional CA-orbit obtained from the one-dimensional initial configuration $(\bar{a}(0,c))_{c\in\mathbb{Z}}$ through the linear recursion

$$a(r,c) = \sum_{j\in R_1} \rho(1,j)a(r-1,c-j)\,,$$

with R_1 a finite subset of $\mathbb{Z}$, and fixed coefficients $\rho(1,j) \in \mathcal{R}^*$ for all $j \in R_1$ ($\mathcal{R}^*$ is the set of nonzero elements in $\mathcal{R}$). Let (v,h) be a fixed element of $\mathbb{N}\times\mathbb{Z}$, and let α and β be fixed parameters in $\mathcal{R}^*$, then the (v,h)-*shift-add correlation pattern* $\bar{s}_{v,h}(r,c)$ is defined as the two-dimensional sequence $(s_{v,h}(r,c))_{(r,c)\in\mathbb{N}\times\mathbb{Z}}$, with

$$s_{v,h}(r,c) \stackrel{\mathsf{def}}{=} \alpha a(r,c) + \beta a(r+v,c+h).$$

(all operations are in $\mathcal{R}$ of course.)
The *array* $\bar{\mathcal{S}}(v,h)$ of shift-add (correlation) patterns is now defined as the two-dimensional sequence of patterns $(\bar{s}_{v,h}(r,c))_{(v,h)\in\mathbb{N}\times\mathbb{Z}}$.

In a similar way, with (r,c) a fixed element in $\mathbb{N}\times\mathbb{Z}$, we can consider the *dual* (r,c)-shift-add pattern as the two-dimensional sequence

$$\bar{s}^*_{r,c}(v,h) \stackrel{\mathsf{def}}{=} (s^*_{r,c}(v,h) \stackrel{\mathsf{def}}{=} \alpha a(r,c) + \beta a(r+v,c+h))_{(v,h)\in\mathbb{N}\times\mathbb{Z}}.$$

Also, the *array* of *dual* shift-add patterns $\bar{\mathcal{S}}^*(r,c)$ is defined as the two-dimensional *sequence of patterns* $\bar{s}^*_{r,c}(v,h))_{(r,c)\in\mathbb{N}\times\mathbb{Z}}$.

When represented graphically as arrays of patterns, both $\bar{\mathcal{S}}(v,h)$ and $\bar{\mathcal{S}}^*(r,c)$ can be seen as a dual pair of natural visualisations of the four-dimensional sequence

$$(s(v,h,r,c)\stackrel{\mathsf{def}}{=}\alpha a(r,c)+\beta a(r+v,c+h))_{(v,h,r,c)\in\mathbb{N}\times\mathbb{Z}\times\mathbb{N}\times\mathbb{Z}}.$$

Regular CA and associated pseudo-CA

A (one-dimensional) CA with the rule (in polynomial representation) $R(x)=\sum_{j\in R_1}\rho(1,j)x^j$, evolves according to the recursion

$$a(r,c)=\sum_{j\in R_1}\rho(1,j)a(r-1,c-j)\,,$$

starting with some initial state configuration $\bar{a}(0,c)$.

The corresponding *pseudo*-CA is defined as a CA-like structure that evolves according to the modified recursion

$$q(r,c)=\sum_{j\in R_1}\rho(1,j)q(r-1,c-j)+\gamma\theta(1)\,,$$

starting with the initial configuration $\bar{q}(0,c)=\bar{a}(0,c)$, and where γ is a fixed element of $\mathcal{R}^*$ and

$$\theta(1)\stackrel{\mathsf{def}}{=}1-\sum_{j\in R_1}\rho(1,j).$$

It is an interesting property to observe that, whereas a regular CA also satisfies the relationship

$$a(r,c)=\sum_{j\in R_v}\rho(v,j)a(r-v,c-j)$$

with $\rho(v,j)=\sum_{i\in R_1}\rho(1,j)\rho(v-1,c-j)$ and with $R_v\stackrel{\mathsf{def}}{=}\{j|\rho(v,j)\neq 0\}$ (for all $v\in\mathbb{N}$), the pseudo-CA satifies a corresponding relationship

$$q(r,c)=\sum_{j\in R_v}\rho(v,j)a(r-v,c-j)+\gamma\theta(v)\quad ,$$

where $\theta(v)\stackrel{\mathsf{def}}{=}1-\sum_{j\in R_v}\rho(v,j)$.

The relationship between shift-add correlation patterns and pseudo-CA

Consider a linear CA over $\mathcal{R}$, evolving under a given rule $R(x)$. For the associated shift-add correlations patterns and their duals, the following facts hold:

- $\bar{s}_{v,h}(r,c)$-patterns are regular CA-patterns evolving under the rule $R(x)$.
- the $\bar{\mathcal{S}}(v,h)$-*array* of $\bar{s}_{v,h}(r,c)$-patterns has the structure of a pseudo-CA orbit evolving under the rule $R(x)$.
- the dual $\bar{s}^*_{r,c}(v,h)$-patterns are pseudo-CA patterns evolving under the rule $R(x)$.
- the dual $\bar{\mathcal{S}}^*(r,c)$-*array* of $\bar{s}^*_{r,c}(v,h)$- patterns has the structure of a regular CA orbit evolving under the rule $R(x)$.

References

1 Barbé, A.,von Haeseler, F., Peitgen, H.-O., Skordev, G. [1995] "Coarse-graining invariant patterns of one-dimensional two-state linear cellular automata," *Int. J. Bifurcation and Chaos*, **5**(6), 1611-1631.

2 Barbé, A. [1996] "Coarse-graining invariant orbits of one-dimensional $\mathbb{Z}_p$ -linear cellular automata," to appear in *Int. J. Bifurcation and Chaos.*

3 Barbé, A. [1996] "Shift-add correlation patterns of orbits of one-dimensional linear cellular automata," to appear in *Int. J. Bifurcation and Chaos.*

Algebraic Aspects of One Dimensional Reversible Cellular Automata

Tim Boykett, Universität Linz.
tim@bruckner.stoch.uni-linz.ac.at

Pedersen's breakthrough work in [Ped92] has opened 1D CA to algebraic analysis. His work involves looking at CA as groupoids, examples of which permeate abstract algebra. A further development of his work gives a simple abstract algebraic characterisation of 1D reversible CA as $(2,2)$–algebra $(S, \bullet, \cdot)$ satisfying the axioms

$$(a \bullet b) \circ (b \bullet c) = b \quad (1)$$
$$(a \circ b) \bullet (b \circ c) = b \quad (2)$$

This structure is known as a *semicentral bigroupoid* as it generalises the *central groupoids* of Trevor Evans and Donald Knuth.

Elementary results, published in [Boy95] show an equivalent combinatoric formulation. These *rectangular structures* are easily seen to be equivalent to the tables of David Hillman in [Hil91]. They have a number of other interesting properties, however, relating to some questions in matrix theory and digraph theory.

Recent work has resulted in efficient combinatoric algorithms to exhaustively enumerate examples, reaching currently up to order 10. In this formulation, algebraic concepts of particle–like behaviour can be formulated, resulting in the discovery of significant amounts of such behaviour in the examples constructed.

A survey of these results will be presented, along with various interesting open questions.

References

[Boy95] Tim Boykett. Combinatorial construction of one dimensional reversible cellular automata. In G. Pilz, editor, *Contributions to General Algebra 9*, pages 81–90. Hölder–Pichler–Tempsky, Teubner, 1995.

[Hil91] David Hillman. The structure of reversible one–dimensional cellular automata. *Physica A*, 52:277–292, 1991.

[Ped92] John Pederson. Cellular automata as algebraic systems. *Complex Systems*, 6:237–250, 1992.

Time computability of functions in (bounded) one-way cellular automata

Thomas Buchholz and Martin Kutrib

AG Informatik
Universität Gießen
Arndtstr. 2, D–35392 Giessen

{buchholz,kutrib}@informatik.uni-giessen.de

The classical grammar-based language theory classifies languages by their structural complexity. Another classification (computational complexity) is based on the amount of time, space, or other resources needed to recognize a language on some device, such as Turing machines or cellular automata. That is, the resources available are bounded by a function f. In order to obtain a infinite dense hierarchy one way (by using diagonalization) is to ensure that only a slight increase in the growth rate of f yields to a new complexity class. Therefore some "well-behaved" complexity functions are used, where the notion of well-behaviour is concretized in terms of constructibility of functions with respect to the investigated resources and devices.

We transfer the notion of time-constructibility to (bounded) one-way cellular automata (where we will call it time-computability, then). Surely, we cannot hope to obtain an infinite hierarchy of complexity classes for OCAs using such functions (diagonalization does not work). However we are interested in the spectrum of OCA-time-computable functions as is for understanding the capabilities of one-way cellular automata. In the talk we will give some examples of OCA-time-computable functions and show the existence of several gaps in the spectrum. Further we will charakterize OCA-time-computable functions by a special family of languages each of which is acceptable by an OCA in real-time.

References

[1] Thomas Buchholz and Martin Kutrib. On time computability of functions in one-way cellular automata. Report 9502, AG Informatik, University Giessen, Giessen, 1995.

[2] Jacques Mazoyer and Véronique Terrier. Signals in one dimensional cellular automata. Research Report RR 94-50, Ecole Normale Supérieure de Lyon, Lyon, 1994.

[3] Roland Vollmar. *Algorithmen in Zellularautomaten.* Teubner, Stuttgart, 1979.

Growing Patterns in 1D Cellular Automata

Bruno Durand

We study limit evolutions of cellular automata (CA) both theoretically and experimentally. We show that either all orbits enter the limit set after less than a finitely bounded number of states, or almost all orbits never enter the limit set. We link this result with a classification of CAs according to their limit behavior due to Čulik et al: in the first case, the considered CA belongs to class 1 while in the second case, it belongs to class 2.

By experiments, we try to measure the convergence speed of orbits to their accumulation points. We compute the maximum number of nested growing segments in a space-time diagram representing a finite portion of an orbit. We observe that, in the average case, this criterion depends only on the CA and not on the configuration. We also observe two kinds of CA wrt our criterion which correspond to the intuitive notions of chaos and regularity. We do further experiments to explain the fact that the proportion of chaotic diagrams grows with the number of states of the considered CA.

Sand Dripping in Linear Space

Jérôme Olivier Durand-Lose*†

LaBRI, URA CNRS 1 304,
Université Bordeaux I,
351, cours de la Libération,
33 405 Talence Cedex,
France.

e-mail: `jdurand@labri.u-bordeaux.fr`

Abstract

We study the evolution of an empty linear sand pile system which receives a sand grain in its first pile each time. We encode it with another cellular automata, the Chip Firing Game. Geometrical observations lead to a signal encoding of configurations. The interactions of signals are studied. Configurations are divided in two parts of slopes 1 and 2. We make asymptotic approximations of various parameters defining the configurations. Especially, we find that the configurations are expanding regularly in terms of the square root of the number of fallen grains and the ratio of the width of the two parts of the configurations is $\sqrt{2}$.

Sand Piles Model (SPM) and Chip Firing Games (CFG) are very simple systems based on local interactions which conserve the total number of grains. In SPM, a grain moves to a neighbor site if the number of grains in the two piles differs by more than a given threshold, whereas in CFG a site gives a chip to each of its neighbors if its number of chips is above the number of neighbors. In the linear case, both games are 1-D cellular automata and are equivalent via a simple encoding [7].

These models were first introduced in physics to study the so called '$1/f$ phenomenon' and 'Self-Organized Criticality' in avalanches, earthquakes, plasmas, ... [12].

SPM and CFG, like Petri Nets [10], are used to model parallel computing, in particular the flows of informations in a system. Grains model data or tasks, and piles, a processor network [11]. The goal is to find a simple, fast and costless local reorganization which ensures that all processors will have almost the same amount of work.

*This work was partially supported by ECOS and the French Cooperation in Chile.

†This research was done while the author was in the Departamento de Ingeniería Matemática, Facultad de Ciencias Físicas y Matemáticas, Universidad de Chile, Santiago, Chile.

We limit the problem to linear space. Goles and Kiwi have widely studied the case of linear SPM and the related CFG [6, 7]. They have detailed the dynamic as well as demonstrated the unicity of the final configuration whatever the order of the iterations is for various sequential cases.

Anderson *et al.* studied a similar problem in [1]. They also found signals in CFG, but their model allows that more than one grain travel between two piles at the same time.

We study the parallel updating of a system which is empty at start and that, while relaxing, receives grains in its first site. It can be understood as sand dripping in a thin but long hourglass.

Triangle straps appear on 3-D representations of the evolution of the system. Using a height difference notation, we switch to the CFG model. All configuration are then of the form $2^* \, (13)^* \, (02)^* \, 1^*$. They are made of two parts of distinct slopes (2 for $2^* \, (13)^*$ then 1 for $(02)^* \, 1^*$). Each one is itself composed of two sections of different motifs.

The configurations are parted in four sections amounting to the four motifs, delimited by signal-like frontiers. The first signals, between 2^* and $(13)^*$, and last one, between $(02)^*$ and 1^*, are always moving between the borders of the configurations and the second one, between $(13)^*$ and $(02)^*$. The second signal is fixed, it is only moved by the two other ones. It somehow counts the comming and going of the two others.

We give asymptotic approximations of the different parameters and geometrical shape. As expected, since the grains must fill a surface, we found that the height and width of the system are growing in $o(\sqrt{n})$, where n is the number of fallen grains. Moreover the relative length of the two parts converges towards $\sqrt{2}$.

References

[1] R. Anderson, L. Lovász, P. Shor, J. Spencer, E. Tardos, and S. Winograd. Disks, Balls and Walls: Analysis of a Combinatorial Game. *American Mathematics Monthly*, (96):481–493, 1989.

[2] J. Bitar and E. Goles. Parallel Chip Firing Games on Graphs. *Theoretical Computer Science*, 92:291–300, 1992.

[3] A. Bjöner and L. Lovász. Chip-Firing Games on Directed Graphs. *Journal of Algebraic Combinatorics*, 1:305–328, 1992.

[4] A. Bjöner, L. Lovász, and P. W. Shor. Chip-Firing Games on Graphs. *Europ. J. Combinatorics*, 12:283–291, 1991.

[5] J. O. Durand-Lose. *Automates Cellulaires, Automates à Partitions et Tas de Sable*. PhD thesis, LaBRI, 1996.

[6] E. Goles and M. Kiwi. Dynamics of sand-piles games on graphs. In *LATIN'92. Lect. Notes in Comp. Sci. 583*, pages 219–230. Springer-Verlag, 1992.

[7] E. Goles and M. Kiwi. Games On Line Graphs and Sand Piles. *Theoretical Computer Science*, 115:321–349, 1993.

[8] P. Grassberger and S. Manna. Some More Sandpiles. 1990.

[9] E. Prisner. Parallel chip firing on digraphs. *Complex Systems*, 1993.

[10] C. Reutenauer. *Aspects Mathématiques des Réseaux de Pétri*. Masson, 1989.

[11] R. Subramanian and I. Scherson. An Analisys of Diffusive Load-Balancing. In *ACM Symposium on Parallel Algorithms and Architecture*, pages 220–225, 1994.

[12] Y.-C. Zhang. Scaling Theory of Self-Orgnized Criticality. *Physical Review Letters*, 63(5):470–473, 1989.

Cellular automata and attributed parallel array grammars

Rudolf FREUND
Technische Universität Wien, Institut für Computersprachen
Resselgasse 3, A-1040 Wien, Austria
email: freund@csdec1.tuwien.ac.at

Fritz TAFILL
Technische Universität Wien, Institut für Elektrotechnik und Elektronik,
Arbeitsgruppe für Neuronale Elekronik
Gußhausstraße 25 – 29, A-1040 Wien, Austria
email: fritz@synapse.iaee.tuwien.ac.at

ABSTRACT. The concept of n-dimensional parallel array grammars (e.g. see [1]) is a useful means for the formal syntactic description of n-dimensional cellular automata ([7]); for the two-dimensional case, a survey of possible applications of array automata like picture recognition is given in [5]. As n-dimensional parallel array grammars are a more general model than n-dimensional cellular automata, they also allow for simpler solutions of complex problems. Attributed parallel array grammars have been shown to be a useful tool for the formal description of the static and dynamic characteristics of neural networks; because of the underlying grid structure, Kohonen's model of self-organizing feature maps (e.g. see [6]) is especially well suited for being represented by n-dimensional parallel array grammars (e.g. see [3], [4]). Using the concept of different levels of scaling like in [2], even hierarchical networks can be modelled by attributed parallel array grammars.

References

[1] R. Freund, "Aspects of n-dimensional Lindenmayer systems", *Developments in Language Theory,* Turku, Finland, 1993.

[2] R. Freund, "Multi-level eco-array grammars", *Artificial Life – Grammatical Models* (Gh. Paun, ed., Black Sea University Press, Bucureşti, 1995), pp. 166 – 174.

[3] R. Freund, F. Tafill, "Modelling Kohonen networks by attributed parallel array systems", *SPIE'93,* Innsbruck, Austria, 1993.

[4] R. Freund, F. Tafill, "Formal representation of neural networks", *ICANN'94,* Sorrento, Italy, 1994.

[5] K. Inoue, I. Takanami, "A survey of two-dimensional automata theory", *Information Sciences* **55** (1991), pp. 99 – 121.

[6] H. Ritter, K. Schulten, T. Martinetz, *Neuronale Netze* (Addison-Wesley, Deutschland, 1990).

[7] S. Wolfram (ed.), *Theory and applications of cellular automata* (World Scientific Publ., Singapore, 1986).

Bilinear Cellular Automata

Ron Bartlett
Mathematical Sciences
Denison University
Granville, Ohio 43023 USA
bartlett@sunshine.mathsci.denison.edu

Max Garzon
Mathematical Sciences
The University of Memphis
Memphis, Tennessee 38152 USA
garzonm@hermes.msci.umem.edu

Abstract

Bilinear cellular automata (CA) are those whose next state may be expressed as a bilinear form (an inner product) of the neighboring states. In this paper we show that, unlike linear CA, the bilinear CA over $\mathbf{Z}_p^p$ are π-universal, i.e. capable of simulating any CA of the same dimension, and hence also capable of simulating any (universal) Turing machine. Evidence is given that the bilinear CA over $\mathbf{Z}_m$, the integers modulo m, may be universal as well. (Although, like Conway's Game of Life, this appears to be difficult to establish, even for a prime number of states). Assuming that the bilinear CAs over $\mathbf{Z}_m$ are T-universal, we also give a fairly complete Wolfram type classification of the bilinear CA over $\mathbf{Z}_m$ according to properties of the coefficient matrix. As further evidence, we show bilinear automata are good models of discrete gravitational fields (monomial types), solitons, bounded growth, edge detection, and other applications.

CA-like error propagation in fuzzy cellular automata

Frédéric Geurts

We describe and analyze the surprising evolution of an overflow error E which has spontaneously occurred during the simulation of a class of continuous complex systems called "cellular coupled map lattices" ($\kappa - CML$) or fuzzy CA. In fact, this error evolves as a boolean cellular automaton (CA), regardless of and destroying the underlying system evolution.

The error and its evolution are fully reproducible, and its behavior can be fully characterized. An interesting and important aspect of this phenomenon is that it shows a quasi-natural occurrence of an artificial world.

(Non)-Automaticity of Orbits of Linear Cellular Automata

Fritz v. Haeseler[1]

Center for Complex Systems and Visualization

University of Bremen

D-28334 Bremen

This is an abridged version of a joint paper, see [3], on the automaticty of orbits of linear cellular automata. We present an extension of results in [2, 6].

1 Preliminaries

Let us quickly recall what k-automatic sequences and double sequences are. For more details on automatic sequences the reader is referred to e.g. [1].

Definition 1 *Let $p \geq 2$ be an integer. A sequence $\{x(n)\}_{n\in\mathbb{N}}$ with values in a finite set is called p-automatic if its p-kernel, i.e., the set of subsequences*

$$\{n \mapsto x(p^i n + j) \mid i \geq 0, 0 \leq j \leq p^i - 1\},$$

is finite.
The double sequence $\{u(n,k)\}_{n,k\in\mathbb{N}}$ with values in a finite set is called p-automatic if the p-kernel, i.e., the set

$$\{n \mapsto u(p^i n + j, p^i k + l) \mid i \geq 0, 0 \leq j, l \leq p^i - 1\},$$

is finite.

Most important for our considerations is the following version of Cobham's Theorem, see [4].

Theorem 2 (Cobham)

- *Let p_1, p_2 be different prime numbers. A sequence $\{x(n)\}$ is p_1-automatic and p_2-automatic if and only if $\{x(n)\}$ is eventually periodic.*

- *Let p be a prime number. If $\{x(n)\}$ is not eventually periodic and p-automatic, then $\{x(n)\}$ is k-automatic if and only if $k = p^a$ for some $a \in \mathbb{N}$.*

Let m be a natural number. We denote the set of residue classes of $\mathbb{Z}$ modulo m by $\mathbb{Z}_m$. Furthermore, we consider the set $\mathbb{Z}_m$ as a ring with ring structure induced

[1]supported by the Deutsche Forschungsgemeinschaft Forschungsgruppe "Dynamische Systeme"

by the ring structure of $\mathbb{Z}$. Any polynomial $R(x) \in \mathbb{Z}_m[x]$ induces a linear cellular automaton on the configuration space

$$\Sigma(\mathbb{Z}_m) = \left\{ f(x) = \sum_{j \in \mathbb{Z}} a_j x^j \mid a_j \in \mathbb{Z}_m \right\}$$

by $f(x) \mapsto f(x)R(x)$. Given an initial condition $a(x) \in \mathbb{Z}_m[x]$, we are concerned with the orbit description of $a(x)$ under the cellular automaton induced by a polynomial $R(x) \in \mathbb{Z}_m[x]$. We write the orbit as a formal power series in two variables, i.e.,

$$F(X,Y) = \sum_{t=0}^{\infty} a(X)R(X)^t Y^t = \sum_{n,t=0}^{\infty} f(n,t) X^n Y^t.$$

We thus obtain a double sequence $(f(n,t))$, and we are interested under what conditions is the double sequence automatic. Some first results in that regard are:

Theorem 3 ([2, 6]) *Let m be a natural number, $m \geq 2$, and let $R(x) = 1 + x \in \mathbb{Z}_m[x]$ and $a(x) = 1$, then $(f(n,t))$ is p-automatic (p a prime number) if and only if $m = p^\alpha$ for some $\alpha \in \mathbb{N}$.*

More generally, we have

Theorem 4 ([2]) *Let p be a prime number and let $R(x), a(x) \in \mathbb{Z}_{p^\beta}[x], \beta \in \mathbb{N}$. Then $F(X,Y)$ is p-automatic.*

The proof of the above theorem relies on the little Fermat Theorem. It states that $R(x)^p = R(x^p)$ for all $R(x) \in \mathbb{Z}_p[x]$ and p a prime number. This observation yields the following

Theorem 5 ([3]) *Let $R(x) \in \mathbb{Z}_m[x]$ such that $R(x)^k = R(x^k)$ for a $k \geq 2$, then $(f(n,t))$ is k-automatic.*

2 The main result

In this section, we state the main result together with a sketch of the proof. For details we refer the reader to [3].

Theorem 6 ([3]) *Let $r(x), a(x) \in \mathbb{Z}[x]$ and let $m \geq 2$ be a natural number. Let*

$$F(X,Y) = \sum_{n,t=0}^{\infty} f(n,t) X^n Y^t = \sum_{t=0}^{\infty} a(x) r(x)^t Y^t$$

denote the orbit of $a(x)$ under the cellular automaton $r(x)$ in $\mathbb{Z}$. Furthermore, let χ be the number of different prime divisors p of m for which $r(x)$ mod p is not a monomial. Then the following holds

- *If $\chi \geq 2$, there is no value of $k \geq 2$ for which $(f(n,t) \text{ mod } m)$ is k-automatic.*

- *If $\chi = 1$, then $(f(n,t) \text{ mod } m)$ is k-automatic if and only if $k = p^{\alpha}$, where $\alpha \geq 2$ and p is the prime divisor for which $r(x)$ mod m is not a monomial.*

- *If $\chi = 0$, then $(f(n,t) \text{ mod } m)$ is k-automatic for all $k \geq 2$.*

The main ingredient of the proof of the above theorem is the following lemma

Lemma 7 (Slice Lemma) *Let p be a prime number and let $r(x)$ mod p be not a monomial, let $a(x) \in \mathbb{Z}_p[x]$. Then there exist $\alpha, \beta \in \mathbb{N}$ such that the sequence $u_p(n) = \{f(\alpha n, \beta n) \text{ mod } p\}_{n \in \mathbb{N}}$ is not eventually periodic.*

The proof of the lemma makes use of the geometrical description, cf.[5], of the orbit. We are now able to sketch the proof of the main theorem.

- Let p, q be different prime numbers such that $r(x)$ mod $p, r(x)$ mod q are not monomials, respectively. Now suppose $(f(n,m) \bmod m)$ to be k-automatic, then, by the Slice Lemma, there exists a sequence $u_p(n)$ which is k-automatic and p-automatic. By Cobham's Theorem, it follows that $k = p^a$. On the other hand, there exists a sequence $u_q(n)$ which is k-automatic and q-automatic, therefore $k = q^b$; which is a contradiction.

- Let p be the only prime divisor of m such that $r(x)$ mod p is not a monomial, let $m = p^a q_1^{a_1} \cdots q_w^{a_w}$ be the prime factorisation of m. Then there exist $\alpha_j, j = 1, \ldots, w$, such that $g_j(x) = r(x)^{\alpha_j}$ has the property $g_j(x)^k = g_j(x^k) \bmod q_i^{a_i}$. Moreover, there exists an α such that $g(x) = r(x)^{\alpha}$ has the property $g(x)^p = g(x^p) \bmod p^a$. Then $G(x) = r(x)^{\alpha\alpha_1 \cdots \alpha_w}$ has the property $G(x)^p = G(x^p) \bmod p$. Hence $(f(n,t) \bmod m)$ is p-automatic. By the Slice Lemma, we obtain that $(f(n,t) \bmod m)$ is k-automatic if an only if k is of the form $k = p^{\alpha}$.

- A similar reasoning as for the second case shows that there exists a natural number M such that $G(x) = r(x)^M$ has the property $G(x)^k = G(x^k) \bmod m$. Therefore the sequence $(f(n,t) \bmod m)$ is k-automatic for all $k \geq 2$

References

[1] J.-P. Allouche, Automates finis en théorie des nombres, Expo. Math. 5 (1986), 239-266

[2] J.-P. Allouche, F. v.Haeseler, H.-O. Peitgen, G. Skordev, Linear cellular automata, finite automata, and Pascal's triangle, Discrete Appl. Math. (to appear)

[3] J.-P. Allouche, F. v.Haeseler, H.-O. Peitgen, A. Petersen, G. Skordev, Automaticity of double sequences generated by one-dimensional linear cellular automata, submitted

[4] A. Cobham, On the base-dependence of sets of numbers recognizable by finite automata, Math. Systems Theory 3 (1969), 186-192

[5] F. v.Haeseler, H.-O. Peitgen, G. Skordev, On the fractal structure of rescaled evolution sets of cellular automat and attractors of dynamical systems, Inst. Dyn. Syst., University of Bremen, Report 278, 1992

[6] I. Korec, Pascal's triangles modulo n and modular trellises, Computers and Artificial Intelligence 9 (1990), 105-113

Crystal meshes and symmetry relationships: An approach based on cellular hypergraphs

Peter Hartmann
Fachgebiet Mikroprogrammierung
Fachbereich Informatik
TH Darmstadt
Alexanderstraße 10, D-64283 Darmstadt, Germany
Tel.: [+49] 6151 / 16-3474 Fax: [+49] 6151 / 16-5410
E.-Mail: hartmann@isa.informatik.th-darmstadt.de

Abstract

Like three-dimensional cellular automata, crystals are based on regular grids, too. Consequently, if we discuss questions about the symmetry of cellular automata, we can profit from results obtained in the field of crystallography. In this contribution it will be described how the elementary cells of a crystal can be formulated using three-dimensional cellular hypergraphs. These graphs allow us to model the geometrical properties of the elementary cells (e.g. distances and angles) on a high level of abstraction. Furthermore, the symmetry elements (e.g. mirror planes, symmetry axes, . . .) can be expressed in terms of constraints on some cellular hypergraph. Finally, if there is given the graph of the elementary cell and a set of symmetry constraints, it is possible to construct the complete crystal mesh from this data. This allows us to build cellular automata models of crystals with arbitrary symmetry properties.

Keywords: Structure of cellular automata, cellular automata as models for real phenomena

Extended abstract

The results presented in this contribution were initiated by a discussion which I had several years ago with a colleague of mine: He was working at the classification of cellular automata models, and consequently it was necessary to clarify which repetitive regular (e.g. symmetric) structures can be built in three-dimensional space.

It is clear that we can profit from the results obtained in *crystallography* when discussing this question, because this discipline provides us with a catalogue of all conceivable symmetric repetitive meshes (see e.g. the textbooks [1, 5]). On the other hand, crystallography can profit from cellular automata models, too, when physical processes which take place in a crystal need to be simulated. But this requires that we are equipped with a description language for cellular systems (e.g. *CDL*, for an introduction see [4]) which allows us to deal with any imaginable symmetry relationships (for actual research from the point of view of crystallography, see e.g. [6]).

So I resumed the research mentioned above because of the finding that *cellular hypergraphs* are well suited for describing the elementary cells of a crystal as well as how the elementary cells are composed to form the complete crystal mesh. Initially cellular hypergraphs have been developed with respect to inhomogeneous and dynamically changing structures [2]. But they can be used as well for describing structures having some local inhomogeneity in combination with an overall regular character. And furthermore, the *relative addressing mechanisms* defined for cellular hypergraphs (see [3]) allow us to formulate the constraints on any symmetry elements of a given crystal.

The talk will consist of three major sections: The first section will give an overview about the essentials of crystallography. Based on the different types of *symmetry axes*, the terms *crystal system*, *point group* and *space group* will be illustrated. In the second section the fundamentals about cellular hypergraphs and about relative addressing within these graphs will be recalled. In the final section it will be demonstrated how the elementary cell of a sample crystal can be described using a cellular hypergraph. And it will be shown how the description of the elementary cell can be combined with additional symmetry constraints, such that a complete crystal mesh can be constructed.

Our project is granted by the *Deutsche Forschungsgemeinschaft*.

References

[1] W. Borchardt-Ott. *Kristallographie*. Springer-Verlag, 1993.

[2] P. Hartmann. Parallel Replacement Systems on Geometric Hypergraphs: A Mathematical Tool for Handling Dynamic Geometric Sceneries. *Proceedings of the VIth International Workshop on Parallel Processing by Cellular Automata and Arrays, PARCELLA Workshop Potsdam*, 12:81–90, September 1994.

[3] P. Hartmann. Models of Information Processing in Inhomogeneous Cellular Structures. *First International Workshop on Information Processing in Cells and Tissues, Liverpool*, 1995.

[4] Christian Hochberger, Rolf Hoffmann, and S. Waldschmidt. Compilation of CDL for different Target Architectures. *Proceedings of the third International Conference on Parallel Computing Technologies (PaCT-95), St. Petersburg, Russia*, 1995.

[5] D. McKie and C. McKie. *Essentials of Cristallography*. Blackwell Scientific Publications, 1986.

[6] I. Sens. *Herleitung möglicher Kristallstrukturtypen*. Dissertation am Fachbereich Chemie der Philipps-Universität Marburg, 1993.

Constant speed-up on one dimensional calculators

Olivier Heen
LITP-IBP, Université Paris 7,
2 Place Jussieu, 75251 Paris, FRANCE.

1 Recognizers and calculators

A simple cellular automaton is a finite line of cells with nearest neighborhood and a quiescent state $\sharp$. Language *recognizers* stop when the leftmost cell enters a special state (either accepting or rejecting) which is the result of the computation. Function *calculators* stop when the leftmost cell enters a terminating state, and the result is the whole configuration. In this case, the output can be stable or synchronized.

2 Elementary complexity results

There is a large panel of complexity results for recognizers such as constant or linear speed-up theorems.

Theorem 1 (speed-up for recognizers [IKM85, MR92, Hee95]) *Any recognizer with s states working in time $u \mapsto |u| + t(u)$ can be simulated by a recognizer with $\mathcal{O}(s^{k+1})$ states working in time $u \mapsto |u| + \sup\{t(u) - k, 0\}$ and by a recognizer with $\mathcal{O}(s^k)$ states working in time $u \mapsto |u| + \lceil t(u)/k \rceil$ (where k is a positive integer).*

For calculators, however, there is no general linear speed-up technique (although there might be some for "slow enough" synchronized calculators). Constant speed-up is possible [CC84] but requires an exponential growth of the set of states.

3 New speed-up methods for calculators

We discuss new techniques to achieve the speed-up with a polynomial increasing of the set of states (as it is the case for recognizers).

From a calculator with s states working in time $u \mapsto |u| + 1 + t(u)$ we build a new calculator with $\mathcal{O}(s^{2k+1})$ states working in time $u \mapsto |u| + 1 + \sup\{t(u) - k, 0\}$, by using the so called *redunding* method and a minimal time synchronization from both ends of the line.

Then we present a better method leading to $\mathcal{O}(s^{k+1})$ states, without synchronization. The figure below illustrates this method with the conventions that ct is the state of the c^{th} cell at time t in the initial computation and that $a = 10$.

♯	00	10	20	30	40	50	60	70	♯
♯	• ♯ 01	11	21	31	41	51	61	70 ♯ •	♯
♯	♯ 02	• 02 12	22	32	42	52	62 72 •	72 ♯	♯
♯	♯ ♯	03 13	• 13 23	33	43	53 63 •	63 73	♯ ♯	♯
♯	$	05 04	14 24	• 24 34	44 54 •	54 64	74 75	$	♯
♯	$	06	16 15	+ 25 35	45 55 +	65 66	76	$	♯
♯	$	07	+ 17 26	27 37	47 57	56 67 +	77	$	♯
♯	$	+ 08 17	18 28	38	48	58 68	67 67 +	$	♯
♯	+ ♯ 08	09 19	29	39	49	59	69	78 ♯ +	♯
♯	+ ♯ 0a	1a	2a	3a	4a	5a	6a	7a ♯ +	♯

From an information theoretical point of view, there seems to be no obstacle to constant speed-up with less than a polynomial growth of the set of states. Althougth our method uses product states, they appear very sparsely in the space-time diagram (that is with density $\mathcal{O}(|u|)$).

References

[CC84] C. Choffrut and K. Culik. On real-time cellular automata and treillis automata. *Acta Informatica*, 21:393–407, 1984.

[Hee95] O. Heen. Linear speed-up without synchronization. Rapport LITP 57, Institut Blaise Pascal, 1995.

[IKM85] O. H. Ibarra, S.M. Kim, and S. Moran. Sequential machine caracterization of treillis and cellular automata and application. *SIAM J. Comput.*, 14:426–427, 1985.

[MR92] J. Mazoyer and N. Reimen. A linear speed-up theorem for cellular automata. *Theoretical Computer Science*, 101:59–98, 1992.

CDL – A Language for Cellular Processing

Christian Hochberger (hochberg@informatik.th-darmstadt.de)
Fachgebiet Mikroprogrammierung, Institut für Systemarchitektur,

Technische Hochschule Darmstadt
Alexanderstraße 10, D-64283 Darmstadt, Germany

1 Why Another Language?

Our goal is to describe complex cellular automata in a concise and readable way. The description of complex cellular automata models requires two supporting language elements:

1. The cell state must be a record of arbitrary types like enumerations, integer, subranges of both and maybe even floating popint values. It might also be desirable to have unions, to reduce the memory complexity of the cell state.

2. Testing of complex conditions requires deeply nested *if–then–else* or *case* statements. Support for complex automata must include special language elements that simplify the testing of complex conditions.

The descriptions shall be usable on arbitrary target architectures including special hardware simulators. Classical programming languages and already existing cellular programming languages fail to overcome these constraints, thus we created the language CDL. Some references to cellular programming languages: Cellang[1], CAL[2], Ceprol[3]

2 Features of CDL

Our language has the following features:

- The cell state can be a record or union of arbitrary types. The basic types supported are: enumeration, integer, float (may not be available in all implementations) and the special type `celladdress`.

- Through enumerations the CDL provides symbolic state names. The compiler does not necessarily have to encode the symbolic state names as they are ordered in the enumeration. This gives the compiler the opportunity to optimize the code.

- Special statements (`all, one, num`) are provided, which make the testing of complex conditions easy.

- In CDL all loops are unrollable at compile time. This makes it easy to translate the language into combinational logic.
- CDL contains a very simple yet powerful visualization concept. A colour can be assigned to cells fulfilling certain conditions.

3 Target Architectures for CDL

Three target architectures are currently supported by CDL:

1. A software simulator, which is based on X11 and Unix. Thus it is very portable. The CDL compiler translates the rule description into a C–function which is then compiled by the system compiler. Additionally the compiler generates control information for the visualizer. Details on the transformation process can be found at [4].

   This simulator has several drawbacks, so we are in the process of reimplementing it.

2. The CEPRA8–L machine[5]. This is a specialized hardware for the simulation of cellular automata. It is capable of simulating approximately 5 million cells per second. The memory complexity of a cell is limited to eight bit. The visualization of the entire field is done concurrently with the computation of the next generation. The next state is evaluated through a FPGA. Eight FPGAs work in parallel.

   The CDL compiler generates logic equations from the description. These equations are then processed by a set of commercial tools. The resulting design data can be downloaded into the FPGAs to load a new rule. For further information [6] may be referenced.

3. The CEPRA1–L machine. It uses a FPGA to compute the next state like the CEPRA8–L machine. The cellular field of the CEPRA1–L is stored in the host systems memory. This allows arbitrary sized fields, but implies the usage of the host systems graphic adaptor for the visualization.

4 Future Enhancements

There are several plans to enhance CDL:

- CDL will support multiple rules that are based on the same cell structure. This is especially useful for simulators which try to optimize the computation of the field. More details on this subject will be presented at the workshop by R. Schneider.
- CDL will be enhanced to support more complex visualization concepts, e.g. building patterns from cell clusters.
- CDL will include new elements to describe moving objects. The user will only need to specify the direction for an object and how conflicts shall be resolved, if objects would collide.

References

[1] Eckart, J.D.: A Cellular Automata Simulation System, SIGPLAN Notices **26** (August 1991) 80–85

[2] Palmer, I. J.: Scamper, available by anonymous ftp from ftp.uu.net.

[3] Seutter, F.: CEPROL – Eine zellulare Programmiersprache, Dissertation TU–Braunschweig 1985

[4] Hochberger, C., Hoffmann, R., Waldschmidt, S.: Compilation of CDL for Different Target Architectures, In Parallel Computing Technologies (LNCS 964), Springer Verlag, Editor: Malyshkin, V.

[5] Hoffmann, R., Völkmann, K.-P., Sobolewski, M.: The Cellular Processing Machine CEPRA-8L, in Parcella 94,179-188, Mathematical Research Vol. 81, Akademie Verlag 1994 Editors: Jesshope, Jossifov, Wilhelmi

[6] Waldschmidt, S., Hochberger, C.: FPGA Synthesis for Cellular Processing, In Proceedings of the 1995 IEEE/ACM International Workshop on Logic Synthesis

Architectures for Cellular Processing

Rolf Hoffmann and Klaus-Peter Völkmann
Microprogramming and Computer Architecture
Technical University of Darmstadt
Alexanderstr. 10, D-64283 Darmstadt
{hoffmann, voelkmann}@informatik.th-darmstadt.de

1 Motivation

With *Cellular Processing (CP)* we mean the mapping of applications onto the CA model, the implementation and the execution/simulation on a system with an appropriate environment (compiler, simulator, control, vizualisation, analysis). Especially we want to support applications, where the cells are complex (a data record) and the array consists of a large number of cells or the CA field is 3-dimensional.

Software simulations take a long time, e.g. if one cell operation takes 10 microseconds and the field is consists of 512^3 cells, the time to compute one generation will be 22 minutes. 100 generations will then need 40 hours. For the broader use and acceptance of the CA model for practical applications, the rate of generations per second must be much higher, to allow online vizualisation and direct user interaction.

At present, the following requirements can only be fulfilled if multiprocessor systems or hardware support are available.

- large number of cells
- large or complex cell states
- complex rules
- many neighbours (neighborhood distance ≤ 1, 3 dimensions)
- many generations per experiment
- many similar experiments with parameter variations
- online vizualization and analysis
- fast user interaction rates.

In the project "Hardware supported simulation system for the graph based and three dimensional cellular processing" supported by the DFG, we are developing a special coprocessor which shall accelerate the simulation by two orders of magnitude compared to software simulations on standard single processor systems. The design is based on the experience we gained through the realizations of the CEPRA-8 [1,2] and CEPRA-1L[3] designed for the two dimensional support.

2 Some Architectural Principles

Memory Table. The state bits of the center cell and the neighborhood cells are concatenated to an address for a memory. The memory holds the rule as a truth table with indexed access through the address. This technique is limited to a few bits per cell because of the limited memory capacity. In the 3-dimensional case and Von-Neumann neighbourhood the number of memory inputs is 7 multiplied by the number b of bits per cell. Thus the needed memory capacity is

$C = 3Dn * 2^{7n}$. (e.g. $n = 3D4 : C = 3D128$ Megabytes)

Local Processing. For simplification we consider an array of n^2 cells. Each processor computes a subarray of size n^2/p. In order to reduce the cost of the architecure, the processors shall be connected to a common communication bus.

The computation inside the processor is sequential and takes t_F for one cell. The total *computation time* for all subarrays will be

$T_F = 3D(n^2/p)t_F$.

At the boarders of the subarrays cell state information must be interchanged. In the case of the Von-Neumann-Neighbourhood (without wrap around the field) the number of such cells is $4n(\sqrt{p} - 1)$.

The *communication time* will be

$T_{COMM} = 3D4n(\sqrt{p} - 1) \cdot t_{COMM}$,

where t_{COMM} is the time for one state transfer between processors.

The following characteristics are generally of interest, which depend on the ratio t_F/t_{COMM}:

- time for a new generation $T = 3DT_F + T_{COMM}$
- percentage of communication T_{COMM}/T
- speed up $T(p = 3D1)/T(p)$
- efficiency S/p.

Example. $p = 3D256$ and $n = 3D512$ and $t_{COMM} = 3D1\mu s$:

$t_F = 3D50\mu s \Rightarrow$ T =3D 51 ms + 31 ms =3D 82 ms communication=3D37%, speed-up=3D160, efficiency=3D0.62

Valuation. This architecture has good performance as long as communication time is relatively short and not too many processors are used.

Most applications need frequent output/input of the whole generation to/from video store, global data store or disk. The main disadvantage of this architecture is the long time for output/input due to the distribution of the cell data.

Central Store and Pipelining. Two central stores $MEMORY_{old}$ and $MEMORY_{new}$ hold the generations. The cell states from the old generation are read sequentially via shift registers into a 3×3 neigbourhood-field (window) with direct acces to 9 cells. The shift registers hold two complete lines of the field (NORTH and CENTER cells). With this shifting technique a new neigbourhood-field can be set up in every store cycle. The neigbourhood-field is the input to a pipeline which computes the rule stepwise through the pipeline stages. The output of the pipeline is stored in $MEMORY_{new}$. Before the next generation is computed, the two memories are interchanged.

Valuation. Advantages of this architecture are quick access to the = central store for input/output and the computation time of one read/write cycle (at best) for one cell operation (COP). A difficulty is that the stages of the pipeline must be especially designed for the application and must work on high speed.

Interleaved Processing. This architecture consists of a central store and p processors with data buffers and local stores. The components are connected by a common bus. Cell state information is first distributed via the bus to the input buffers. Thereby advantage is taken by the *broadcasting principle* because a cell state is needed in 4 (von Neumann neighbourhood) to 8 (Moore neighbourhood) processors as neighbourhood information. Each processor starts its computation when the neighbourhood information is completely local to the processor. The starting points of the computations are cyclically interleaved. (p time steps =3D one cycle). Each processor should compute his result in less than one cycle (computation time limit =3D one cycle). The results are cyclically interleaved transferred back to the central store and at the same time to those processors, who need this information as new neighbourhood information.

Valuation. Processors have enough time (one cycle) to compute their results, meaning that low cost processors with optimal sequential algorithms may be used. One time step lasts 4 read/write cycles (NORTH; CENTER; SOUTH; write result) if 3×3 buffers are used in the processors. If in the processors larger buffers (complete lines) are available and the write cycle can be performed in parallel with the read cycle, the time step can further be reduced to one read/write cycle like in a pipeline architecture.

3 An Architecture for 3 Dimensions

In our project we have designed a copressor for 3-dimensional cellular processing. It is based on shifting in 3 dimensions and pipelining.

The architecture consists of (1) a *memory* holding one generation, (2) a *plane shifter*, (3) a *line array shifter*, (4) a *computing window* and (5) *p processing =* *elements*.

The xyz-memory holds n^3 cells. p cells in z-axis shall be processed in parallel. The xy-plane shifter consists of 3 z-planes. Each plane holds p cells. The bus between the xyz-memory and the xy-plane shifter is p cell wide.

The line array shifter is of size n in x-axis, 3 in y-axis and $p+2$ in z-axis. A xz-slice output from the plane shifter can be loaded in n steps into it. It can perform shifts in x direction, thereby loading the computing window.

The computing window holds p center cells (to be computed) in z-axis an= d all their neighbours. It is of size $p+2$ in z-axis, 3 in x-axis and 3 in y-axis. It is loaded shift by shift from the line array shifter. The p processing elements have parallel access to the window and are internally pipelined with the shift clock.

Succesive planes are loaded from the memory to the plane shifter, which consist of 3 planes (i=3D1,2,3). In the start-up plane-cycle all 3 planes are loaded from memory. In the following plane steps the planes i are shifted to planes $i+1$ and a new memory plane is loaded into the plane 1.

Between each plane step approx. n line steps are performed. In each line step the plane shifter is shifted in y-direction and thereby $p+2$ lines are loaded into the line array shifter.

Between each line step approx. n window steps are performed. In each window step the line array shifter is shifted in x-direction and thereby $3*3*(p+2)$ cells of the computing window are updated.

The total capacity of memory cells is

$C = 3Dn^3 + 3pn^2 + 3(p+2)n + 9(p+2)$.

The computing time is the sum of the times needed for (1) loading the plane shifter, (2) loading the line array shifter and (3) the time to compute a new cell state which is overlapped with the writing back into the memory. The bus between memory and plane shifter is p word wide, and between plane shifter and line array shifter $p+2$ word wide. The total execution time will result in

$T = 3D3n^3/p$.

After adding of buffers and interleaving of the memory the execution time may be reduced to n^3/p.

Valuation. This architecure is easy scalable in p and is not very expensive to realize. Using field programmable logical array for the processing elements the required performance can cost effectively be realized.

References

[1] R. Hoffmann, K.-P. Völkmann, M. Sobolewski:
The Cellular Processing Machine CEPRA-8.
Proceedings of the VIth International Workshop on Parallel Processing by Cellular Automata and Arrays, Potsdam, Sept. 1994.

[2] R. Hoffmann, K.-P. Völkmann:
CEPRA-8L: A Celluar Processing Machine

in: Massiv Parallelism Hardware, Software and Applications, Editor M. Mango Furnari, Capri 1994

[3] Th. Hahn, Chr. Hochberger, R. Hoffmann, K.-P. Völkmann
Der CEPRA-1L Zellularprozessor
submitted for Workshop on Custom Computing Dagstuhl 6/96

Firing Squad Synchronization Problem in One and Two Dimensional Reversible Cellular Automaton

Katsunobu Imai,* Taichi Adachi, Shinichi Furusaka and Kenichi Morita
Faculty of Engineering, Hiroshima University
Higashi-Hiroshima-shi, 739 Japan

April 3, 1996

Abstract

The Firing Squad Synchronization Problem (FSSP) is to construct a one-dimensional three-neighbor cellular automata of arbitrary finite length such that one of the end cell (general) makes all the other cells (soldiers) be in a particular state (firing state) at a certain time. The problem was first solved by Minsky and McCarthy. They constructed a solution synchronizing n cells in $3n$ steps using a divide-and-conquer method. At present 6-state minimal time solution has been given by Mazoyer[1]. Two-dimensional case was studied by Rosenstiehl, and Kobayashi showed some special classes of figures that have "linear" firing time [2].

We studied FSSP on reversible (i.e., backward deterministic) cellular automata (RCA). First we proved that, in the case of RCA, there is no solution under the usual condition for FSSP where the firing state is only one. So we defined a little weaker condition suitable for RCA which is as follows.

[A synchronizing condition for one-dimensional RCA]

1. Define a set $F(\subset Q)$ of finite number of firing states, and regard that the cells synchronize if the state of each cell is in F.

*E-mail: imai@ke.sys.hiroshima-u.ac.jp

2. Not only the general and soldier cells but the other cells are allowed to change its internal states.

Under this condition, we showed that $3n$ time solution can be embedded in an RCA with 99 states that satisfies the above condition (Fig.2). Constructing solution, we used the framework of partitioned CA (PCA)[3], which can be regarded as a subclass of CA. It has been proved that local function of a PCA(Fig.1) is one-to-one iff their global function is one-to-one (i.e. reversible). So we can construct a RCA easily.

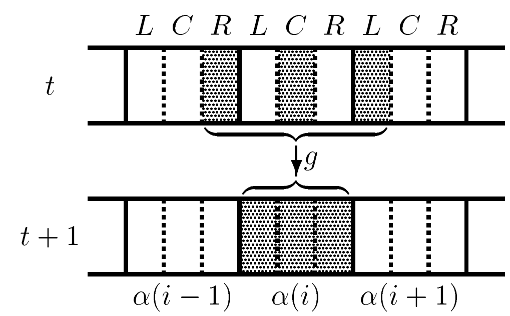


Figure 1: Domain and range of a local function of one-dimensional PCA.

The same approach can be usable for two-dimensional cases. We extend the synchronizing condition into two-dimensional RCA and embeded the solutions of Kobayashi and Rosenstiehl in two-dimensional reversible PCA.

References

[1] Mazoyer, J.: A six-state minimal time solution to the firing squad synchronization problem, *Theoret. Comput. Sci.*, **50**, (1987) 183–238.

[2] Kobayashi, K.: The firing squad synchronization problem for two-dimensional arrays, *Inform. and Control*, **34**, (1977) 177–197.

[3] Morita, K., and Harao, M.: Computation universality of one-dimensional reversible (injective) cellular automata, *Trans. IEICE Japan*, **E72**, (1989) 758–762.

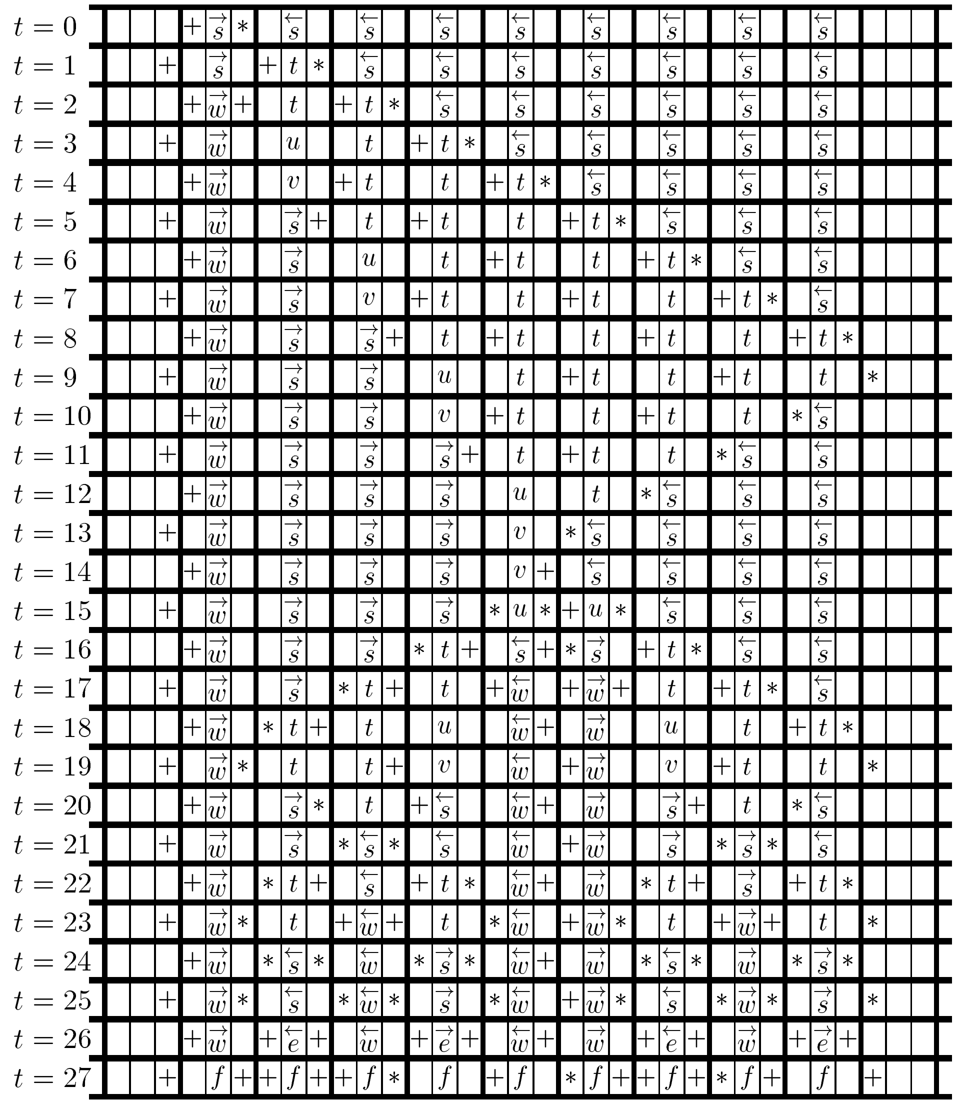


Figure 2: Synchronization of 9 cells by the 99-state reversible 1D PCA P.

Algebraical constructions of reversible generalized Pascal triangles and one-dimensional cellular automata

Ivan Korec, Bratislava

Generalized Pascal triangles (GPT) are defined so that they immediately correspond to computations of one-sided one-dimensional cellular automata (1DCA) from finite initial configurations. (The mentioned automata have binary local transition functions and neighborhood types $(0,1)$ or $(-1,0)$.) Formally, GPT are associated to some finite algebras $\mathcal{A} = \langle\, \mathbf{A}; *, \mathrm{o}\rangle$ of signature $(2,0)$ and words $w \in \mathbf{A}^+$. $\mathrm{GPT}(\mathcal{A}; w)$ is constructed top-down from $\mathcal{A}$ and w similarly as the classical Pascal triangle is constructed from $\langle \mathbb{N}; +, 0\rangle$ and the constant 1. In the relation to one-sided 1DCA the binary operation of the underlying algebra corresponds to the local transition rule and its constant to the quiescent state. The relationship to other one-dimensional CA also can be established, although it is not so immediate.

A GPT will be called reversible if it also can be locally constructed bottom-up, using a binary operation. An algebra will be called reversible if all its GPT are reversible. Several kinds of reversibility of algebras will be introduced; all of them lead to reversible 1DCA. (However, general reversibility of 1DCA is not completely covered by them because it allows reverse local transition rules with more than two arguments.)

Some constructions of reversible algebras from groupoids satisfying cancellation laws (both or one of them) are presented. Elements of the constructed algebra belong to the cartesian square of the groupoid, and the constructed operation is determined by a formula $(x_1, x_0) * (y_1, y_0) = (T_1, T_0)$, where T_1, T_0 are suitable terms composed from x_i, y_i, and the groupoid operation symbol. Further, two constructions of reversible algebras from another reversible algebras will be given; we call them permutation of values and permutation of arguments. These constructions preserve base sets, and usually also the constants o of the algebras. (Hence only permutations of $\mathbf{A} \setminus \{\mathrm{o}\}$ are usually considered, and nontrivial results can be obtained only for $\mathrm{card}(\mathbf{A}) > 2$.) The constructions are very simple from the algebraic point of view but nevertheless they can substantially change the types of the corresponding GPT (and also corresponding 1DCA). For example, we can obtain an algebra with GPT containing all words from $\mathbf{A}^+$ in its rows from a semilinear algebra $\mathcal{A} = \langle\, \mathbf{A}; *, \mathrm{o}\rangle$ of degree 1. (We say that $\mathcal{A} = \langle\, \mathbf{A}; *, \mathrm{o}\rangle$ is a semilinear algebra of degree 1 if the set of rows of every of its GPT is a regular language.) The idea can be immediately applied also to the classical 1DCA (i.e., those with the neighborhood type $(-1,0,1)$), and also to higher-dimensional cellular automata.

Linear Cellular Automata over Fields and Factorization of Polynomials

Ehler Lange

It was shown by v. Haeseler, Peitgen, Skordev that the self-similar evolution patterns of so-called Fermat-Cellular Automata can be completely decoded by means of substitutions or hierarchical iterated function systems. Furthermore, they provided a simple algorithm for setting up these substitutions. Linear CA over finite fields clearly belong to the class of Fermat-CA and results concerning the algebraic properties of the induced substitutions are presented in the talk. Also we describe a remarkable relationship between such substitutions and the factorization algorithms of Berlekamp and Niederreiter for polynomials over finite fields. We finally obtain a new approach to both of these factorization methods.

References

[1] E. R. Berlekamp: Factoring Polynomials Over Finite Fields, Bell System Tech. J. 46:1853-1859 (1967)

[2] F. v. Haeseler, H.-O. Peitgen, G. Skordev: Linear cellular automata, substitutions, hierarchical iterated function systems and attractors, Univ. Bremen, Inst.Dyn.Syst., Rep. 254 (1991); in: Fractal geometry and computer graphics, ed. J.L. Encarnacao et al, Springer 1992

[3] E. Lange: Cellular Automata, Substitutions, and Factorization of Polynomials Over Finite Fields, in: S. D. Cohen, H. Niederreiter eds, Proceedings of the 3rd International Conference on Finite Fields and Applications 1995 (to appear)

[4] H. Niederreiter: Factorization of Polynomials and some Linear-Algebra Problems Over Finite Fields, Linear Algebra Appl. 192:301-328 (1993)

Chaos in cellular automaton systems with majority rule *

Danuta Makowiec,
Instytut Fizyki Teoretycznej i Astrofizyki, Uniwersytet Gdański, POLAND
e-mail: fizdm@univ.gda.pl

1. Introduction

The roughest, qualitative property which is expected to be elaborated when a complex dynamical system is taken into considerations, consists of predicting of a system long time behaviour. Systems which have the unique limit state and converge to it from any initial conditions may be said to forget everything, when time tends to infinity. In these cases the statistical analysis of time series, by dealing with time averages, provides an unique ergodic measure [1,2,3]. Dynamical systems usually exhibit some transient behaviour before they reach an asymptotic motion. In particular, if there exists more than one stationary state in the system, then the transient process can be extremely sensitive to the properties of an initial state and depending on properties of an initial state, is headed to a distinct stationary state. Cellular automata seems to be the simplest complex system for which this process can be observed

Generally, cellular automata can be seen as Markov chains— the systems of transition probabilities which converts any initial probability distribution defined in the space of lattice configurations into, so-called, the natural stationary measure [1,2,4]. The problem of construction of the probability distribution from a system of positive transition probabilities has been solved in *lattice* $\times$ *time* configuration space [3,5]— the stationary Markov chain is the limit Gibbs distribution with the Hamiltonian directly determined by transition probabilities. However, this description fails when a cellular automata local rule is of a deterministic type and also it does not give the measure on the lattice configuration space itself. It is known that the natural measure concentrates on "natural" configurations, corresponding to, so-called ground states [5], of the, in general, unknown Hamiltonian. Therefore, the deterministic cellular automata offer limit models for investigation the most fundamental problem of equilibrium statistical physics, which

* *Work partially supported by KBN PB 129/P03/95/09*

is to describe all Gibbs measures for a given Hamiltonian and observe relations between them and their coexistence.

In particular, system of two-level spins occupying a square lattice, which locally interact with majority vote (*Centre-East-North* neighbours $\rightarrow$ *Centre* spin, called Toom model [6,7]) though its simplicity, allow to study the mentioned problem. This model is known [8] that any initial configuration is attracted to few, very regular configurations. Moreover, if an initial configuration is close to the maximal disordered configuration what means that each spin takes any of its states either *up* or *down* with probability $\frac{1}{2}$, then the critical behaviour is observed. It denotes that there is an uncertainty which of ground state configurations attracts a given initial configuration. Therefore, in the following we will concentrate on ground states attractor boundaries for the Toom model considered on a square lattice imposed with periodic boundary conditions. Using the modified box-counting method [4] we show the fractal character of the attractor boundary.

The probabilistic Toom model is equivalent to the equilibrium statistical mechanics of the Ising model on *hcp* [3] or fcc [9,10] three dimensional triangular lattice.

2. Fractal character of boundary

It appears that for each lattice size L there exists a p interval around $\frac{1}{2}$ where the considered model stabilises with any of following configurations:

— homogeneous with all spins *up*;

— homogeneous with all spins *down*;

— *mixed* configuration— the final pattern consists of two homogeneous regular clusters in stripe shape, going vertically, horizontally or along the West-South to North-East diagonal;

— *zigzag* configurations— there are two separated areas with different mixed configurations on a final pattern;

— there are three large areas on the lattice with different mixed states;

...

At $p = \frac{1}{2}$ probability that cellular automata is led to any of the first three types from the above list seems to be independent of lattice size and approximately stabilises at large L with $\frac{1}{3}$ [8]. There are no priori hints to which of these configurations the system is driven.

Let us say that our initial conditions are uncertain by the amount ε in the sense that when we say that the initial configuration is σ_0, what we really know is only that the initial configuration σ differs form σ_0 by one spin state. Thus the radius of initial uncertainty is

$$\varepsilon = |\sigma - \sigma_0| = (\frac{1}{L})^2 \tag{1}$$

We call an initial configuration σ_0 ε-certain if it yields all configurations distinct by ε to the same attractor [4]. An initial configuration σ_0 is ε-uncertain, if there exists a configuration which lies within a distance ε and which yields to different attracting point. In this case, one can say that the ε-uncertain configuration lies within a distance ε to the basin boundary.

For typical initial configurations the probability to obtain an ε-uncertain initial condition is the fraction of the volume of the configuration space which lies within ε distance to the boundary. Denote this fraction by $f(\varepsilon)$. It is known that [4] for a simple non fractal boundary $f(\varepsilon)$ scales linearly with ε, $f(\varepsilon) \sim \varepsilon$. However, when the boundary is fractal, then $f(\varepsilon)$ has the following scaling [4]:

$$f(\varepsilon) \sim \varepsilon^{\alpha} \tag{2}$$

where α, called the *uncertainty exponent* [4], is less than 1.

We estimate the uncertainty exponent α on the basis of the following experiment. For a given lattice size, L, we pick an initial configuration at random by tossing at each lattice site *up* spin state at the same probability p. Then we perturb the initial configuration by flipping one spin state and repeat it one-by-one along all sites of the lattice. So that, the uncertainty of initial configuration is (1) and we deal with all configurations within the initial uncertainty. We record which attractor (all spins *up*, all spins *down*, *mixed* or *zigzag* configuration) each of these initial configurations goes to. If all of them do not all go to the same attractor, then we count the original initial configuration as uncertain. We randomly choose 100–150 initial configurations for $p = 0.51 \pm 0.003$. Then we determine the fraction $\hat{f}(\varepsilon)$ of these that are uncertain. We repeat this experiment for different values of $\varepsilon : (\frac{1}{12})^2,\ (\frac{1}{24})^2,\ (\frac{1}{36})^2,\ (\frac{1}{48})^2,\ (\frac{1}{60})^2,\ (\frac{1}{72})^2$. According to [11, 12] we can expect that $f(\varepsilon)$ is approximately proportional to $\hat{f}(\varepsilon)$, so that α can be extracted from scaling $\hat{f}$ with ε. Fig.1 is the plot of the results on log-log axes. We have too little points to make the fit precise. However, we can learn about tendency whether or not α is lower than 1. The slope of the best line fit gives the power of ε as $\alpha \approx 0.014$ what denotes the " huge" fractality of the boundary.

There are some more regularities that have been observed in the performed experiments which allow to say more about relations between stationary configurations. They can be summarised as follows:

1. There was no single experiment where a configuration primary driving to *mixed* stationary configuration was ε-certain. There were always large areas in the initial configurations which were sensitive to the spin flip. There was only one experiment in which the configuration yielding all spins *down* occurred ε-certain. These observations were held independently of lattice size L. They suggest that initial configurations which provide all spins *downs* or *mixed*

final configurations are ε-uncertain independently of lattice size. Initial configurations which led to all spins *up* stabilisation could appear either ε-certain or ε-uncertain. However, the probability that such a configuration is ε-certain depends on L, Fig.2.

2. We noticed a large discrepancy between numbers of unstable spins in ε-uncertain configurations, Fig.3. The unstable spins usually form kind of islands in the sea of stable spins. Anyway, the islands consisting of one isolated spin have been observed, too. Moreover, a single spin flip in any initial configuration which unperturbed led the system to the *mixed* stationary configuration, provided the stabilisation at either any of homogeneous configuration or a mixed configuration, however *always* with the same geometric properties. For example, initial patterns with two vertical stripes were changed to all spins *up*, or all spins *down*, or two vertical stripes but with the different from original size. On the other hand, the single spin perturbation of a configuration which unperturbed led to any homogeneous configuration could effect all geometric types of *mixed* stationary configurations, but the opposite homogeneous state had not been observed.
3. The most uncertain initial configuration observed in our experiments was the configuration originally leading to *zigzag* stabilisation. However, we observed *zigzag* stationary configuration only twice— once at $L = 36$ and once at $L = 60$.

3. Closing remarks

In the following we found that typical random spin configurations initially distinct by a single spin state can provide different stationary configurations. Our rough estimation for the uncertainty parameter α of the boundary between basins of attractors provides for α the number much less than 1 what suggests that the boundary between basins of attractors is fractal. Therefore, the observed final state sensitivity which makes inefficient any predictions to cellular automata stabilisation, cannot be improved by increasing the lattice size. However, as these properties are ruled by local fluctuations, one can expect that the chaotic behaviour should be less sensitive to one spin flip with the growing lattice size.

Moreover, because of observed distinct qualitative properties of stabilisation just because of a single spin perturbation, we strongly support the opinion that *mixed* and any other higher complex configurations are configurations of higher energy than the two basic homogeneous ones. We believe that so huge presence of the stabilisation with these higher energy configurations can be connected with the geometric properties of boundaries in a space of periodic configurations on the infinite lattice. This extra non-local interaction which demands the complete adjustment of the final configurations to the boundaries, effects in lowering of energy contained in configurations with stripes in the proper shape. In consequence, the energy gap between the ground and some of the first level states becomes indistinguishable by experiment. We have some preliminary results that violating the square periodic boundaries, for example, be setting random size of each row, the final configurations space reduces drastically. In the mentioned example there were present only both homogeneous configurations and horizontal stripes configurations.

It has been proved rigorously [5,13] that even in the thermodynamic limit when the lattice period grows up to infinity the stationary state somehow remembers its periodic boundary conditions .

Since the famous two-dimensional Ising model, that is the simplest lattice model which deals with ferromagnetic interactions, which is governed by the Hamiltonian

$$\mathcal{H} = -\mathcal{J} \sum_{|i-j|=1} \sigma_i \sigma_j, \qquad \mathcal{J} > 0 \tag{3}$$

yields two homogeneous ground states, we are closing with the conclusion that the Hamiltonian (3) can be considered as the first approximation for the energy contained in the finite Toom cellular automata model.

Acknowledgement

We are grateful to organisers of Cellular Automata Workshop for their kind invitation and hospitality.

References

1. J-P.Eckmann, D.Ruelle *Rev.Mod.Phys* **57** (1985) 617
2. D.Ruelle *Chaotic evolution and strange attractors* Cambridge University Press, 1989.
3. J.L.Lebowitz, C.Maes, E.Speer *J.Stat.Phys* **59** (1990) 117
4. E.Ott *Chaos in dynamical systems* Cambridge University Press, 1993.
5. Ya.G.Sinai *Theory of phase transitions: rigorous results* Akademiai Kiado, Budapest, 1982
6. A.L. Toom, N.B.Vasilyev, O.N.Stavskaya, L.G.Mityushin, G.L. Kurdyumov and S.A.Prigorov *Discrete local Markov systems* in *Stochastic cellular systems:ergodicity, memory, morphogenesis* edited by R.L.Dobrushin, V.I.Kryukov and A.L.Toom, Manchester University Press, 1990.
7. A.Toom *Cellular Automata With Errors: Problems for Students of Probability* in *Topics of Contemporary Probability and its Applications* J.Kaurie Snell, ed. CRC press, 1995.
8. D.Makowiec *Acta Phys.Pol.* **B** to appear (1996) *Cellular automata with voting rule*
9. E.Domany *Phys.Rev.Lett.* **52** (1984) 871
10. E.Domany, W.Kintzel *Phys.Rev.Lett.* **53** (1984) 311
11. C.Grebogi, H.E.Nusse, E.Ott, J.A.Yorke *Basic Sets : Sets that Determine the Dimension of Basin Boundaries* in *Dynamical Systems* edited by J.C.Alexander, Lecture Notes in Mathematics No 1342, Springer- Verlag, Berlin 1988 (pp 220-250)
12. S.W.McDonald, C.Grebogi, E.Ott, J.A.Yorke *Physica* **D17** (1985) 125
13. G.Giacomin, J.L.Lebowitz, Ch. Maes *J.Stat.Phys.* **80** (1995)

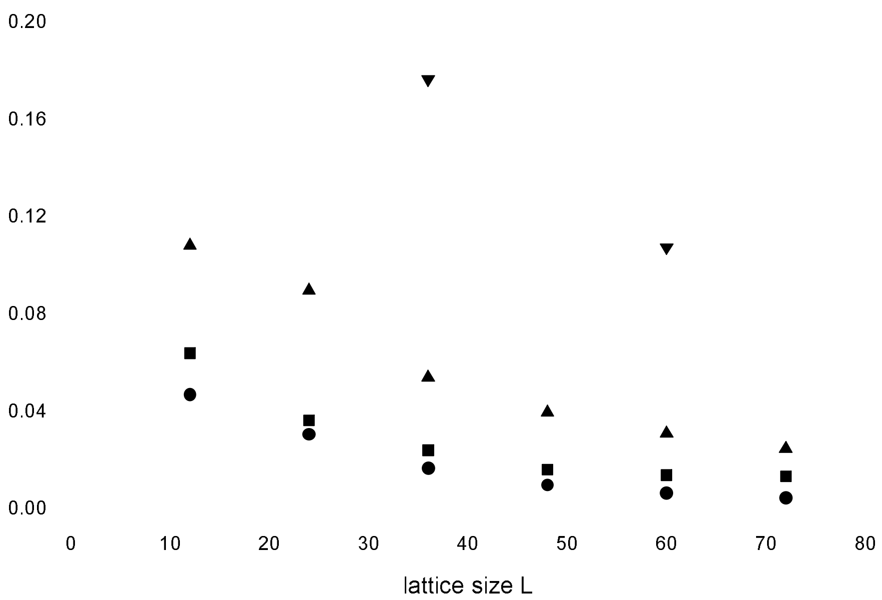


Fig.1: α estimation

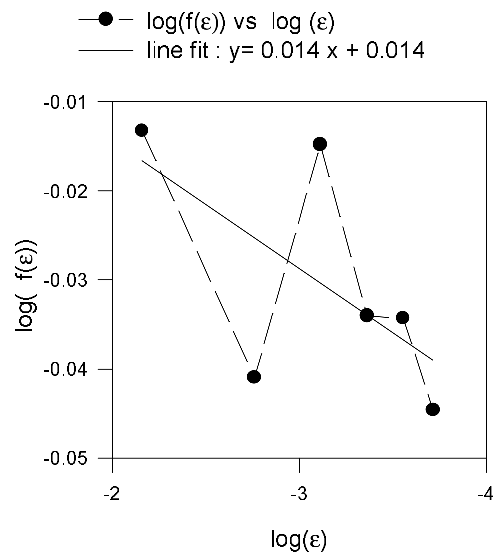


Fig.2: Fraction of ε–certain configuration among events with *up* stabilization

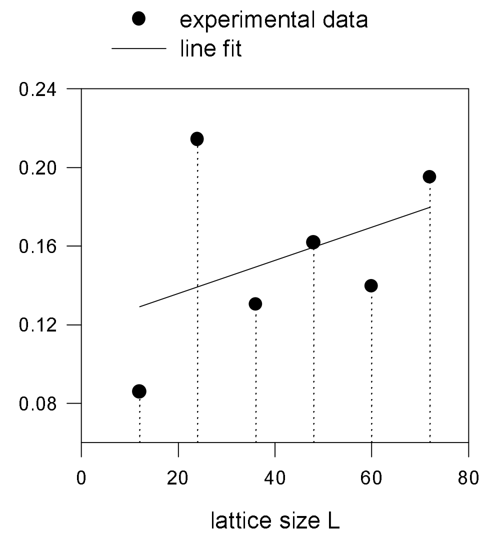
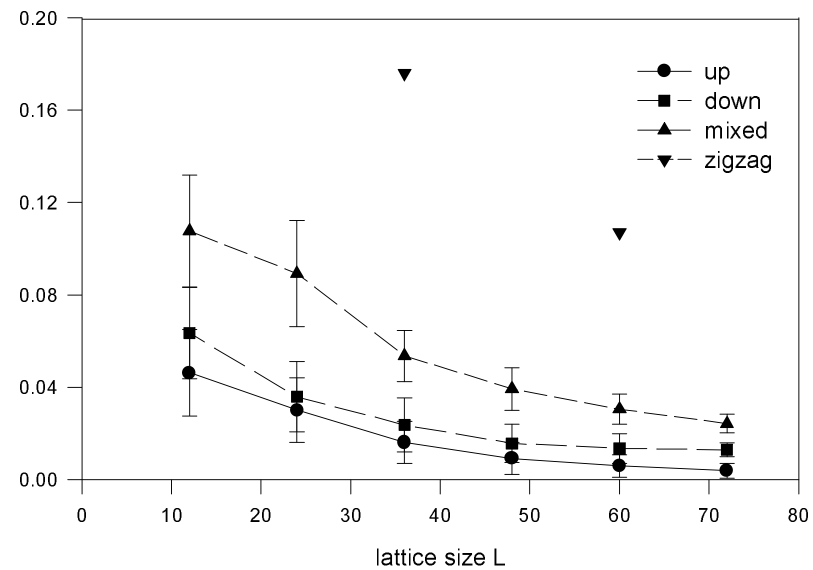


Fig.3: Fraction of unstable spins in ε–uncertain configuration (with errors)



Towards a new approach
to universality about Turing machines

Maurice MARGENSTERN and Lioudmila PAVLOTSKAÏA
LITP/IBP, Moscow Institute
LRIM (Metz University), of Power Engeneering
Metz University Institute of Technology

Key words : computability, universal machines

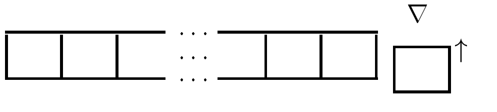
Our talk will introduce a new approach towards universality in the field of classical Turing macines, *i.e.* deterministic Turing machines with a single head and a single tape, infinite in both directions.

This approach is motivated by the relatively small size of universal Turing machines obtained in the eighties by Yu. Rogozhin, see [4]. The smallest known classical Turing machine there appeared as a machine with only 23 instructions.

How can this number be lowered?

1. A new notion

Our basic idea, see [2], consists in setting together a Turing machine with a finite automaton in the following way. Consider the initial datum of Turing machine M as a word w. After a certain time of computation, if the machine did not reach two indentical configurations inside the initial segment I delimited by w at initial time t_0, then its head makes an attempt to read the next symbol *outside* of I. If this attempt is made on the left side, we consider that the machine halts. If it is made on the right side, we consider that the asked letter is supplied by a given finite automaton, say A, as illustrated by the following figure.



Notice that A always provides the blank symbol, a classical machine is obtained. And so, this new notion generalizes classical Turing computations.

2. The results

Rogozhin's machines are constructed according to the tag-system technique introduced in the last fifties and early sixties by several people.

In [3], a universal Turing machine is constructed with 17 instructions thanks to its association with a finite automaton which produces a periodic infinite word. Below is given the program of this machine, say F. It should be noticed that machine F does not simulate a tag-system but a two-register machine, following the simulation pattern defined in [1] in order to construct universal classical Turing machines with only two left instructions.

Below also, is the program of machine G which, associated with a suitable finite automaton, is able to simulate any classical Turing machine. Machine G is thus universal. It has only eight instructions:

	0	1	2	3	4	5	6
1	$5R$	$4R2$	$0L$	$5R$	$1L$	$0L$	$R3$
2	$5R$	$4R$		$5R4$			
3	$5R$	R	$5R2$	$5R$			
3	R		$L1$	$1R$			

	0	1	2	3	4
1	$1L$	$0R$	$R2$	$4L$	$3R$
2		R		$4L1$	$3R$

By loosing a bit more the simulation performed by machine G, a machine with only *five* instructions, say H can be constructed which is universal when associated with a suitable finite automaton. Here is its table:

	0	1	2
1	$1L$	$0R$	$R2$
2		R	$1L1$

In [2], tables for automata associated to, respectively, machines G and H are provided with full proofs of the correctness of the computation starting from an appropriate encoding of the register initial contents.

Acknowledgements

This work was granted by NATO Scientific Affair Division, *HighTech. EV N*∘950525.

References

1. Margenstern M. et Pavlotskaïa L. Deux machines de Turing universelles à au plus deux instructions gauches. *CRAS, Paris*, **320**, I, 1395-1400 (1995)
2. Margenstern M. et Pavlotskaïa L. Vers une nouvelle approche de l'universalité concernant les machines de Turing, LITP/IBP research report N°95.58 (1995)
3. Pavlotskaya L.M. On machines, universal by extensions. *TCS special issue UMC*, to appear, (1996)
4. Rogozhin Iu.V. Sem' universal'nikh machin T'juringa. Matematicheskie Issledovanija, 69, Kichinev, Moldova, 76-90, (1982)

Waves in the BZ-reaction: experiments and CA simulations

Mario Markus, André Czajka and Torsten Schulte

Max–Planck–Institut für molekulare Physiologie
Postfach 102664, 44026 Dortmund, Germany

In everyday-interaction between experimentalists and theorists, we performed measurements and CA simulations of periodic and turbulent waves in an excitable medium.
The experiments were carried out with the ruthenium-catalyzed Belousov-Zhabotinsky reaction, which has the advantage that its periodic and chaotic modes can be controlled by light [1-3] or by oxygen [4]. Simulations were performed with a cellular automaton based on a quasi-stochastic spatial discretization [5]. Such a discretization is used to solve the problem of numerical anisotropy involving the propagation from cell-shape into wave-shape. The influence of light or oxygen was incorporated into the CA rules by considering that these agents displace the $\dot{u}=0$ nullcline (u: autocatalytic substance) towards lower u-values.

The following phenomena were investigated:
i) Target patterns and spiral waves. A calibration, allowing to express CA units in terms of physical units, leads to quantitative agreement between measurements and simulations.
ii) Wave–splitting (backfiring) after a light pulse, leading to self–annihilation of a spiral, as well as to soliton–like behaviour.
iii) Wavefront instabilities leading to spatio–temporal chaos, namely aperiodic 'rippled' wavefronts, as well as wave disruptions. Experimentally, we found that these phenomena occur only if above the gel, which immobilizes the catalyst, there is an aqueous solution with all reactants except the catalyst. The CA calculations — including these two compartments — yielded patterns comparable to the experimental observations.

[1] Kuhnert, L., Nature 319, 393–394 (1986)
[2] Markus, M., Nagy–Ungvarai, Zs., and Hess, B., Science 257, 225–227 (1992)
[3] Markus, M., Kloss, G. and Kusch, I. Nature 371, 402–404 (1994)
[4] Czajka, A., Diplomarbeit, Universität Dortmund (1996)
[5] Markus, M. and Hess, B., Nature 347, 56–58 (1990)

CA simulations and observations in biological and geological processes. Is there evidence for class 4 in nature?

Ingo Kusch[1], Mario Markus[1], Pedro Almeida[2] and António Ribeiro[2]

[1]Max-Planck-Institut für molekulare Physiologie
Postfach 102664, 44026 Dortmund, Germany

[2]Departamento de Geologia, Faculdade de Ciências
Universidade de Lisboa;
58 rua da Escola Politécnica, P-1200 Lisboa, Portugal

We developed CA with binary output for simulations of: i) the pigmentation of mollusc shells; ii) stick-slip processes in earthquake dynamics; and iii) two- and three-dimensional Turing patterns.

The CA for shell pigmentation and for Turing patterns were constructed via a one-to-one assignment of CA rules to interaction arrows in a system based on Turing's [1] morphogenetic model.

Excellent agreement of CA simulations is obtained for a large number of natural mollusc shells. The patterns can be classified using Wolfram's nomenclature (classes 1 through 4). In particular, class 4 CA agree with mixed patterns (aperiodic alternation of periodicity and chaos) in real shells. This is remarkable for two reasons: i) it may by an indication for the existence of class 4 in nature; and ii) we obtain class 4 with the same rules leading to other classes, as opposed to the approach with PDEs [2] in which special systems of up to 6 PDEs had to be introduced just to simulate the cases of mixed patterns.

A binary version of a Burridge-Knopoff-type model [3,4] for stick-slip processes in geophysical plate-displacements also yields CA classes 1 through 4. There is agreement of CA simulations with the measured Gutenberg-Richter law, as well as with the observation of a characteristic seism. We find a high sensitivity of the results with respect to the energy dissipation p, transitions between CA classes occuring back and forth within very small p-intervalls. This implies an additional uncertainty for predictions in this system.

Two-dimensional Turing-patterns have since long been assumed to give positional information for genetic pattern-determination in skins and furs. We find transitions from stripes to spots for increasing inhibitory parameters, as well as transitions from squared to hexagonal to disorded patterns for increasing system size. Size-dependence was also obtained in a generalization to three dimensions; here, the CA yielded patterns that can be classified cristallographically, as in solid-state-physics: bcc, fcc, hexagonal, or disorded lattices of spots.

In our biological and geological CA simulations we have well defined control-parameters - e.g. the energy dissipation p - which allow bifurcation analysis. We thus do not have the problem encountered in the parametrization of Wolfram's automata (see [5]), in which the parameter λ has no physical meaning, nor is it associated with a unique set of rules.

[1] Turing, A. Phil.Trans.Roy.Soc. B237, 37-72 (1952)
[2] Meinhardt, H., The Algorithmic Beauty of Sea Shells, Springer, Berlin (1995)
[3] Burridge, R. and Knopoff, L., Bull.Sei.Soc.Am. 57, 341-371 (1967)
[4] Bak, P. and Tang, C., J.Geophys.Res. 94, 15.635-15.637 (1989)
[5] Li, W., Packard, N.H. and Langton, C.G., Physica D45, 77-94 (1990)

Cellular automata and Spatial Machines

Bruno Martin

Laboratoire d'informatique, Signaux et Systèmes de Sophia-Antipolis
I3S-C.N.R.S. - U.R.A. 1376, 650 Route des Colles
B.P. 145, F-06903 SOPHIA-ANTIPOLIS Cedex
`bmartin@unice.fr`

Definitions

We first recall briefly the definition of cellular automata we use in the sequel. A *cellular automaton* is a bi-infinite array of identical cells indexed by $\mathbb{Z}$, the set of integers. Each cell is a finite state machine $C = (Q, \delta)$ where Q denotes a finite set of states and δ the local transition function, a mapping $\delta : Q^3 \to Q$ which gives the next state of the cells in parallel.

The model of spatial machines has been introduced in [4] in order to modelize more properly realistic parallel machines. According to [4], spatial machines capture some basic assumptions as to physical computing machines: some minimal physical volume is required for any entity which participates in the computation, such as a memory bit, a processor and a message bit travelling through space, and a message bit cannot travel faster than the speed of light. The model is supposed to be more realistic than a general cellular automaton since the number of processors is limited to be finite although they can move and send messages in an infinite space.

Informally, a spatial machine is a finite set of finite processors moving and exchanging messages in 3 dimensional euclidian space. Each processor is like a Turing machine's unit of control where the tape has been supressed. They communicate through sequences of bits moving in the 3 dimensional space. Several distinctions about the model have been made according to the number of processors and their mobility. Let us denote by *stationnary* a spatial machine whose processors are all required to be immobile and by *sequential* a two processors spatial machine, one mobile and one immobile (see [3, 4] for further details). It has been proved in [4] that stationnary spatial machines are not more powerful than finite automata and that sequential spatial machines are as powerful as Turing machines.

1 Simulations of cellular automata by spatial machines

In [10] we proved:

THEOREM 1 *There exists a sequential spatial machine capable to simulate any k-states totalistic cellular automaton. Furthermore, if the computation-time of the cellular automaton is $T(n)$, the corresponding computation-time of the spatial machine will be $(T^2(n) + nT(n) + O(1))(\lceil \log_2 k \rceil + O(1))$.*

This result corresponds to a sequential simulation of a given totalistic[1] cellular automaton. We propose an alternative parallel simulation of cellular automata which improves the time-complexity.

THEOREM 2 *There exists a spatial machine capable to simulate any k-states totalistic cellular automaton. Furthermore, if the computation-time of the cellular automaton is $T(n)$, the corresponding computation-time of the spatial machine will be $T(n)O(k\lceil \log_2 k \rceil)$.*

One might think that this time could be easily decreased to $T(n)O(\lceil \log_2 k \rceil)$ but it is not the case since in our simulation we do not encode a cellular automaton's transition function as the transition function of the spatial machine's processors. This simulation permits then to simulate directly any cellular automaton on a spatial machine given the description of the cellular automaton and its input.

2 Applications to universality

Since cellular automata are computation universal [1, 9], we can deduce the universality of spatial machines from the previous simulations. Let us make the following distinctions on the way universality is obtained. First remind that a universal machine U is *computation-universal* if it can compute any partial recursive function. This formal definition authorizes us several ways to prove the universality of a machine model.

The first one is the *universality by simulation* where machine U from model $\mathcal{M}$ can simulate directly the behavior of any given machine A from model $\mathcal{M}'$ on any input. U receives as an input the program of machine A, a reasonable encoding of the input x of A and simulates each step of the computation of A on input x. Consider for instance the constant-time

[1]See [1] for a definition of totalistic cellular automata

simulation of a cellular automaton by a PRAM [8] or by a (parallel) spatial machine. If both U and A are in the same model, we call this kind of universality *intrinsic* (see [1, 9] for two intrinsic universal cellular automata).

The second is the *universality by hereditary*. In this case, machine U from model $\mathcal{M}$ simulates a universal machine U' from model $\mathcal{M}'$ with U' a machine universal by simulation. This is the way universality of a model is usually carried out. Consider for example the simulation of a universal Turing machine by a spatial machine [4], the simulation of a universal cellular automaton by a sequential spatial machine or the simulations of a universal Turing machine by cellular automata [5, 7, 11].

The last one is the *universality by construction* where given the description of machine A from model $\mathcal{M}$ and its input x, a *constructor* C from model $\mathcal{M}$ constructs explicitely this machine. This is the way universality of von Neumann machine is proved (see [2] for further explanation).

3 Decidability and complexity results

The simulation of a Turing machine by a sequential spatial machine provides easily some decidability and complexity results.

The following problem is undecidable :

INSTANCE : Let S be a spatial machine, P a processor and q a state of P.

QUESTION : Does state q appear in the computation of S?

It sufficies to prove the result for a sequential spatial machine since any p processors spatial machine can be simulated by a sequential one [3] and by reduction from the halting problem for Turing machines.

The *Generic Machine Simulation Problem* for spatial machines is P-complete, which is an easy corollary of GMSP for Turing machines. The proof is similar to the one of the P-completeness of GMSP for cellular automata referred as problem A.12.3 in [6].

4 Conclusion

As a consequence of our simulations, we can construct two universal spatial machines (one by simulation and the other by hereditary). Furthermore, the spatial machines are restricted to the framework of Turing-computability since they do not form a massively parallel model of computation but they seem to modelize more properly some features of feasible parallel computers.

References

[1] J. Albert and K. Culik II. A simple universal cellular automaton and its one-way and totalistic version. *Complex Systems*, 1:1–16, 1987.

[2] A. Arbib. From universal Turing machines to self-reproduction in *The universal Turing machine*, pages 177–189. Oxford Science Publications, 1988.

[3] Y. Feldman and E. Shapiro. Spatial machines: Towards a more realistic approach to parallel computation. Technical Report CS88-05, The Weizmann Institute of Science, May 1988.

[4] Y. Feldman and E. Shapiro. Spatial machines, a more realistic approach to parallel computation. *Comm. of the ACM*, 35(10):61–73, 1992.

[5] D. Gordon. On the computational power of totalistic cellular automata. *Math. System Theory*, 20:43–52, 1987.

[6] R. Greenlaw, H.J. Hoover, and W.L. Ruzzo. *Limits to parallel computation.* Oxford University Press, 1995.

[7] K. Lindgren and M. G. Nordhal. Universal computation in simple one-dimensional cellular automata. *Complex Systems*, 4:299–318, 1990.

[8] B. Martin. Efficient unidimensional universal cellular automaton. In *MFCS'92*, number 629 in Lecture Notes in Computer Science, pages 374–382. Springer Verlag, 1992.

[9] B. Martin. A universal cellular automaton in quasi-linear time and its S-m-n form. *Theoretical Computer Science*, 123:199–237, 1994.

[10] B. Martin. Machines spatiales universelles. *Technique et Science Informatiques*, 14(5):551–566, 1995.

[11] A.R. Smith III. Simple computation-universal cellular spaces. *J. ACM*, 18:339–353, 1971.

The Application of C.A. to Economics: Simulation of Economies with Snobs and Conformists

Ken Miura[*] Sobei H. Oda[†] Kanji Ueda[‡] Yasunori Baba[§]

It is usually assumed in economic theories that a consumer's utility depends only on his or her own consumption: a person's preference is not affected by other people's behaviour. However, this is not always the case; to buy a suit, an antique, or a computer, one may consider how many other people purchase the same product. If a person's decision depends on the number of those who make the same decision, it is said that there are network externalities. This paper develops a model of cellular automata to describe markets where network externalities are observed.

This paper describes the dynamics of demand generated by network externalities in terms of cellular automata. The basic idea is regarding each cell as a person whose reservation price for a commodity, the maximum price he or she may pay for the commodity, depends on the number of his or her neighbours who purchased the product in the previous period. This approach regards - in contrast to the equilibrium theory - a market as a place where people repeatedly gather local information, do a small calculation to makes a decision, and put it into practice. As a result of explicitly taking account of difference in locally available information and interactions between information processing and economic activities, the analysis reveals some patterns of the market dynamics: (1) If all are conformists who prefer doing as their neighbours do, a (quasi)equilibrium state (the state where

[*]Faculty of Engineering, Kobe University, 1-1 Rokkodai-cho, Nada-ku, Kobe 657, Japan (E.mail: miura@mi-2.mech.kobe-u.ac.jp).

[†]Faculty of Economics, Kyoto Sangyo University, Motoyama, Kamigamo, Kita-ku, Kyoto 603, Japan (E.mail: oda@cc.kyoto-su.ac.jp).

[‡]Faculty of Engineering, Kobe University, 1-1 Rokkodai-cho, Nada-ku, Kobe 657, Japan (E.mail: ueda@mech.kobe-u.ac.jp).

[§]Research into Artifacts Centre for Engineering, University of Tokyo, 4-6-1 Komaba, Meguro-ku, Tokyo 153.

individuals cease to change their decisions) is usually attained after some periods, but the number of those who buy the product sensibly depends on the initial conditions. (2) The situation is reversed if some are so snobbish that they hate to follow their neighbours: although individuals do not cease to change their decisions, the number of purchasers usually fluctuate around a value which is independent of the initial conditions with small amplitude.

In addition to the introduction of locally different birth and death rules into the life game, we believe, our analysis could shed a new light on the existing theories of marketing and economics: the long-run equilibrium - if there exists such a situation - can be quite different from the one excepted by the equilibrium analysis.

Automaticity of rational functions

Antje Petersen*
Institute for Dynamical Systems
University of Bremen
D-28334 Bremen
antje@mathematik.uni-bremen.de

It is a well known fact that for a rational function $f(x) = \sum_{n=0}^{\infty} s_n x^n = \frac{p(x)}{q(x)}$ with coefficients s_n in a finite field the sequence $(s_n)_{n\in\mathbb{N}}$ is ultimately periodic. Therefore the sequence is k–automatic for all $k \geq 2$. (About k–automaticity see e.g. [1].)

In the following a rational function will be called k–automatic if and only if the sequence defined by its coefficients is k–automatic. Moreover a sequence (rsp. function), which is k-automatic for all integers $k \geq 2$, will simply be called automatic.

For two–dimensional sequences defined by rational functions

$$R(X,Y) = \sum_{n,m=0}^{\infty} s_{nm} X^n Y^m = \frac{P(X,Y)}{Q(X,Y)}$$

with coefficients s_{nm} in a finite field the situation is much more complicated. The description of automaticity we have here is not as simple as in the one–dimensional case.

From [4] we learn that a sequence $\mathbf{s} : \mathbb{N}^2 \to A$ is automatic if and only if each $M_a = \mathbf{s}^{-1}(a)$ is a rational subset of the monoid $\mathbb{N}^2$. These rational sets give a grid–like pattern on $\mathbb{N}^2$, which is similar to the notion of periodicity we have in one dimension.

The next theorem gives a characterization of those rational functions $R(X,Y) \in \mathbb{Z}_m[X,Y]$, where $\mathbb{Z}_m = \mathbb{Z}/m\mathbb{Z}$, which are automatic.

Theorem 1 *Let $m \geq 2$ be an integer. A rational function $R(X,Y) = \frac{P(X,Y)}{Q(X,Y)}$ with $P(X,Y)$, $Q(X,Y) \in \mathbb{Z}_m[X,Y]$ and $Q(0,0) = 1$ is automatic if and only if there exists a polynomial $S(X,Y) \in \mathbb{Z}_m[X,Y]$ such that $Q(X,Y)S(X,Y) = \prod(1 - \delta_i(X,Y))$, where the $\delta_i(X,Y)$ are finitely many monomials.*

This characterization together with the following two Lemmata provides a tool to generalize the results about the automaticity of the Pascal triangle, [2, 6], which is defined by the rational function $R(X,Y) = \frac{1}{1-Y(1+X)}$, and more general about the automaticity of orbits of linear cellular automata [3].

Let p be a prime number.

*Supported by DFG "Graduiertenkolleg Komplexe Dynamische Systeme"

Lemma 1 *Then* $R(X,Y) \bmod p^\alpha$ *is p–automatic for all* $\alpha \geq 1$.

Lemma 2 *If* $R(X,Y) \bmod p$ *is automatic, so is* $R(X,Y) \bmod p^\alpha$.

For the main theorem we need the following notation.
Notation Let $m \in \mathbb{N}$ and let p be a prime divisor of m, then we define

$$\begin{aligned} M_m(R(X,Y)) = \ & \{p : \textit{there does not exist a polynomial } S(X,Y) \in \mathbb{Z}_p[X,Y] \\ & \quad \textit{with } S(X,Y)Q(X,Y) = \prod(1-\delta_i(X,Y)) \textit{ mod } p\}. \end{aligned}$$

Furthermore, denote by $\chi_m(R)$ the cardinality of $M_m(R(X,Y))$.

Theorem 2 *Let* $\chi_m(R)$ *and* $M_m(R(X,Y))$ *be defined as above, then for*

- $\chi_m(R) \geq 2$ *:* $R(X,Y) \bmod m$ *is not k–automatic for any* $k \geq 2$
- $\chi_m(R) = 1$ *:* $R(X,Y) \bmod m$ *is p–automatic, where* $p \in M_m(R(X,Y))$
- $\chi_m(R) = 0$ *:* $R(X,Y) \bmod m$ *is automatic.*

References

[1] J.-P. Allouche, *Automates finis en théorie des nombres*, Expo. Math. **5** (1987), 239–266.

[2] J.-P. Allouche, F. v. Haeseler, H.-O. Peitgen, G. Skordev, *Linear cellular automata, finite automata and Pascal's triangle*, Discrete Appl. Math. (to appear).

[3] J.-P. Allouche, F. v. Haeseler, H.-O. Peitgen, A. Petersen, G. Skordev, *Automaticity of double sequences generated by one–dimensional linear cellular automata*, submitted.

[4] V. Bruyère, G. Hansel, C. Michaux, R. Villemaire, *Logic and p-recognizable sets of integers*, Bull. Belg. Math. Soc. **1** (1994), 191–238; 577.

[5] A. Cobham, *On the base-dependence of sets of numbers recognizable by finite automata*, Math. Systems Theory **3** (1969), 186–192.

[6] I. Korec, *Pascal triangles modulo n and modular trellises*, Computers and Artificial Intelligence **9** (1990), 105–113.

ACL – Building Cellular Processing Kernels Based On Attributes

Ralf Schneider
Microprogramming and Computer Architecture
Technical University of Darmstadt
Alexanderstr. 10, Darmstadt, D-64283, Germany
schneider@informatik.th-darmstadt.de

Abstract

Cellular processing (CP) is defined as an extended interpretation of Cellular Automata (CA) in the sense of an extension of the internal cell state complexity and a broader variety of the cell topologies. The range from a few bits to complex record-like cell state structures with floating point values are covered by the following approach. Also different regular grids are included to map application models in the fields of mathematics, biology, physics and artificial worlds. Simulation (processing and visualization) of these models is realized by Cellular Processing kernels (CP kernels), which provide special aspects and features of CP. Since different target system architectures are used for CP simulation all hardware and application specific features can not be combined in one single CP kernel without loosing efficiency, transparency and portability. This is a major criticism of universal CP kernels which try to cover all features of cellular processing at once. In this paper a new approach is introduced which automatically generates user specific CP kernels based CP attributes and optimization strategies using the high programming model ACL.

Extended Abstract

Cellular models can be defined by a number of CP attributes concerning their *internal data structure*, *algorithmic behavior*, the *target system architecture* and the *optimization strategies*. Cellular processing is done by cellular processing kernels (CP kernels) which consider CP attributes to map cellular models onto specific target system architectures. Single and multi-processor CP systems differ on various aspects and once manually created programs will have to be modified and adjusted for each target system architecture by experienced programmers. Therefore CP kernels should have following properties:

- Best possible adaption to the target system architecture
- Efficient storing of the cellular space
- Additional application specific optimization strategies on algorithmic level

- Portability on source code and abstract level
- Extendibility and software re-usability
- Simple and fast source code without time consuming condition queries (condition-less run-through)

With the new introduced high level programming model ACL (*Abstract Cellular model Language*) these aims will be reached by considering the model properties (algorithmic behavior, internal data structure, optimization strategies) when mapping cellular models onto different target system architectures. The local rules as part of the algorithmic behavior will be described in CDL (*Cellular Description Language*) which has a "PASCAL/C"-like syntax.

Based on CP attributes, the entire CP kernel is generated by ACL in a special transformation process on source code level. The generated CP kernel will be compilable on the chosen target system architecture. This transformation process will be embedded into a programming environment to simplify the generation of CP kernels for the user. ACL can be divided into transformation processes of selections and shaping using substitution and replacement techniques on a LISP-like data structure. ACL is also a meta language level which describes the CP kernel as an abstract program. It is therefore highly flexible to application or hardware specific requirements. The idea of the high level program model ACL will be introduced using a 2D-traffic simulation as an example.

With special target language skeletons (TL-skeletons) defining the target language structures the next transformation step leads to the final CP kernel. This CP kernel is compilable with the standard compiling techniques. Any language as "C", "C++" or "Pascal" can be included as language level for a target system architecture.

With the flexible use of natural formed ACL-commands and the substitution and replacement technique it is possible to construct complex programs for cellular processing on different target system architectures. The ACL-program skeleton and the TL-skeletons provide a very simple mapping of special CP features onto target system architectures. For adaption of new or extended tasks software re-usability is the main criterion even in this area of interests. Due to the structures of the ACL-transformation process it is simple to extend and modify existing CP kernels for special tasks and applications without re-engineering or re-developing.

References

[1] M. Cannataro, S. D. Gregorio, R. Rongo, W. Spataro, G. Spezzano, and D. Talia. A parallel cellular automata environment on multicomputers for computational science. *Parallel Computing*, 99(21), 1995.

[2] J. Darlington, A. Field, P. Harrison, P. Kelly, D. Sharp, Q. Wu, and R. While. Parallel programming using skeleton functions. *Parallel Architectures And Languages, Europe, PARLE 93*, 1993.

[3] K. M. Decker, J. J. Dvorak, and R. M. Rehmann. A knowledge-based scientific parallel programming environment. *Swiss Scientific Computing Center, CSCS-TR-93-07*, 1993.

[4] M. M. Gutzmann and S. Kindermann. Transformation based development of efficient programs for massively parallel architectures. *Dep. of Computer Science, University of Jena/Erlangen, Germany.*

[5] C. Hochberger, R. Hoffmann, and S. Waldschmidt. Compilation of CDL for different target architectures. *Proceedings of the Third International Conference PaCT-9, St. Petersburg*, 1995.

[6] R. Hoffmann. Cellular processing architectures. *Mitteilungen - Gesellschaft für Informatik e.V., Parallel-Algorithmen und Rechnerstrukturen, PARS Workshop Dresden*, 1993.

[7] J. K. Ousterhout. *Tcl and Tk Toolkit.* Addison-Wesley, Reading, Massachusetts, 1994.

[8] R. Schneider. Ein Verfahren zur Laufzeitoptimierung bei der zellularen Verarbeitung. *Proceedings of the PARS Conference Stuttgart*, 1995.

A Note on Firing Squad Synchronization Algorithms

Hiroshi Umeo (Osaka Electro-Communication University)

The famous firing squad synchronization problem(FSSP for short) is stated as follows: Given an array of n identical cellular automata, including a "general" at the left end which is activated at time t = 0, we want to design the automata so that at some future time all the cells will simultaneously and for the first time enter a special "firing" state. Since the birth of FSSP proposed by J. Myhill in about 1957, a lot of synchronization algorithms have been developed. It has been known that Goto was the first-in-the-world designer who developed time-optimum synchronization algorithm, which was reported at Harvard University course note. Since his result was not published as a paper, it has not been circulated widely. The course note was referenced and quoted by several classical papers on FSSP. Last year we had a cellular meeting at Dagstuhl and there I found some researchers who were interested in Goto's FSSP algorithm. Communicating with Prof. Goto, He indicated just one existing figure showing his algorithm with a very small explanation expressed in 173 Japanese characters and asked me to introduce the algorithm. In this talk we reconstruct Goto's time-optimum FSSP algorithm based on one figure. The talk includes some algorithmic strategies like guessing a riddle.

Cellular Automata for Reaction-Diffusion Systems

Jörg R. Weimar
Institute for Scientific Computing,
Technical University Braunschweig
D–38092 Braunschweig, Germany
Tel. +49-531-191-3006;
E-mail J.Weimar@tu-bs.de

Abstract

Two classes of cellular automata for reaction-diffusion systems are presented. The first type, reactive lattice gas automata, realize diffusion through a random walk of particles on a lattice, while the second class is based on a local average, and closely related to finite difference schemes. The details of the reactive dynamics can either be derived from microscopic considerations (collision and reaction probabilities), or systematically constructed from the desired macroscopic reaction rate laws without regard to an underlying microscopic mechanism. For quantitatively correct modelling, the CA usually needs to be probabilistic, but in some cases a deterministic version proves sufficient.

Reactive Lattice Gas Automata

Reaction-diffusion systems are described by a set of partial differential equations of the form

$$\frac{\partial x_i(\mathbf{r},t)}{\partial t} = D_i \nabla^2 x_i(\mathbf{r},t) + f_i\Big(\{x_j(\mathbf{r},t)\}\Big) \tag{1}$$

where ∇^2 is the spatial Laplacian, and the functions f_i are the (usually nonlinear) reactive terms that depend on several different reactive species x_i.

In real systems, diffusion comes from the overall effect of many molecules executing independent random movements. The first CA method is closely related to the mechanism by which diffusion appears in real systems. We replace molecules by idealized particles and let these particles execute independent random walks on a regular lattice. These automata are usually called reactive lattice gas automata [1, 2, 5, 3].

The cellular automaton evolves on a square lattice on which particles propagate in two dimensions, with nearest neighbor interactions only.

Each particle has associated a velocity, which indicates the direction and distance the particle will move in one time step. In each cell, we allow at most one particle with each of five velocities, so that the state of a cell can be encoded with five bits, each of them indicating the presence of a particle with a particular velocity.

The transition rule is divided into two (later three) parts: Each time step consists of a propagation step (P) followed by a redistribution step (R).

Later we insert a "chemical" reactive step (C) after the propagation step. During the propagation step, the particles (bits representing the state) are copied from the respective neighbors so that each particle is displaced according to its velocity. In the second part, the redistribution step, the elastic collisions in real system are simulated, which lead to a randomization of the particle velocities. In the CA, we simply assign a new random velocity to each particle, taking into account the exclusion principle. This means that on a node with k particles, we have to choose one out of the $\binom{5}{k}$ possible configurations with the same number of particles. There is no interaction with the neighbors at this step, since all interaction takes place through the propagation step. The repeated application of the propagation and the redistribution steps leads to a random walk for each particle, since each particle makes a step of distance one (or zero) in a random direction at each time step.

The reactive part of the cellular automaton with diffusing particles couples the lattices for different species. This step usually needs to be probabilistic. Like the redistribution step, this step is also local to each node. Given that on one node there are α_i particles of species i, we replace these by β_i particles with probability $A(\{\alpha\},\{\beta\})$ (with $\{\alpha\}$ we denote the vector of values α_i for all i). All the details of the reactive dynamics lie in the construction of these probabilities.

One possibility for the construction of the probabilities is to start from a microscopic view: We describe the reactions we want to model as a list of reactive steps

$$\sum_i n_{u,i} X_i \xrightarrow{k_u} \sum_i m_{u,i} X_i. \tag{2}$$

The probability of changing a node with configuration $\{\alpha\}$ to configuration $\{\beta\}$ is set to

$$A(\{\alpha\},\{\beta\}) = V\,\Delta t \sum_u k_u \prod_i \frac{\alpha_i\,!}{(\alpha_i - n_{u,i})\,!}\,\frac{(V-n_{u,i})!}{V!}, \tag{3}$$

where $V = 5$ in our case.

Moving Average Cellular Automata

We now present a second class of cellular automata [4, 5] which models diffusion and reaction from a purely macroscopic point of view. For the diffusive part, we use a modified finite difference scheme, while for the reactive part we use probabilistic rules.

The operation

$$x(r, t+\Delta t) = \frac{1}{2R+1} \sum_{i=-R}^{R} x(r + i\,\Delta r, t) \tag{4}$$

leads to diffusive behavior with $D = \frac{R(R+1)}{6}\frac{\Delta r^2}{\Delta t}$. This local sum operation can be efficiently performed using a moving average technique. The result of this convolution operation applied to $x(\mathbf{r},t)$ is approximately

$$\Phi_D(x(\mathbf{r},t)) = x(\mathbf{r},t) + \tilde{D}\nabla^2 x(\mathbf{r},t). \quad (5)$$

We define the operator for the reactive part as

$$\Phi_R(c_0 x) = x(\mathbf{r},t) + \Delta t f(x(\mathbf{r},t)). \quad (6)$$

As defined, the output of operator Φ_R is not an integer number. We introduce another operator Φ_T for truncation, which takes the real numbers which result from operator Φ_R and produces an integer. The complete dynamics of the CA is given by $\Phi_T \circ \Phi_R \circ \Phi_D$.

For the truncation operator we have many choices, the most useful being

$$\Phi_T(x) = \begin{cases} \lfloor x \rfloor & \text{with probability} \quad 1-p \\ \lfloor x \rfloor + 1 & \text{with probability} \quad p = x - \lfloor x \rfloor. \end{cases} \quad (7)$$

References

[1] Jean Pierre Boon, David Dab, Raymond Kapral, and Anna Lawniczak. Lattice gas automata for reactive systems. Physics Reports, 1995. (to appear).

[2] David Dab, Anna Lawniczak, Jean-Pierre Boon, and Raymond Kapral. Cellular-automaton model for reactive systems. *Phys. Rev. Lett.*, 64(20):2462–2465, 1990.

[3] Jörg R. Weimar, David Dab, Jean-Pierre Boon, and Sauro Succi. Fluctuation correlations in reaction-diffusion systems: Reactive lattice gas automata approach. *Europhysics Letters*, 20(7):627–632, 1 Dec. 1992.

[4] Jörg R. Weimar and Jean-Pierre Boon. Class of cellular automata for reaction-diffusion systems. *Physical Review E*, 49(2):1749–1752, 1994.

[5] Jörg R. Weimar. *Cellular Automata for Reactive Systems.* PhD thesis, Université Libre de Bruxelles, Belgium, 1995.

On parallel Turing machines with multi-head control units

Thomas Worsch*
http://liinwww.ira.uka.de/~worsch/

Parallel Turing machines

Hemmerling (1979) was probably the first to consider a model where several finite automata are working on one common tape cooperatively. Later Wiedermann (1984) generalized the model to so-called *parallel Turing machines* (PTM) which consist of a fixed finite number of tapes with a (possibly varying) number of Turing machine like control units (CU) working on it cooperatively. The CU are identical, i.e. they all have the same number of heads on each tape and work according to the same rule, which includes the ability to delete itself or generate additional CUs.

While one-tape one-head PTM are linear-time and linear-space equivalent to one-dimensional cellular automata and hence in the first machine class $\mathcal{C}_1$ (see van Emde Boas, 1990), it has been argued by Wiedermann (1992) that multi-head PTM are not. It turns out that this essentially depends on the definition of space complexity s for PTM, namely $s = r_t$ where $r_t(n)$ is the number of tape cells used by the PTM for inputs of length n.

We suggest another definition, $s = r_t + \log \binom{r_t^h}{r_h}$ where h is the number heads of the CU and r_h is the maximum number of CU simultaneously working on the tapes (Worsch, 1996). This definition is motivated by the analogous problem for the *storage modification machines* of Schönhage (1980) as discussed e.g. by van Emde Boas (1989).

PTM versus other models

Using the new definition of space complexity for PTM one can describe simulations by CA and vice versa which imply that PTM *are* in $\mathcal{C}_1$.

Wiedermann (1992) observed that it is possible to distinguish different types of models within $\mathcal{C}_1$ and introduced the notion of *weak parallel machines*. These are characterized by their ability to process whole *sequences*

*Lehrstuhl Informatik für Ingenieure und Naturwissenschaftler, Universität Karlsruhe, Am Fasanengarten 5, D-76128 Karlsruhe, Germany.

of inputs in a pipelined fashion. Cellular automata are an example of a weak parallel model while sequential Turing machines probably aren't.

We extend this idea a little bit further by having a look specifically at *"really long" sequences of inputs* (for some precise meaning of "really long"). Surprisingly it turns out that in some sense multi-head PTM probably are even more powerful than cellular automata and not a weak parallel model.

Informally one might say, that multi-head PTM are in the first machine class for short input sequences and in the second machine class for long input sequences.

References

HEMMERLING, A. (1979): Systeme von Turing-Automaten und Zellularräume auf rahmbaren Pseudomustermengen. *Journal of Information Processing and Cybernetics EIK*, **15**, 47–72.

SCHÖNHAGE, A. (1980): Storage modification machines. *SIAM Journal on Computing*, **9**, 490–508.

VAN EMDE BOAS, P. (1989): Space measures for storage modification machines. *Information Processing Letters*, **30**, 103–110.

VAN EMDE BOAS, P. (1990): *Machine Models and Simulations*, vol. A of *Handbook of Theoretical Computer Science*, chap. 1, pp. 1–66. Elsevier Sceince Publishers, Amsterdam.

WIEDERMANN, J. (1984): Parallel Turing machines. Tech. report RUU-CS-84-11, University Utrecht, Utrecht.

WIEDERMANN, J. (1992): Weak parallel machines: a new class of physically feasible parallel machine models. In: I. M. Havel, V. Koubek (eds.), *MFCS '92, 17th International Symposium Mathematical Foundations of Computer Science*, vol. 629 of *LNCS*, pp. 95–111, Springer.

WORSCH, TH. (1996): On parallel Turing machines with multi-head control units. Interner Bericht 11/96, Universität Karlsruhe.

List of Participants

- **Jean-Paul Allouche**
  CNRS, LRI
  Université Paris Sud
  Bâtiment 490
  91405 Orsay Cedex
  France
  Email: allouche@lri.lri.fr

- **André Barbé**
  Department of Electrical Engineering
  Katholieke Universiteit Leuven
  Kard. Mercierlaan, 94
  3001 Leuven
  Belgium
  Email: Andre.Barbe@esat.kuleuven.ac.be

- **Tim Boykett**
  Mathematik
  Universität Linz
  4040 Linz
  Austria
  Email: tim@bruckner.stoch.uni-linz.ac.at
  WWW: http://bruckner.stoch.uni-linz.ac.at/

- **Thomas Buchholz**
  AG Informatik
  Universität Gießen
  Arndtstr. 2
  35392 Gießen
  Germany
  Email: buchholz@informatik.uni-giessen.de
  WWW: http://www.informatik.uni-giessen.de/staff/buchholz.html

- **Gianpiero Cattaneo**
  Dipartimento Scienze dell'Informazione
  Universitá di Milano
  Via Comelico 39
  20135 Milano
  Italy
  Email: cattang@dsi.unimi.it

- **Marianne Delorme**
  LIP ENS-Lyon
  Ecole Normale Supérieure de Lyon
  46 allée d'Italie
  69364 Lyon Cedex 07
  France
  Email: mdelorme@lip.ens-lyon.fr

- **Bruno Durand**
  LIP ENS-Lyon
  Ecole Normale Supérieure de Lyon
  46 allée d'Italie
  69364 Lyon Cedex 07
  France
  Email: bdurand@ens-lyon.fr

- **Jérôme Olivier Durand-Lose**
  LaBRI
  Université Bordeaux I
  351, cours de la Libération
  33 405 TALENCE cedex
  France
  Email: jdurand@labri.u-bordeaux.fr

- **Rudolf Freund**
  Institut für Computersprachen 185/2
  Technische Universität Wien
  Resselg. 3
  A-1040 Wien
  Austria
  Email: rudi@logic.tuwien.ac.at
  WWW: http://logic.tuwien.ac.at/staff/rudi/home.html

- **Max Garzon**
  Department of Mathematical Sciences
  The University of Memphis
  Memphis, TN 38152
  USA
  Email: garzonm@cc.memphis.edu
  WWW: http://www.msci.memphis.edu/~garzonm/cas.html

- **Frédéric Geurts**
  FSA/INGI/INFO
  Université catholique de Louvain
  Place Sainte Barbe 2
  B-1348 Louvain-la-Neuve
  Belgium
  Email: gf@info.ucl.ac.be
  WWW: http://www.info.ucl.ac.be/people/FGeurts/geurts.html

- **Jozef Gruska**
  Institute of Mathematics/Informatics
  Slovak Academy of Sciences
  Dubravská cesta 9
  84000 Bratislava 4
  Slovakia
  Email: gruska@savba.sk

- **Fritz von Haeseler**
  CeVis
  Universität Bremen
  Postfach 330440
  28334 Bremen
  Germany
  Email: fritz@mathematik.uni-Bremen.de

- **Peter Hartmann**
  FG Mikroprogrammierung
  Technische Hochschule Darmstadt
  Alexanderstr. 10
  64283 Darmstadt
  Germany
  Email: hartmann@informatik.th-darmstadt.de

- **Olivier Heen**
  LITP IBP
  Université Paris 7
  2 Place Jussieu
  75251 Paris
  France
  Email: heen@litp.ibp.fr
  WWW: http://www-litp.ibp.fr/~heen/

- **Christian Hochberger**
  FG Mikroprogrammierung
  Technische Hochschule Darmstadt
  Alexanderstr. 10
  64283 Darmstadt
  Germany
  Email: hochberg@informatik.th-darmstadt.de
  WWW: http://www.informatik.th-darmstadt.de/~hochberg

- **Rolf Hoffmann**
  FG Mikroprogrammierung
  Technische Hochschule Darmstadt
  Alexanderstr. 10
  64283 Darmstadt
  Germany
  Email: hoffmann@isa.informatik.th-darmstadt.de

- **Katsunobu Imai**
  Faculty of Engineering
  Hiroshima University
  1-4-1, Kagamiyama
  Higashi-Hiroshima, 739
  Japan
  Email: imai@ke.sys.hiroshima-u.ac.jp
  WWW: http://kepi.ke.sys.hiroshima-u.ac.jp/projects/rca/

- **Ivan Korec**
  Mathematical Institute
  Slovak Academy of Sciences
  Štefánikova 49
  814 73 Bratislava
  Slovakia
  Email: korec@savba.sk

- **Henner Kröger**
  AG Informatik
  Universität Gießen
  Arndtstr. 2
  35392 Gießen
  Germany
  Email: kroeger@informatik.uni-giessen.de

- **Martin Kutrib**
  AG Informatik
  Universität Gießen
  Arndtstr. 2
  35392 Gießen
  Germany
  Email: kutrib@informatik.uni-giessen.de
  WWW: http://www.informatik.uni-giessen.de/staff/kutrib.html

- **Ehler Lange**
  CeVis
  Universität Bremen
  Postfach 330440
  28334 Bremen
  Germany
  Email: lange@mathematik.uni-bremen.de

- **Danuta Makowiec**
  Institute of Theoretical Physics and Astrophysics
  Gdansk University
  ul.Wita Stwosza 57
  80-952 GDANSK
  Poland
  Email: fizdm@paula.univ.gda.pl

- **Maurice Margenstern**
  I.U.T. de Metz
  Département d'Informatique
  Université de Metz
  Ile du Saulcy
  F-57045 METZ Cédex 1
  France
  Email: margens@iut.univ-metz.fr

- **Mario Markus**
  Max-Planck-Institut für molekulare Physiologie
  Rheinlanddamm 201
  44139 Dortmund
  Germany
  Email: markus@mpi-dortmund.mpg.de

- **Bruno Martin**
  Laboratoire I3S - CNRS - URA 1376
  Bâtiment ESSI
  930 Route des Colles
  B.P. 145
  06903 SOPHIA-ANTIPOLIS Cedex
  France
  Email: bmartin@unice.fr
  WWW: http://wwwi3s.unice.fr/∼bmartin/index.html

- **Giancarlo Mauri**
  Dipartimento Scienze dell'Informazione
  Universitá di Milano
  via Comelico 39
  20135 Milano
  Italy
  Email: mauri@hermes.mc.dsi.unimi.it

- **Ken Miura**
  Kobe University
  4 Kitahiyoshi-cho Imakumano
  Higashiyama-ku Kyoto, 605
  Japan
  Email: miuraken@mbox.kyoto-inet.or.jp
  WWW: http://133.30.74.220/miura/index.html

- **Takao Namiki**
  Department of Mathematics
  Hokkaido University
  KITA-KU KITA 10 NISHI 8
  SAPPORO, 060
  Japan
  Email: nami@math.hokudai.ac.jp
  WWW: http://www.math.hokudai.ac.jp/∼nami

- **Antje Petersen**
  CeVis
  Universität Bremen
  Postfach 330440
  28334 Bremen
  Germany
  Email: antje@mathematik.uni-bremen.de

- **Jörg Richstein**
  AG Informatik
  Universität Gießen
  Arndtstr. 2
  35392 Gießen
  Germany
  Email: richstein@informatik.uni-giessen.de
  WWW: http://www.informatik.uni-giessen.de/staff/richstein.html

- **Ralf Schneider**
  FG Mikroprogrammierung
  Technische Hochschule Darmstadt
  Alexanderstr. 10
  64283 Darmstadt
  Germany
  Email: schneider@informatik.th-darmstadt.de

- **Guentcho Skordev**
  CeVis
  Universität Bremen
  Postfach 330440
  28334 Bremen
  Germany
  Email: skordev@mathematik.uni-Bremen.de

- **Hiroshi Umeo**
  Osaka Electro-Communication University
  Neyagawa-shi, Hatsu-cho, 18-8
  Osaka, 572
  Japan
  Email: umeo@umeolab.osakac.ac.jp

- **Roland Vollmar**
  Lehrstuhl Informatik für Ingenieure
  und Naturwissenschaftler
  Universität Karlsruhe
  Am Fasanengarten 5
  76128 Karlsruhe
  Germany
  Email: vollmar@ira.uka.de

- **Jörg Richard Weimar**
  Institut für Wissenschaftliches Rechnen
  Technische Universität Braunschweig
  38092 Braunschweig
  Germany
  Email: j.weimar@tu-bs.de
  WWW: http://www.tu-bs.de/institute/WiR/weimar

- **Thomas Worsch**
  Lehrstuhl Informatik für Ingenieure
  und Naturwissenschaftler
  Universität Karlsruhe
  Am Fasanengarten 5
  76128 Karlsruhe
  Germany
  Email: worsch@ira.uka.de
  WWW: http://liinwww.ira.uka.de/~worsch/

- **Jean-Baptiste Yunès**
  LITP/IBP
  Université Paris 7
  2, place Jussieu
  75251 Paris cedex 05
  France
  Email: Jean-Baptiste.Yunes@litp.ibp.fr