A Vital Two-Dimensional Assembler Automaton

Peter Dittrich, Mathias Wulf, and **Wolfgang Banzhaf**

Informatik XI
University of Dortmund
D-44221 Dortmund, Germany
ls11-www.cs.uni-dortmund.de

Abstract

The aim of this paper is twofold. First it reports on a two-dimensional assembler automaton (called CoreSys) which is able to support digital life without the need of memory protection. Secondly, we would like to discuss two conceptual hypotheses by using CoreSys as a concrete example of digital life: (a) Digital life, or in general strong artificial life, should be studied in its own right. (b) Working with artificial life creates pre-rational (implicit) knowledge about life and life-like processes.

Introduction

This work follows a line of research initiated by Dewdney, 1984 who suggested an assembler automaton as a world where assembler programs can compete. Rasmussen et al., 1990, Ray, 1992, and others have applied this idea to build computer simulations in order to understand life. In these systems the core memory is organized in a linear way and thus organisms consists of a linear sequence of assembler instructions. Avida (Adami and Brown, 1994) and other systems (e.g. (Pargellis, 1996)) introduced a two-dimensional lattice space where lattice sites hold whole (linear) organisms. By introducing the lattice structure these systems allow the study of pattern formation on population level. They also showed a higher diversity.

In our approach the world (core) is also a two-dimensional lattice of cells. But each cell can hold only one instruction. So, organisms are spread over many cells and are usually not linear. Program flow may change its direction in contrast to program flow in automata with linear memory. In other words, the program flow (or the executing CPU) can "go around the corner". This fine grained approach can also be found in (Maley, 1993) and (de Dinechin, 1997b).

The 2-D Assembler Automaton

The important features which distinguish CoreSys from (Ray, 1992) and (Maley, 1993) are described now briefly. For more details and a complete formal specification of CoreSys refer to (Wulf, 1997).

Core: Each memory cell holds one instruction and a relative direction information[1] (left, right, forward). This direction information is used when pointers are incremented (Fig. 1).

CPU and its registers: Each CPU possesses an instruction pointer (IP), a call-stack (CS), and six registers (ax - af). Four of these registers can be pointers and two are pure integer registers. The instruction pointer IP (program counter) and other pointers which refer to memory locations posses two components: As usual the address (in x-y-coordinates) of the memory cell and in addition a direction information (north, south, east, west). The direction is needed for increment and decrement operations. After increment or decrement of a pointer its direction information is replaced by the direction specified by the memory cell just left.

Addressing: As in Tierra relative addressing is performed by pattern matching. Here a pattern consists of a sequence of NOP-x operations *and* a shape (Fig. 1). For a match of two patterns the shapes of the patterns must be identical but the sequence of NOP-x operations has to be complementary (as in Tierra). By this method we can userelatively small patterns which leads to a faster pattern search compared to Maley, 1993 where the shapes have also to be complementary. It should also be noted that in principal loops can be created just by a loop of instructions in the 2-D cell plane. In experiments with organisms using these loops the hand-designed organisms where not able to survive (de Dinechin, 1997a).

Task and memory management: The task management is similar to Tierra's. But there is no memory management required because organisms can overwrite each other. Instead different aging strategies are implemented (see below). A task is terminated ("killed") if an illegal operation is performed.

Properties

The properties of CoreSys are investigated by simulations. All results reported here are obtained by ini-

[1] Instead of adding a direction information to each cell, Maley introduced two operations LEFT and RIGHT which change the direction of the instruction pointer.

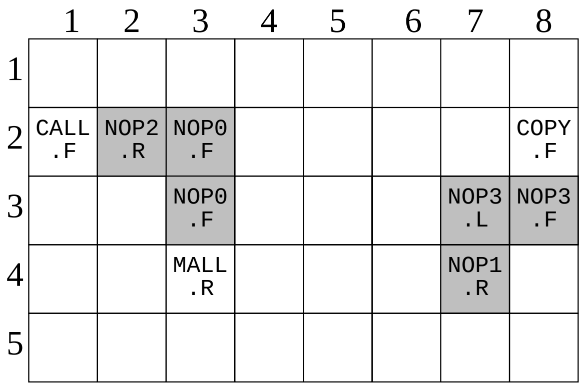

Figure 1: Example of addressing by the key-lock method. Here a key with the sequence 2-0-0 is generated during the CALL operation. A lock with the required shape and sequence 1-3-3 is also present. If the search radius is large enough the lock is found and the IP jumps to the COPY operation. If not, execution continues with MALL.

tializing the core with manually constructed replicators (ancestors). We have also performed experiments with randomly initialized cores. In those experiments self-replicators did not appear, which is consistent with experiments made with linear assembler programs (Rauhe, 1995; Pargellis, 1996).

Reproductive Stability without Passive Mutation

In this section we investigate the reproductive stability of CoreSys . Here we say, a system has reached a **reproductively stable** state, if it is self-maintaining in a stable way. Thus all components are created actively by components of the system.

The experiments with CoreSys show that initializing the system with self-replicators does not necessarily lead to a reproductively stable state. For certain parameter settings the system degenerates until no self-replicators are present in the core. de Dinechin, 1997b observed a similar degeneration in his 2-D assembler automaton (de Dinechin, 1997a). Maley, 1993 introduced a memory management such that the memory occupied by an organism is write-protected against the action of other organisms. Here we show that this protection is not necessary if *aging of organisms* is introduced.

Selection Without Passive Mutation

Before we perform evolutionary experiments with passive mutation we check whether selection takes place and if so, on which factors. Experiments are performed by initializing the core with two different organisms which only differ by one of the following factors: (1) shape ratio (ratio of diameter in x-direction to diameter in y-direction), (2) compactness (fraction of cells occupied in

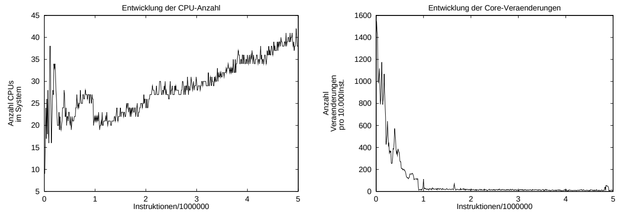

Figure 2: Typical behavior without aging. **Left:** The number of active CPUs increases with time. **Right:** Number of core cells changed during 10,000 executed instructions. This value approaches zero indicating that all replicators die out. Simulation parameters: Initial ancestors: ADAM, core size: 1000 x 1000, simulation time 5×10^6 instructions, search radius: 40 cells, mutation rate: 0, memory allocation radius: 20 cells.

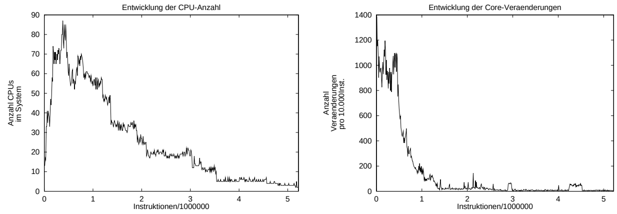

Figure 3: Typical behavior for (fast) cellular aging. The population degenerates similarly to the case without aging. Same parameter setting as in Fig. 2.

the smallest rectangular surrounding the organism) , and (3) size (replication rate).

Figure 5 demonstrates that organisms which posses a shape ratio close to one have selective advantage. The explanation for this is probably the fact that these organisms are overwritten by a lower probability. The same holds for compact organisms which also have a selective advantage against non-compact organisms (Fig. 6). Other simulations also showed that fast replicating organisms have a selective advantage against slow replicating organisms. Note that this is not necessarily the case in general.

Evolution Under Passive Mutation

The experiments described previously have demonstrated that the system is able to enter a reproductively stable state (where genetic information is conserved and transmitted to offsprings in the sens of Eigen and Schuster, 1977) and selection takes place which also depends on shape. Although evolution can take place in principal without passive (external) mutation (Dittrich and Banzhaf, 1998) in experiments with CoreSys without passive mutation evolution of shape has not been observed (Wulf, 1997). Thus in order to stimulate evolution of shape we introduce passive mutation of the shape as an explicit mutation operator. The operators cause

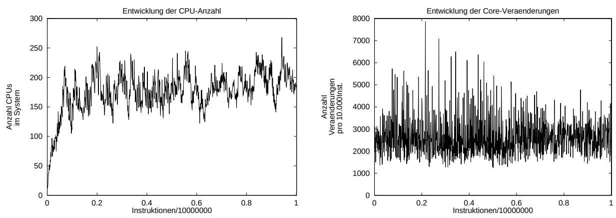

Figure 4: Typical behavior for organism aging. The system enters a reproductively stable state. Although heavy noise-like fluctuations can be observed the average number of active CPUs and the average number of changed core cells converge to positive values. About 28 core cells are changed during 100 instruction executions. Same parameter setting as in Fig. 2 but with organism aging threshold (max. life time): 2500.

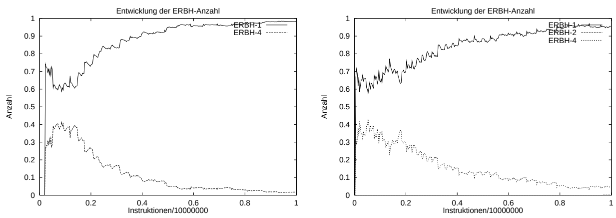

Figure 5: Typical development of a population with organisms of different shape ratio. The ERBH-1-Replicators with a balanced ratio of sides dominate the population. In both simulations an equal but varying number of initial ancestor-organisms were participating (PSR,PSL).

(with low probability during a reproduction process) a change in the target shape of the growing offspring. This is achieved by changing the direction of pointers used to address memory locations.

Figure 7 and Fig. 9 show that not only evolution of shape takes place but also evolution of function (length of the replication cycle). It should be noted that in the presented experiment, first the replication rate surprisingly increases while at the same time the shape changes. Later, the replication rate increases again. A typical evolution sequence of the shape is depicted in Fig. 8.

Discussion

In our opinion the usefulness of (strong) artificial life research has been convincingly motivated by hundreds of papers, as well as for commercial applications as for scientific explanation of biological systems. Here we focus on only two potential motivations, namely: (1) Strong artificial life systems should be studied in their own right. (2) Working with artificial life creates pre-rational (implicit) knowledge about life and life-like processes.

As demonstrated by many authors and also here, digital life can be easily created in different media. Digital

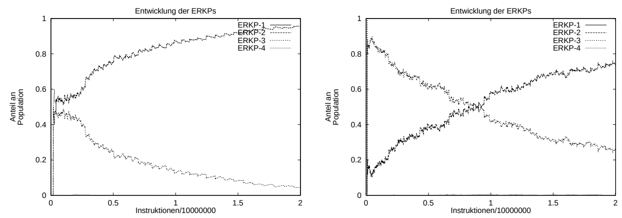

Figure 6: Typical development of a population with organisms of different compactness. **Left:** Simulation initialized with 50 compact and 50 non-compact organisms. **Right:** Simulation initialized with 5 compact and 50 non-compact organisms. The number of compact organisms increases in both cases.

life (or in general artificial life) is present in our environment. It is not only present in computer simulations created by a relatively small number of scientists, but also present in systems of our every day life. Examples are the internet or the law system. E.g., in the internet chain letters are spread and "mutated" by the users either consciously or by clicking on tempting attachments. The law system can be viewed as a dynamic collection of rules where the dynamics of modification is (at least partly) controlled by the system itself. Further more the number of life like systems increases over time. This becomes obvious by looking at the information processing systems surrounding us. Their unavailability[2] (after Schleiermacher) and autonomy increases because more and more automatic tools for their development are applied and learning components are integrated. If (1) is accepted a research aim would be to create AL systems which are "alive" as much as possible in order to reveal their potential properties.

How is the knowledge represented that we gain by investigating AL systems? Beside the (important) communicable knowledge pre-rational (implicit) knowledge about life-like systems is created which can not be communicated. This phenomenon appears also (with increasing intensity) in Biology, Sociology, and Psychology. Whether this phenomenon can be used (e.g. in education) should be left for discussion.

Acknowledgment: Financial support has been provided by the DFG grant Ba 1042/7-1.

References

Adami, C. and Brown, C. T. (1994). Evolutionary learning in the 2D artificial life system avida. In Brooks, R. A. and Maes, P., editors, *Artificial Life IV*, pages 377–381, Cambridge, MA. MIT Press.

de Dinechin, F. (1997a). Personal communication, ECAL'97, Brighton, UK.

de Dinechin, F. (1997b). Self-replication in a 2d von neumann architecture. Technical report, IRISA, campus

[2]Spatial restrictions do not allow an explanation here.

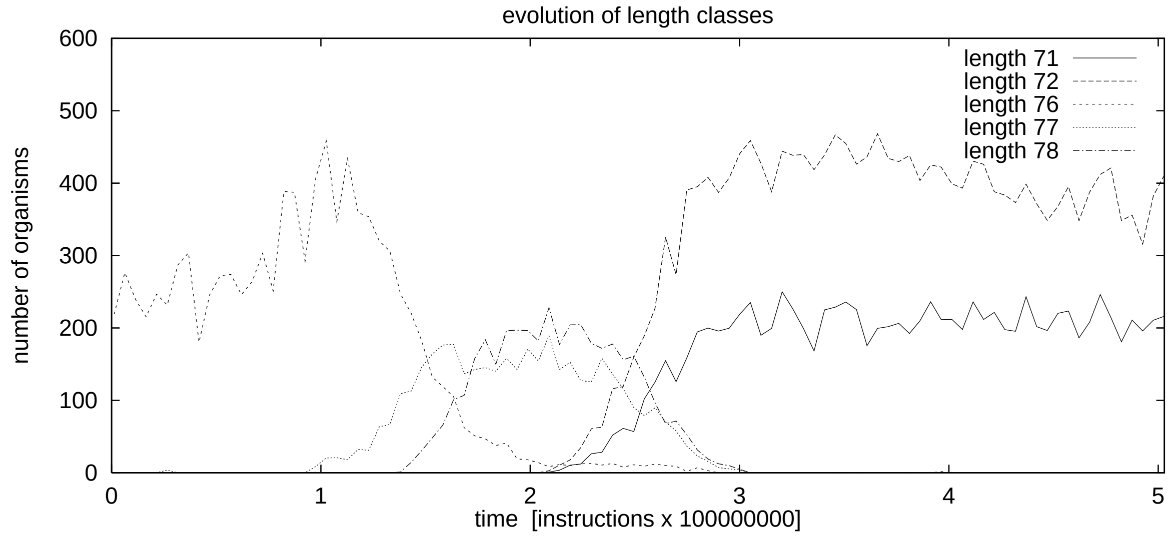

Figure 7: Evolution experiment with passive (explicit) mutation. Concentration of organisms with length 71, 72, 76, 77, and 78 instructions. Note that organisms of length 77 and 78 displace organisms of length 76 (at about $t = 2$) although the shorter organisms of length 76 posses a higher replication rate (shorter replication cycle). Simulation parameters: Initial ancestors: VERYLONG (linear), core size: 2000 x 2000, simulation time 500×10^6 instructions, search radius: 40 cells, shape mutation rate: 0.0001, memory allocation radius: 20 cells.

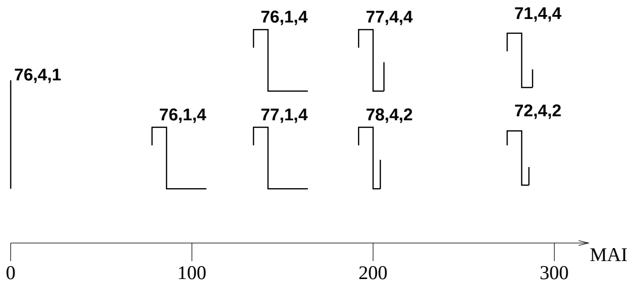

Figure 8: Typical evolution of the dominant shape. In addition to the qualitative program shape the length ERBH and ERKP classes are given.

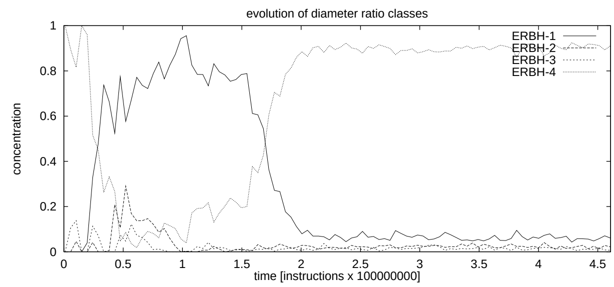

Figure 9: Evolution of ERBH classes that encompass organisms of specific ratios of their diameter. ERBH-1: $[2:2, 2:3[$, ERBH-2: $[2:3, 2:4[$, ERBH-3: $[2:4, 2:5[$, and ERBH-4: $[2:5, 2:\infty[$

de Beaulieu, Rennes, France.

Dewdney, A. K. (1984). In the game called core war hostile programs engage in a battle of bits. *Sci. Amer.*, 250:14–22.

Dittrich, P. and Banzhaf, W. (1998). Self-evolution in a constructive binary string system. *Artificial Life*, 4(2):203–220.

Eigen, M. and Schuster, P. (1977). The hypercycle: a principle of natural self-organisation, part A. *Naturwissenschaften*, 64(11):541–565.

Maley, C. (1993). A model of early evolution in two dimensions. Master's thesis, Oxford University.

Pargellis, A. N. (1996). The spontaneous generation of digital "life". *Physica D*, 91(1-2):86–96.

Rasmussen, S., Knudsen, C., Feldberg, R., and Hindsholm, M. (1990). The coreworld: Emergence and evolution of cooperative structures in a computational chemistry. *Physica D*, 42:111–134.

Rauhe, H. (1995). Investigations of the origin of self-replicating digital organisms. Diploma thesis, Max-Planck-Institut für Züchtungsforschung, Köln. (in german).

Ray, T. S. (1992). An approach to the synthesis of life. In Langton, C. G., Taylor, C., Farmer, J. D., and Rasmussen, S., editors, *Artificial Life II: proceedings of the second artificial life workshop*, pages 371–408, Redwood City, CA. Addison-Wesley.

Wulf, M. (1997). Self-organization of two-dimensional programs. Diploma theses, Dept. of Computer Science, University of Dortmund. (in german).