Emergence of a Super-Turing Computational Potential an Artificial Living Systems

Wiedermann, Jiří
2001

Institute of Computer Science
Academy of Sciences of the Czech Republic

Emergence of a Super-Turing Computational Potential in Artificial Living Systems

Jiří Wiedermann Jan van Leeuwen

Technical report No. 833

February 2001

Pod Vodárenskou věží 2, 182 07 Prague 8 phone: +420 2 688 42 44, fax: +420 2 858 57 89,
e-mail:ics@cs.cas.cz

Institute of Computer Science
Academy of Sciences of the Czech Republic

Emergence of a Super-Turing Computational Potential in Artificial Living Systems[1]

Jiří Wiedermann[2] Jan van Leeuwen[3]

Technical report No. 833

February 2001

Abstract:

The computational potential of artificial living systems can be studied without knowing the algorithms that govern the behavior of such systems. What is needed is a formal model that neither overestimates nor underestimates their true computational power. Our basic model of a single organism will be the so-called cognitive automaton. It may be any device whose computational power is equivalent to a finite state automaton but which may work under a different scenario than standard automata. In the simplest case such a scenario involves a potentially infinite, unpredictable interaction of the model with an active or passive environment to which the model reacts by learning and adjusting its behaviour or even by purposefully modifying the environment in which it operates. One can also model the evolution of the respective systems caused by their architectural changes. An interesting example is offered by communities of cognitive automata. All the respective computational systems show the emergence of a computational power that is not present at the individual level. In all but trivial cases the resulting systems possess a super-Turing computing power. That is, the respective models cannot be simulated by a standard Turing machine and in principle they may solve non-computable tasks. The main tool for deriving the results is non-uniform computational complexity theory.

Keywords:
cognitive automaton, families of automata, communities of automata, Turing machne with advice, interactive computing, non-uniform complexity

---

[1]This research was partially supported by GA ČR grant No. 201/00/1489 and by EC Contract IST-1999-14186 (Project ALCOM-FT).

[2]Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 2, 182 07 Prague 8, Czech Republic, email:jiri.wiedermann@cs.cas.cz

[3]Department of Computer Science, Utrecht University, Padualaan 14, 3584 CH Utrecht, the Netherlands email:jan@cs.uu.nl

1 Introduction

A tantalizing question in computational mind modeling is the following: if it is true that the mind can be modeled by computational means, how can we explain the fact that mathematicians are often able to prove "mechanically unprovable" theorems, i.e. theorems whose truth or falsity cannot be algorithmically proved within a given formal system (e.g. corresponding to a computer simulating the mind) due to Gödel's incompleteness theorem. In an extensive discussion of the respective problems R. Penrose [6] conjectures that there must be some so far unknown faculty of the brain that gives it a non-computable, non-algorithmic, "super-Turing" power in some cases.

In this paper we offer a plausible explanation of this phenomenon in the realm of artificial living systems. We will show that under a certain not-commonly considered computational scenario the ability to surpass the computational limits of a single Turing machine can emerge in non-uniformly evolving families or communities of far simpler computational devices than Turing machines, viz. finite automata.

The plan of the approach is as follows. First, in Section 2 we introduce our basic tool for modeling a single living organism — an interactive cognitive automaton seen as a finite discrete-state computational device. Then, in Section 3 we model the evolution of such devices by means of potentially infinite sequences of cognitive automata of increasing size and show that the resulting "families" possess the super-Turing computing power. For a formal treatment of the respective issues we use basic notions from non-uniform complexity theory. Next, in Section 4 we show that so-called active cognitive automata that can move in an interactive environment and modify it at will, gain the computing power equivalent to that of the standard (or interactive) Turing machine. Finally, in Section 5 we consider evolving communities of communicating active cognitive automata and for such communities we will show the emergence of super-Turing computing power. The merits of the respective results from the viewpoint of computational cognition will be discussed in Section 6.

All the above mentioned results are based on (old and new) results from non-uniform computational complexity theory. The paper opens a new application area for this theory by interpreting its results in terms of cognitive and evolutionary systems. Doing so sheds new light on the computational potential of the respective systems.

Proofs are omitted in this extended abstract.

2 Cognitive Automata

When modeling living organisms in order to study their computing potential it is important to keep in mind that the computational power of a model can be studied without actually knowing the concrete algorithms that are used by the model in concrete situations. What we have to know is the set of the elementary actions of the given model (its "instruction set") and the scenario of its interaction with its environment (what data can appear at its input, whether and how this data depends on previous outputs from the system, whether the system can "off-load" its data to the environment, and so on).

The second fact that we have to take into account is the crucial difference between the requirements put upon a model in case one merely wants to simulate the behaviour or actions of some (living) system, and those in case we also want to investigate its computational potential. In the former case the choice of a more powerful model than is necessary is acceptable since this can simplify the task of simulation. In the latter case the same choice would lead to the overestimation of the computing potential of the system at hand. Thus, in the latter case the model must neither be too powerful nor too weak: it must exactly capture the computing power of the modeled system.

Fortunately, also in such a case we are in a much better situation than it might appear. Despite their unprecedented complexity (when measured in terms of complexity of human artifacts) it is commonly believed that each living organism can enter into only a finite — albeit in most cases astronomic — number of distinguishable internal configurations. From the space limitations put upon this paper we cannot afford to give exhaustive arguments in favor of this fact. Instead, *we take it as our fundamental assumption that a living organism interacting with its environment can in principle be modeled by a finite discrete-state machine.* In the sequel we will call any finite discrete-state machine used in the above mentioned modeling context a *cognitive automaton.* A finite automaton presents a paradigmatic example of such a device. Other examples of cognitive automata are combinatorial circuits [1], discrete neural (cf. [5], [17]) or neuroidal [11] nets, neuromata [7] and various other computational models of the brain (cf. [17], [18]).

The next thing that our model has to capture is the fundamental difference between the standard scenario of computations by a finite automaton or Turing machine, and that of "computations" by a cognitive automaton. In the former case we assume a finite input that is known, fixed prior to the start of the computation. After starting the computation neither additional inputs nor changes in already existing, but not-yet-read inputs are allowed. In the next run of the machine, with new input data, both the finite automaton and the Turing machine must start again from the same initial configuration as in the case of previous inputs. There is no way to transfer information from past runs to the present one. Under this computational scenario, the respective machines are prevented from learning from their past experience.

Contrary to this, living organisms interacting with their environment process their inputs as delivered by their sensory systems without interruption. The inputs "appear", in an on-line manner, unexpectedly and as a rule they must be processed in real-time (this seems to be a necessary condition for the emergence of at least a rudimentary form of consciousness — cf. [20]). In principle the respective computations never terminate and are practically limited only by the lifespan of the organisms at hand. Moreover, the inputs stream into their cognitive systems in parallel via numerous channels and the systems process them also in a parallel manner. In most cases, once read, the original input is no longer available. The number of input channels depends on the size (or complexity) of the system at hand. In addition, especially the inputs into more complex systems that modify their environment or communicate with other systems may depend on the previous actions of a system or the reactions of other systems. Thus the systems gain a potential ability to learn from their own mistakes or experience. The respective computational scenario is a scenario of perpetual interactive learning.

Formally, any cognitive automaton realizes a translation $\Phi_{m,n}$ that transforms infinite input sequences of Boolean m-tuples into similar sequences of n-tuples, for certain $m, n \geq 1$. Depending on the type of formal device used in place of the cognitive automaton, this device can read and produce the tuples in parallel via many input/output ports (as in the case of neural nets), or these tuples are "packed" into symbols of some finite alphabet and are read or produced in this packed form in one step via a single port (as in the case of finite automata, or interactive Turing machines from Section 4).

From [21] and [7] the next theorem follows:

Theorem 1 *Let $\Phi_{m,n} : \{\{0,1\}^m\}^\omega \to \{\{0,1\}^n\}^\omega$, with $m, n \geq 1$, be a translation acting on infinite streams. Then for any $m, n \geq 1$ the following are equivalent:*

- *$\Phi_{m,n}$ is realized by a discrete neural net;*
- *$\Phi_{m,n}$ is realized by a neuroidal net;*
- *$\Phi_{m,n}$ is realized by a neuromaton;*
- *$\Phi_{m,n}$ is realized by a finite (Mealy) automaton.*

Any translation $\Phi_{m,n}$ that is realized by one of the above mentioned devices will be called a *regular translation.*

Although the learning potential of finite automata is not quite obvious, it can be easily observed e.g. in the case of neuroids [11]. Namely, they can be seen as "programmable neurons" since this ability was their primary design goal. The basic set of elementary operations of a cognitive automaton in order to obtain the potential for the development of cognitive abilities via learning is proposed in [11], [18] and [20]. Nevertheless, as mentioned above, the exact form of learning algorithm is unimportant for determining the computational power of the respective devices.

3 Families of Cognitive Automata

In order to be able to process more complicated translations than the regular ones and also to reveal a dependence of computational efficiency on the size of the underlying devices we will consider families of cognitive automata.

Let $\mathbf{F}_p = \{\mathcal{N}_1, \mathcal{N}_2, \ldots \mathcal{N}_i, \ldots \mid size(\mathcal{N}_i) \leq p(i)\}$ be an infinite *family of cognitive automata* of increasing size bounded by function p. Such a family is also called a non-uniform family since in general there need not exist an algorithmic way to compute the description of $\mathcal{N}_i$, given i. Thus there need not be a 'uniform' way to describe the members of the family. Intuitively, the only way to describe the family is to enumerate all its members.

Definition 1 *Let $\mathbf{F}_p$ be a family of cognitive automata, and let $\Phi_{m,n}$ be the translation realized by automaton $\mathcal{N}_m \in \mathbf{F}_p$ for some $n \geq 1$. Then the* non-uniform translation $\Psi(\mathbf{F}_p)$ *realized by $\mathbf{F}_p$ is the set of translations realized by the members of $\mathbf{F}_p$, i.e. $\Phi(\mathbf{F}_p) = \{\Phi_{m,n} | m \geq 1\}$.*

Note that in the above definition, for any given $m \geq 1$, an (infinite) input sequence of m-tuples is processed by the automaton $\mathcal{N}_m$ that is "specialized" in the processing of m-tuples. The class of translations realized by families of neuromata (i.e. of discrete neural nets reading their inputs via a single input port [7]) of polynomially bounded size will be denoted as POLY-NA. In addition to this class we will also consider the classes LOG-NA (the translations realized by neuromata of logarithmic size), POLY-NN (the translations realized by standard recurrent, or cyclic, discrete neural nets of polynomial size, reading their inputs in parallel via m ports), and POLY-FA (the translations realized by finite automata with a polynomial number of states).

Our main tool for characterizing the computational efficiency of the families of cognitive automata will be interactive Turing machines with advice. Non-interactive versions of such machines were introduced by Karp and Lipton [3] who established the foundation of non-uniform complexity theory. Machines with advice are akin to oracle machines as already introduced by Turing [9]. Effectively, an oracle allows inserting outside information into the computation. This information may depend on the concrete input and is given for free to the respective oracle machine whenever the oracle is queried. Advice functions are a special kind of oracle, where the queries can depend only on the size of the input to a machine. Intuitively, the information delivered by an oracle makes sense only for the given input; the information offered by an advice can be used for all inputs of the same size and this is the only oracle information given for all these inputs. With the help of advice a machine may easily gain a super-Turing computing power, because there is no requirement of "computability" on advice (cf. [1], [3], or [13]). Advice is not necessarily more restrictive than the use of an arbitrary oracle, because one can combine all oracle-values ever queried in computations on inputs of size n into one advice value $f(n)$. This is why one usually imposes size-bounds on advice.

Definition 2 *An advice function is a function $f : \mathbf{Z}^+ \to \Sigma^*$. An advice is called $S(n)$-bounded if for all n, the length of $f(n)$ is bounded by $S(n)$.*

Technically, a Turing machine with an advice function f operates on its input of size n in much the same way as a standard Turing machines does. However, such machine can also call its advice by entering into a special query state. After doing so, the value of $f(n)$ will appear at the special read-only advice tape. From this moment onward the machine can also use the contents of this tape in its computation.

It is intriguing to consider the effect of providing cognitive automata with some means to query oracles. It is reasonable to assume that a $p(n)$-size bounded cognitive automaton can only issue oracle queries of size $p(n)$ in its computation on inputs of size n.

Theorem 2 *Let $\mathbf{F}_p$ be a family of cognitive automata that use some Turing machine as oracle. Then $\Psi(\mathbf{F}_p)$ can be realized by a Turing machine using $O(p)$-bounded advice.*

A converse statement also holds but its formulation is beyond the scope of the present paper.

For the classes of translations realized by (interactive) Turing machines with advice we introduce a notation similar to that used in the theory of non-uniform complexity classes (cf. [1]).

Definition 3 *Let* $\mathbf{x} = \{x_t\}_{t\geq 0}$, *let* f *be an advice function and let* $\mathbf{x}\diamond\mathbf{f} = \{\langle x_t, f(t)\rangle\}_{t\geq 0}$, *where the broken brackets denote the concatenation of two strings surrounded by the brackets. Then the class* $\mathcal{C}/\mathcal{F}$ *of translations consists of the translations* Φ *for which there exists a* $\Phi_1 \in \mathcal{C}$ *and a* $f \in \mathcal{F}$ *such that for all* $\mathbf{x} : \Phi(\mathbf{x}) = \Phi_1(\mathbf{x} \diamond \mathbf{f})$.

Thus, a translation Φ belongs to $\mathcal{C}/\mathcal{F}$ iff Φ is realized by a Turing machine from complexity class $\mathcal{C}$ with advice function $f \in \mathcal{F}$. Common choices considered for $\mathcal{C}$ that we will use are: $LOGSPACE$ ('deterministic logarithmic space'), $PSPACE$ ('polynomial space'), etc. Common choices for $\mathcal{F}$ are log, the class of logarithmically bounded advice functions, and $poly$, the class of polynomially bounded advice functions.

In non-uniform computational complexity theory and in the theory of neurocomputing (cf. [5]) the following assertion is proved.

Theorem 3 *For the classes of non-uniform translations the following equalities hold:*

- *POLY-NN=POLY-NA=PSPACE/poly*;
- *LOGSPACE-NN=LOGSPACE-NA=LOGSPACE/log*;
- *POLY-FA=LOGSPACE/poly*.

For more information about the complexity of non-uniform computing we refer to [1].

4 From Cognitive Automata to Cognitive Turing Machines

Consider now a cognitive automaton enhanced by an apparatus that enables it to move around in its living environment and to mark the environment in a way that can later be recognized again by the automaton at hand. The resulting device is called an *active cognitive automaton*. It can store and retrieve information in/from its environment and thus it bears a similarity with robotic cognitive systems. Models of finite automata which can read inputs from a two-way input tape and in addition can also mark input tape cells have been studied for years in automata theory (cf. [16]). The respective machines are provably computationally more powerful than non-marking automata. When we allow a finite set of marks that can be placed to or removed from a potentially infinite environment, one obtains an interactive Turing machine [12], [13], [15]. Interactive Turing machines are a similar extension of standard Turing machines as was the extension of finite automata towards active cognitive automata. Thus, an interactive Turing machine is a Turing machine that translates infinite streams of input symbols into similar streams of output symbols under an interactive scenario as described in Section 2. Several further conditions may be imposed on the way the machine interacts, e.g. to model the bounded delay property that cognitive systems often display in their respond behaviour.

Theorem 4 *The computational power of active cognitive automata is equivalent to the computational power of interactive Turing machines.*

Turing [8] saw "his" machine (i.e., the Turing machine) as a formalized model of a "computer", which in his days meant "a person who calculates". Such a person computes with the help of a finite table (that corresponds to a "program") that is held in a person's head, and further using a (squared) paper, pencil and a rubber. In accordance with Turing's own belief generations of researchers working in artificial intelligence and philosophers of mind have believed that the Turing machine as a whole corresponds to the model of the above mentioned human computer. Our previous short discussion suggests that within the model of a computing person one has to distinguish among three components: the machine's finite control (its "program"), its "sensors, effectors and motoric unit" (movable read/write head) and its environment (the tape). Hodges, Turing's biographer, writes in [2]: *" Turing's model is that of a human mind at work".* This is only partially correct: in Turing's model, merely the machine's finite control corresponds to the mind of the modeled calculating person.

5 Communities of Active Cognitive Automata

Ultimately, active cognitive automata are of interest only in large conglomerates, interacting like "agents" of individually limited powers. A community of active cognitive automata (or shortly: a community of agents) is a time-varying set of devices consisting at each moment of time of a finite set of active cognitive automata of the same type sharing the same environment. Each automaton makes use of a piece of its immediate environment as its private external memory giving it the computing power of an interactive Turing machine (as stated in Theorem 4). Each automaton has its own input and output port. The ports of all automata altogether present the input and output ports of the community. The number of these ports varies along with the cardinality of the community. Within their set the automata are identified by a unique name (or address).

The agents (automata) can communicate by sending their outputs as inputs to other automata identified by their addresses, or by writing down a message into their environment which can be read by other automata. One can see it also as if the agents move in their environment and encounter each other randomly, unpredictably, or intentionally, and exchange messages. Who encounters whom, who will send a message, as well as the delivery time of each message is also unpredictable. The idea is to capture in the model any reasonable message delivery mechanism among agents — be it the Internet, snail mail, spoken language in direct contact, via mobile phones, etc. Moreover, the agents are mortal: they emerge and vanish also unpredictably.

The description of a community of agents is given at each time by the list of names of all living agents at that time, the list of all transient messages at that time, including the respective senders and addressees, the time of the expedition of each message, its addressee, and the message delivery times. Note that in general most of the required parameters needed in the instantaneous description of a community at a given time are non-computable (since according to our description of community functioning they

are unpredictable). Nevertheless, at each moment in time they can be given by a finite table. Therefore the description of the whole community at any time is always finite.

In [13], [15] the following result has been proved for the case of "real" agents communicating via the Internet.

Theorem 5 *The computational power of communities of agents is equivalent to the power of interactive Turing machines with an advice function whose size grows linearly with the processing time.*

Note that similar to the case of infinite families of cognitive automata, communities of active cognitive automata have a super-Turing computing power. The source of this power is given by the potentially unlimited cardinality of the community and by the non-computable characteristics of the community size in the unpredictable interaction among community members and those of their existence span (leading to non-uniformity of the resulting system).

6 Afterthoughts

The results in the previous theorems have interesting interpretations in the world of cognitive automata and computational cognition.

Theorem 1 describes the equivalence among the basic types of cognitive automata. In complexity theory numerous other models of non-uniform computation are known — such as combinatorial or threshold circuits and other types of neural nets, especially the biologically motivated ones (cf. [4]). Nonetheless, the computational equivalence of the respective models indicates that computational cognition is a rather robust phenomenon that can in principle be realized by various computational models which are equivalent to finite automata. We said "in principle" because in practice much will depend upon the efficiency of such models. For instance, in [20] a principle of consciousness emergence in cognitive systems is sketched. In the simplest case consciousness takes the role of a control mechanism that, based on feed-back information from a system's sensors, verifies the correct realization of motoric actions to which orders have been issued. If these actions are not performed in accordance with these orders, the consciousness will realize it and take care about the appropriate remedy. In order to fulfill this role consciousness must operate in real time w.r.t. the speed of the system. The system must react fast enough to be able to recognize the erroneous realization of its orders and take the appropriate measures in time that still give opportunity for the realization of rescue actions. In practice such requirements disqualify "slow" systems and support the specialized, fast or "economical" solutions. It is known that there are cognitive tasks that can be realized by a single biological neuron (over n inputs) whereas the equivalent neural nets requires a quadratic number of standard neurons [4].

Theorem 2 suggests that the evolution of families of cognitive automata that use "recursively enumerable" information from an external source and which might lead to the emergence of a super-Turing computing power, can be simulated using Turing machine models with a very limited, global learning facility.

Theorem 3 illustrates the various degrees of efficiency of certain classes of cognitive automata. For instance, the family of neural nets of polynomial size has the power of PSPACE/*poly* whereas families finite automata of the same size only have the power of LOGSPACE/*poly*. It also demonstrates the emergence of super-Turing power in the course of non-uniform evolution within families.

Theorem 4 points to a jump in the computational power of sufficiently developed active cognitive automata equipped by sensors that can scan the environment and by effectors by which the automata can modify their environment. The corresponding individual active cognitive automata (or cognitive robots) gain the power of interactive Turing machines. In other words, the original finite-state computing device capable of reaching but a bounded number of configurations will turn into a device which can reach a potentially unbounded number of configurations. This is a nice argument that qualitatively illustrates e.g. the revolutionary contribution of the development of the script to the development of human civilization.

The study of the computational power of interactive Turing machines was initiated and studied in [12]. Roughly speaking, the respective theory leads to a generalization of standard computability theory to the case of infinite computations. The results from [12] indicate that by merely adding interactive properties and allowing endless computations, one does not break the Turing computational barrier. The resulting devices are not more powerful than classical Turing machines. They simply compute something different than the latter machines. Thus the new quality is only brought into computing by letting non-predictability enter into the game (cf. [15]).

Theorem 5 asserts that a community of active cognitive automata has a much greater power than the sum of the powers of the individual automata. Here we see the emerging non-recursive computing power of unpredictable external information entering into a system. Do the results of Theorem 5 really mean that the corresponding systems can solve undecidable problems? Well, they can, but only under certain assumptions. In order for these systems to simulate a Turing machine with advice they need a cooperating environment. Its role is to deliver the same information as is offered by the advice. Thus the respective results are of a non-constructive nature: both the advice and the corresponding inputs from the environment exist in principle but there is no algorithmic way to obtain them. In practice the assumptions of the existence of external inputs suitable for the solution of a concrete undecidable problem, such as the halting problem, are nor fulfilled. Hence, without such "right" inputs no community of cognitive automata will solve an undecidable problem. On the other hand, no Turing machine without an advice could simulate e.g. the (existing) Internet — simply because the Internet develops in a completely unpredictable, non-algorithmic way. One can say that the current Internet realizes a concrete non-algorithmic translation that, however, emerges somehow "all by itself", by the joint interplay of all users who operate and upgrade the Internet in a completely unpredictable manner. All users jointly play the role of an "advice" — nonetheless the respective advice keeps emerging on-line, incrementally, is "blind", possessing as a whole no purposeful intention. The same holds for the development of a human society — it also evolves in a non-algorithmic way and therefore cannot be modeled by a single computer (without advice).

What remains to be done is answering Penrose's question from the introduction of

this paper. To see the idea, consider the information computed and stored in a long run by the community of cognitive robots. By virtue of Theorem 5 this is non-computable information. Now, each member of this community has access to this information which effectively plays the role of an advice and thanks to this, in principle each member of the community gains a super-Turing computing power.

7 Conclusion

The previous results can be seen as applications of computability theory to artificial life systems. The main result explaining the emergence of the super-Turing computing potential within the respective systems certainly justifies the approach and points to the increasing role that computer science will play in problems related to understanding the nature of the emergence of life in general and intelligence in particular (cf. [19]). The above results point to quite realistic instances where the classical paradigm of a standard Turing machine as the generic model of all computers which is able to capture all computations, is clearly insufficient. It appears that the time has come to reconsider this paradigm and replace it by its extended version — viz. interactive Turing machines with advice. For a more extended discussion of the related issues, see [13].

Bibliography

[1] Balcázar, J. L. — Díaz, J. — Gabarró, J.: Structural Complexity I. Second Edition, Springer, 1995, 208 p.

[2] Hodges, A.: Turing — A Natural Philosopher. Phoenix, 1997, 58 p.

[3] Karp, R.M. — Lipton, R. J.: Some connections between non-uniform and uniform complexity classes, in *Proc. 12th Annual ACM Symposium on the Theory of Computing* (STOC'80), 1980, pp. 302-309.

[4] Maass, W. — Bishop,C., editors: Pulsed Neural Networks. MIT-Press (Cambridge), 1998.

[5] Orponen, P.: An overview of the computational power of recurrent neural networks. Proc. of the Finnish AI Conference (Espoo, Finland, August 2000), Vol. 3: "AI of Tomorrow", 89-96. Finnish AI Society, Vaasa, 2000.

[6] Penrose, R.: The Emperor's New Mind. Concerning Computers, Mind and the Laws of Physics. Oxford University Press, New York, 1989.

[7] Šíma, J. — Wiedermann, J.: Theory of Neuromata. *Journal of the ACM,* Vol. 45, No. 1, 1998, pp. 155–178.

[8] Turing, A. M.: On computable numbers, with an application to the Entscheidungsproblem, Proc. London Math. Soc., 42-2 (1936) 230-265; A correction, ibid., 43 (1937), pp. 544-546.

[9] Turing, A.M: Systems of logic based on ordinals, Proc. London Math. Soc. Series 2, 45 (1939), pp. 161-228.

[10] Turing, A. M.: Computing machinery and intelligence, Mind 59 (1950) 433-460.

[11] Valiant, L.G.: Circuits of the Mind. Oxford University Press, New York, Oxford, 1994, 237 p., ISBN 0–19–508936–X.

[12] van Leeuwen, J. — Wiedermann, J.: On algorithms and interaction, in: M. Nielsen and B. Rovan (Eds), Mathematical Foundations of Computer Science 2000, 25th Int. Symposium (MFCS'2000), Lecture Notes in Computer Science Vol. 1893, Springer-Verlag, Berlin, 2000, pp. 99-112.

[13] van Leeuwen,J. — Wiedermann, J.: The Turing machine paradigm in contemporary computing, in: B. Enquist and W. Schmidt (Eds), Mathematics Unlimited - 2001 and Beyond, Springer-Verlag, 2001, pp. 1139-1155.

[14] van Leeuwen, J. — Wiedermann, J: A computational Model of Interaction in Embedded Systems. Technical Report TR CS-02-2001, Dept. of Computer Science, University of Urecht, 2001

[15] van Leeuwen, J, — Wiedermann, J.: Breaking the Turing Barrier: the Case of the Internet. Manuscript in preparation, February, 2001.

[16] Wagner,K. — Wechsung, G.: Computational Complexity, VEB Deutscher Verlag der Wissenschaften, Berlin 1986, 551 p.

[17] Wiedermann, J.: Toward Computational Models of the Brain: Getting Started. *Neural Network World,* Vol. 7, No. 1, 1997, pp. 89-120.

[18] Wiedermann, J.: Towards Algorithmic Explanation of Mind Evolution and Functioning (Invited Talk). In: L. Brim, J. Gruska and J. Zlatuška (Eds.), Mathematical Foundations of Computer Science, Proc. of the 23-rd International Symposium (MFCS'98), Lecture Notes in Computer Science Vol. 1450, Springer Verlag, Berlin, 1998, pp. 152–166.

[19] Wiedermann, J.: Simulated Cognition: A Gauntlet Thrown to Computer Science. *ACM Computing Surveys*, Vol. 31, Issue 3es, paper No. 16, 1999.

[20] J. Wiedermann: Intelligence as a Large-Scale Learning Phenomenon. Technical Report ICS AV CR No. 792, 1999, 17 p.

[21] Wiedermann, J.: The computational limits to the cognitive power of neuroidal tabula rasa, in: O. Watanabe and T. Yokomori (Eds.), *Algorithmic Learning Theory*, Proc. 10th International Conference (ALT'99), Lecture Notes in Artific. Intelligence, Vol. 1720, Springer Verlag, Berlin, 1999, pp. 63-76.