

The New Computationalism – a Lesson from Embodied Agents

Jozef Kelemen and Alica Kelemenová

Abstract Computationalism is traditionally considered in the context of cognitive science as perhaps the dominant contemporary approach to understand cognition and cognitive phenomena. It consists in application of concepts and methods of theoretical computer science for understanding and (re)constructing phenomena appearing in much broader fields of science, including the natural sciences and also economics, and some other branches of social sciences. The contribution sketches this new situation, and provides an example of a theoretical model rooted in the traditional computationalism which reflects some new requirements.

1 Introduction

In the context of cognitive science, *computationalism* is traditionally considered as perhaps the dominant contemporary approach to understand cognition and cognitive phenomena. In present days computationalism plays a crucial role not only in cognitive science, but also in the field of cognitive psychology, artificial intelligence and also in an important part of advanced cognitive robotics.

According to [6] (p. 71) the central doctrine of the *traditional computationalism* considered as the basic paradigm for the study of cognition consists in the view that cognition is essentially a matter of the computations that a cognitive system performs in certain situations. In this context, computation is considered as the activity performable by Turing machine, so as computation in the Turing sense generally accepted in the field of theoretical computer science. Computationalists also main-

Jozef Kelemen

Institute of Computer Science, Silesian University, Opava, Czech Republic and College of Management, Bratislava, Slovakia, e-mail: kelemen@fpf.slu.cz

Alica Kelemenová

Institute of Computer Science, Silesian University, Opava, Czech Republic and Department of Informatics, Catholic University, Ružomberok, Slovakia, e-mail: kelemenova@fpf.slu.cz

tain that *neural computations* are Turing-computable, that is, computable by Turing machines, which means that all of the *connectionism* present in cognitive science becomes a part of the traditional computationalism.

Having at hand the collection of notions and results formulated and discovered during the 50 years of the existence of *theoretical computer science* research based crucially on the concept of the Turing machine proposed in [18] we are trying to explain the nature of phenomena of (human) intelligence (usual esp. in artificial intelligence and in advanced robotics) at the level which provides the real base for engineering (re)production of it. The crucial hypothesis accepted almost generally as a base for such considerations is the *Church-Turing hypothesis* which says, roughly speaking, that *a function is computable, in the intuitive sense, if and only if it is Turing-computable* [2, 18].

2 A Slump in the Traditional Computationalism?

However, it is possible to treat computationalism in more general context. In this *broader meaning*, computationalism consists in a conceptual framework originated and formulated in theoretical computer science to understand some aspects of (re)construction (some fragments of) phenomena appearing in broader fields of science as those related to cognition and mind. As parts of these fields we recognize (some branches of) natural sciences, like biology, chemistry, physics, astronomy, also some subfield of economy, some branches of social sciences and arts (e.g. some sub-area of the field of new media, computer art, robotic art, etc.). As the present day state-of-the-art in science and engineering signalizes, both the traditional computationalism and the connectionism have some problems how to react to many new situation appearing in some fields of science and engineering.

Let us emphasize R. Brooks' appeal formulated during his plenary talk for the *8th International Conference on Artificial Life* (Sydney, Australia, December 11, 2002) which focused the attention on a need of a new understanding of computing and computability, in other words to reconsider the actual form of computationalism and push our understanding of computation closer to the present-day requirements. Earlier, in *Nature* (p. 410), R. Brooks wrote:

We have become very good at modeling fluids, materials, planetary dynamics, nuclear explosions and all manner of physical systems. Put some parameters into the program, let it crank, and outcome accurate predictions of the physical character of modeled system. But we are not good at modeling living systems, at small or large scales. Something is wrong. What is wrong?

There are a number of possibilities: (1) we might just be getting a few parameters wrong; (2) we might be building models that are below some complexity threshold; (3) perhaps it is still a lack of computing power; and (4) we might be missing something fundamental and currently unimaginable in our models of biology.

The important and general lesson from the fields like artificial intelligence, advanced robotics, artificial life and cognitive science is, that *the Turing machine uni-*

versality as a mathematical concept which states that all kinds of computers are equally good devices for performing computational tasks, might be misleading in situations, when we consider *machines embedded in their real physical environments*. The fact that an active agent is embedded in its dynamically changing environment may cause two fundamental consequences:

(1) The *input-output relation*, required when we consider processes as Turing machine computations, seems to be an unrealistic requirement, because of the often unpredictable *environment dynamics*, and because of the fact that real agents form at least in some extent *open systems* functioning in this environment.

(2) The *potentially infinite tape* of the Turing machine as a computing device cannot be required as a realistic part of any real physical system.

The matter is discussed in more details in [16] or in [8], where particular examples were presented for embodied agents in connection with some computationally relevant questions.

Moreover, going through Turing's pioneering paper of artificial intelligence [19] we recognize that Turing seems to have no difficulty in accepting the reality of some - let us use the present day trendy terminology - hypercomputational behaviors. His goal, as it is stated in [17], consists in finding some machine that can perform well the *imitational game* (then called *Turing test*). The core of the Turing test lies, roughly speaking, in the conviction that *(human) intelligence can be expressed as a computable function*. If that machine should happen to be a relatively simple one, like the Turing machine, so much better! But if not, then no reason to resign, let's go to construct a much suitable one!

In the core of the traditional computationalism there exists the believe (a) in the power of symbolic representation, and (b) in the approximation of a large number of processes as computational processes, which transform structures created from symbols into the structures of the same type.

For building a concise computational theory the above convictions seem to be realistic. So, their inclusions into different new computational frameworks are required. In the following we will concentrate on a model in which the above sketched limitations (1) and (2) will be suppressed in certain extent, and which will preserve the requirements (a) and (b), but eliminates in certain extent the requirement of the infinity from the basic computational units.

3 An Example on How to Cope with Limits

In [17] an overview of different approaches how to overcome the limitations of the traditional Turing machine is presented. As it is emphasized there (p. 140), any computation in a Turing machine depends on the controlled manipulation of internal configuration, where each configuration encodes a finite amount of information as *state*, a finite amount of information as *memory*, and a finite amount of information as a *program*. The Church-Turing hypothesis tells us that "cosmetic" changes in the architecture of the Turing machine have no relevance to computational power. So, as

Stannett states in [17], in order to go behind the Turing's computability, we can consider changing the information contents of the temporal structure of computation, or the information contents of the memory, programs, or states.

We add the possibility to consider *decentralization*, and to consider the influence of the *contingent*, the *random*, and similar situations, which appear very often in architecture and functioning of real embodied systems (like real computers, robots, as well as biological systems).

In [8] we have illustrated, using an interesting result by D. Wätjen [21] on the generative power of a specific type of decentralized grammar-theoretic model of language generation, on the so called teams [5] in eco-grammar systems [4], that there exist formalized (formal grammar like) systems set up from decentralized components with higher computational power than those of Turing machines. We have also expressed our conviction that there are no principal reasons to reject the hypothesis that it is possible to construct real robots as a certain kind of implementations of these formalized systems. If we include into the functioning of such robots the activation of their functional modules according a non-recursive (in Turing sense) computation, the behavior of the agents might be non-recursive.

We suppose that this situation may appear if some of the functional parts of the robots are switched on or switch off on the base of the random behavior of the robots environments, for instance. So, we exclude the situation when a computer simulation of randomness are included into the functional architecture of robots. Rather, we suppose the randomness appearing in the environment, a randomness which follows from the ontology of robots situated in their environments.

The just mentioned *ontological randomness*, might be caused by different reasons - e.g. by imprecise work of sensors and actuators of robots, by erroneous behavior of their hardwired or software parts, so by the general cause of their embodiment, by nondeterminism of the behavior of the environment, by the lack of resources necessary for executing the required computations, so by their finite nature, and so. All these influences may be reflected in the specific behavior of the robots and we cannot reject the *hypothesis* that just these kinds of irregularities cause also the phenomenon called robot consciousness. It is also possible, that the organic, effective, and enough rigorous inclusion of this type of randomness, caused in fact by the embodiment of computing systems, and by their finiteness, as well, can contribute to our new understanding of computing machineries, and computing as well. If this possibility turns to reality, then we will have at hand the required new model for rigorous formal study and understanding of a new type of computationalism.

In the following we present a formal framework of the eco-colonies [20] as a simple example of formalized systems which respect the finite nature of their basic building components and we prove - following the idea presented in [21] that in the case of inclusion of a formalized variant of the above mentioned ontological randomness into the framework of the eco-colonies we receive a computational power beyond the classical Turing machines.

4 Eco-colonies Working in Time Varying Teams

In present section we propose the formal model to illustrate previous ideas. For this purpose we have chosen the model of cooperating grammars called eco-colonies.

Eco-colonies are collections of simple grammars (called components or agents) working on common environment. A *component* is specified by its start symbol (an object from the environment, the component can process it) and by its finite set of strings, the finite language of the grammar, which determines actions of the component. The component substitutes its start symbol by some of these strings. *The environment* in the eco-colonies are able to develop itself using its own developing rules in totally parallel way like in Lindenmayer systems. All symbols not substituted by components are changed by inner rules of the environment.

Eco-colonies were chosen for following reasons:

(1) *Behavior of an individual component*, an agent, in this model is very simple. Each component acts on the environment common for whole system and it posses with *finite behavior*.

(2) *The behavior of the environment* itself is characterized as classical *OL behavior*, it means the environment has its own rules and all symbols of the environment (on the places not influenced by the agents) are changed simultaneously using these rules. No interaction between neighboring symbols is considered.

(3) The model allows to study team behavior of the agents and it makes possible to consider *function dependent changing of teams*, which can be used to model and discuss the ontological randomness mentioned above.

To present our formal model we assume that the reader is familiar with the formal language theory [15] including the L system theory [7]. The following model is based on the notation of the colony introduced in [9, 10], which was extended to the notion of the *OL* eco-colony in [20]. While in colonies, only agents take a part in development of the environment and environment itself is passive, the environment in an eco-colony posses own developmental rules, too. Eco-colonies may be considered as special type of eco-grammar systems, where each agent can have opportunity to rewrite one of the several possible symbols. For an information on eco-grammar systems consult [4, 5].

We start with formal definition of *OL* eco-colony.

Definition 1. An *OL* eco-colony is a tuple $\Sigma = (V, E, (S_1, F_1), \dots, (S_n, F_n), w_0)$, where

V is a finite non-empty alphabet,

$E = (V, P)$ is an *OL* scheme (of the environment), $P \subset V \times V^*$ is a finite set of rules over V ,

(S_i, F_i) , $1 \leq i \leq n$, is the i -th component, where $S_i \in V$ is the start symbol of the component and $F_i \subseteq (V - \{S_i\})^*$ is a finite set of words (the action language of the component),

$w_0 \in V^*$ is the axiom.

There were various types of derivation modes (sequential, parallel, team, etc.) introduced to describe the behavior of the grammar systems including colonies [12]

and eco-colonies [20]. For our purpose we choose here *the team* behavior for the 0L eco-colony. Team behavior was introduced generally for grammar systems [3] and used for eco-grammar systems in [5] and [21].

By a team we mean here a collection of active components. It will be determined by its size, *the number* of active components, nondeterministically chosen for a derivation step from all components. Teams varying in time can be determined by a function f defined on natural numbers, where the value $f(t)$ determines size of active components in the t -th step of the computation. We speak on the *function depending* team behavior in this case. We formalize the derivation step of an 0L eco-colony:

Definition 2. Let $\Sigma = (V, E, (S_1, F_1), \dots, (S_n, F_n), w_0)$ be an 0L eco-colony.

A $=k$ team derivation step is a binary relation $\xRightarrow{=k}$ on $V^* \times V^*$ defined as follows:

$\alpha \xRightarrow{=k} \beta$ iff $\alpha = \gamma_0 S_{i_1} \gamma_1 S_{i_2} \gamma_2 \dots \gamma_{k-1} S_{i_k} \gamma_k$, and $\beta = \gamma'_0 f_{i_1} \gamma'_1 f_{i_2} \gamma'_2 \dots \gamma'_{k-1} f_{i_k} \gamma'_k$, where

- $f_{i_s} \in F_{i_s}$ for (S_{i_s}, F_{i_s}) , $1 \leq s \leq k$, with $\{i_1, i_2, \dots, i_k\} \subseteq \{1, 2, \dots, n\}$ and $i_s \neq i_m$ for all s, m , $1 \leq s \neq m \leq k$, and
- $\gamma_s \xRightarrow{E} \gamma'_s$, $\gamma_s, \gamma'_s \in V^*$, $0 \leq s \leq k$, is the derivation step of 0L scheme E , i.e.

$\gamma \xRightarrow{E} \gamma'$ iff $\gamma = a_1 \dots a_s$, $\gamma' = \alpha_1 \dots \alpha_s$ and $(a_i \rightarrow \alpha_i) \in P$ for $1 \leq i \leq s$.

Definition 3. A $\leq k$ team derivation step is the relation $\xRightarrow{\leq k}$ defined by

$$\alpha \xRightarrow{\leq k} \beta \text{ iff } \alpha \xRightarrow{=l} \beta \text{ for some } l \leq k.$$

Definition 4. Let $\Sigma = (V, E, (S_1, F_1), \dots, (S_n, F_n), w_0)$ be an 0L eco-colony and let $f : N \rightarrow \{1, \dots, n\}$ be the function defined on the natural numbers.

The language generated by f teams of Σ denoted by $L(\Sigma, f)$, is determined by

$$L(\Sigma, f) = \{w \in V^* \mid w_0 \xRightarrow{=f(1)} w_1 \xRightarrow{=f(2)} w_2 \dots \xRightarrow{=f(r)} w_r = w, r \geq 0\}.$$

The language generated by $\leq f$ teams of Σ denoted $L(\Sigma, \leq f)$, is determined by

$$L(\Sigma, \leq f) = \{w \in V^* \mid w_0 \xRightarrow{\leq f(1)} w_1 \xRightarrow{\leq f(2)} w_2 \dots \xRightarrow{\leq f(r)} w_r = w, r \geq 0\}.$$

We illustrate the behavior of eco-colonies in the following example.

Example 1. Let $\Sigma_n = (V, E, (S_1, F_1), \dots, (S_n, F_n), w_0)$ be an 0L eco-colony, where

$$V = \{a, b, c\},$$

$$E = \{a \rightarrow a^2, b \rightarrow b, c \rightarrow b\},$$

$$(S_i, F_i) = (b, \{c\}) \text{ for } 1 \leq i \leq n \text{ are } n \text{ identical components and}$$

$$w_0 = a^2 b^{2n}.$$

Typical derivation in Σ_n has form

$$w_0 = a^2 b^{2n} \xRightarrow{=f(1)} a^4 u_1 \xRightarrow{=f(2)} a^8 u_2 \xRightarrow{=f(3)} \dots \xRightarrow{=f(t)} a^{2^{t+1}} u_t \xRightarrow{=f(t+1)} \dots$$

with $u_s \in \{b, c\}^{2^n}$ and $|u_s|_c = f(s)$.

The 0L eco-colony Σ_n generates the languages

$$L(\Sigma_n, f) = \{a^2 b^{2n}\} \cup \bigcup_{k \in N} \{a^{2^{k+1}}\} \text{perm}(\underbrace{b, \dots, b}_{2n-f(k)}, \underbrace{c, \dots, c}_{f(k)})$$

and

$$L(\Sigma_n, \leq f) = \{a^2 b^{2n}\} \cup \bigcup_{k \in N, r \leq f(k)} \{a^{2^{k+1}}\} \text{perm}(\underbrace{b, \dots, b}_{2n-r}, \underbrace{c, \dots, c}_r),$$

where by $\text{perm}(a_1, \dots, a_n)$ we mean all the words containing each of the letters a_1, \dots, a_n exactly ones concatenated in an arbitrary order, i.e.

$$\text{perm}(a_1, \dots, a_n) = \{(a_{i_1} \dots a_{i_n}) \mid \text{for all permutations } (i_1, \dots, i_n) \text{ of } (1, \dots, n)\}.$$

Note that the number of occurrences of letter a in w determines uniquely the length of derivations of w in Σ_n . Moreover, we have

$$|w_k|_c = f(k) \text{ for } w_k = a^{2^{k+1}} u_k \in L(\Sigma_n, f) \text{ and}$$

$$|w_k|_c \leq f(k) \text{ for } w_k = a^{2^{k+1}} u_k \in L(\Sigma_n, \leq f).$$

This property will be used in the proofs in next section.

5 Computability of Functions Versus Recursivity of Environments

In the present section we will study the influence of the computability and non computability of function f to the status of recursivity of languages $L(\Sigma, f)$ and $L(\Sigma, \leq f)$, i.e. we are looking to classify $L(\Sigma, f)$ and $L(\Sigma, \leq f)$ to be recursive, recursively enumerable, or not a recursively enumerable set.

We will use notions of computable function, non computable function, recursive set, recursively enumerable set and not recursively enumerable set as in [14].

Function f is *computable* iff following conditions (i) and (ii) are satisfied for some Turing machine M

(i) if $f(x)$ is defined, then M eventually halts in the final states with $f(x)$ symbols of 1 on the tape,

(ii) if $f(x)$ is undefined, then M never reaches the final state.

A set is *recursively enumerable* iff it is the domain of a computable function $f(x)$.

A set is *recursive* iff it equals the domain of a total computable function $f(x)$. Equivalently, L is recursive iff it is decidable whether $w \in L$ for every w .

Ideas presented in this section follow analogous results of Wätjen in [21]. To prove our results we use the languages of the eco-colonies over fixed (three letters) alphabet from the example presented in previous section. This simplify the systems used in [21], where the alphabets of the systems increase linearly with the number of components.

For every eco-colony Σ and for every computable function $f : N \rightarrow \{1, \dots, n\}$ are languages $L(\Sigma, f)$ and $L(\Sigma, \leq f)$ recursively enumerable.

We present a stronger result for the languages from the example above. For this purpose we will use denotation $L(\Sigma_n, f) = L_{n,f}$ and $L(\Sigma_n, \leq f) = L_{n,\leq f}$.

Theorem 1. *Let $f : N \rightarrow \{0, \dots, n\}$ be a function.*

$L_{n,f}$ is recursive if and only if f is computable.

$L_{n,\leq f}$ is recursive if and only if f is computable.

Proof. Let $L_{n,f}$ ($L_{n,\leq f}$) be recursive. We choose an arbitrary $k \in N$ and consider the words $w_r = a^{2^{k+1}} c^r b^{2n-r}$ for all $r \in \{0, \dots, n\}$. Because of the form of the words in $L_{n,f}$ ($L_{n,\leq f}$) presented in the previous section we know that at least one of such w_r are in $L_{n,f}$ ($L_{n,\leq f}$). Since $L_{n,f}$ ($L_{n,\leq f}$) is recursive we can decide whether $w_r \in L_{n,f}$ ($w_r \in L_{n,\leq f}$) for all w_r . We set $f(k) = \max\{r | w_r \in L_{n,f}\}$ (resp. $f(k) = \max\{r | w_r \in L_{n,\leq f}\}$.) Since maximum of finite set is computable f is computable as well.

To prove the other implication assume that f is computable. Let $w \in V^*$.

If $w = a^2 b^{2n}$ then $w \in L_{n,f}$ and $w \in L_{n,\leq f}$. Otherwise it is decidable whether

$$w \in a^{2^{k+1}} \underbrace{\text{perm}(b, \dots, b)}_{2n-r} \underbrace{c, \dots, c}_r$$

for some $k, r \in N$ and $0 \leq r \leq n$. For k determined by w we compute $f(k)$. We have $w \in L_{n,f}$ if and only if $r = f(k)$ and $w \in L_{n,\leq f}$ if and only if $r \leq f(k)$. Thus both $L_{n,f}$ and $L_{n,\leq f}$ are recursive.

A stronger result holds for $L_{n,f}$ but not for $L_{n,\leq f}$.

Theorem 2. *Let $f : N \rightarrow \{0, \dots, n\}$ be a function. If f is not computable then $L_{n,f}$ is not recursively enumerable.*

Proof. Assume contrary that $L_{n,f}$ is recursively enumerable for not computable f . So there exists an effective listing of all words of $L_{n,f}$. We choose an arbitrary $k \in N$. There exists a word $w_k \in L_{n,f}$ with prefix $a^{2^{k+1}} b$ and this word is listed after a finite number of steps. We can compute $f(k) = |w_k|_c$. This gives that f is computable, a contradiction.

6 Some Concluding Remarks and Questions

In the case of the above mentioned model, and more generally, in all cases when formal models are built on the conceptual base of formal grammars and languages, the rules governing the dynamics of the behavior of agent-like entities are described in the form of rewriting rules. This kind of description defines extremely simple, the so called purely reactive, agents. Each purely reactive agent has its own sensor capacity represented by left-hand side of the corresponding rule, and its own action capacity represented by the right-hand side of rules. The ways of interactions of agents are specified by different derivation modes and rewriting regulations in grammar-like structures.

Understanding of rewriting rules as agents is a fundamental advantage at least from the methodological point of view. We know very well, that some specific multi-agent systems (formal grammars) define well-specified behaviors (formal languages) with interesting relation to different models of computation (to different

types of automata) which have important relations to real engineered (computing) machines. What we do not know, is the answer to the question concerning the universality of the approach accepted for describing languages (behaviors). What kind of behaviors are we able to describe using the just sketched framework of agent (and multi-agent system) inspired grammar-like models behind the Turing-computable ones?

The second question follows from an incorporation of dynamics of the environment in which our simple agents act. In the traditional formal language theory we do not consider any inner dynamics of changes of the strings under rewriting. The only changes result from applying rewritings using rules. In the case of eco-grammar systems, however, the situation is substantially modified by providing an "independent" dynamics describing the environment changes using a specific parallel rewriting mechanism (modeled by L-systems) working independently on the activities of agents. What will happen when more complicated mechanisms of changes will be included into the models? What we know about the situation when, for instance, some finite subsequences (belonging to a language with specific Turing-computability properties) will be randomly replaced by words from another set of words (of known Turing-computability property)? Of course, there are more similar questions which can be formulated in a more or less formal ways. We provided some examples, only.

However, the most fundamental question is, according to our meaning, the following one: Is it possible to receive (define) some stabilized (well defined in the framework of Turing-computability, for instance) behavior in the hardly-predictable behavior of the environment in which the agents act? If yes, in which cases, and what are the conditions of such behavior. From the standpoint of the practice, this question is very important, because to design stabilizing multi-agent systems working in the unstable environment is the main goal of many engineering activities. What are we able to say about the possibility of such design in our theoretical framework?

The last question leads us from the speculations about the universality of our models to the question of their realism. Real (embodied or software) multi-agent systems are never perfectly reliable. To be more concrete, let us mention some interesting phenomena related with reliability of (multi-agent) systems. One among the most often appearing is the phenomenon of dysfunction of some agents which are parts of some whole system. Suppose that some of the components of a complicated multi-agent machine go down. Will the whole machine work well (in some acceptable enough way) after this reduction of its components? What kind of changes will appear in its behavior after dysfunction of some of its parts? How to preserve some appropriate level of the functionality of the machine (its resistance with respect of the "small" changes in its architecture)?

An approach to incorporate reliability into the formal models might be inspired by the incorporation of the fuzzy approaches into the traditional grammar-theoretic models. It seems to be possible to use fuzzy rewriting rules, and in the consequence of the derived strings, and to receive formal languages as fuzzy sets, in such a way. It is possible to use fuzzy components of grammar systems, or the regions of membrane systems, and then to propagate the fuzziness toward the generated sets of

behaviors, etc. It is also possible to compare the behavior of such models with the behavior (generative capacity) of the not fuzzified models. How to define the necessary notions?

In other type of systems, despite of the reliability of the agents, their involvement into the work of the whole system is important. In a society of ants, for instance, it is practically impossible to organize the work of any particular agent. Some ants work in some time period, some of them not, and, moreover, we have absolutely no predictive power to know exactly which ant will or will not act in the next time period. This problem is the problem of randomness in multi-agent systems.

How to cope with the randomness of the impact of particular components of multi-grammar models to the derivative capacity of the whole systems, this is illustrated in certain extent by the present article, in which one example of the form of team-forming function is presented and studied. On the base of the presented result, it seems to be realistic to suppose that the relation of other particular forms of team-forming functions and their computational properties considerably influence the behavior of the multi-grammar models. However, exactly what are the ways of this influence, and in what extent, in dependence on the computational properties of the team-forming functions?

Timing is another way of incorporation the dynamics of components into the behavior of models as defined in [11] for colonies, e.g. by functions defined of the length of the derivation chains. Note that the similar approaches are incorporable also into the fuzzy models, so it seems to be realistic also the ability to combine different models of reliabilities and randomness inside one theoretical model. What is the most perspective way of doing that?

Acknowledgments

JK's work on the subject was supported by the grant MSM 4781305903, and by Gratex International Corp., Bratislava.

AK's work on the subject was supported by Scientific Grant Agency of Slovakia, VEGA, project no 1/0692/08.

References

1. R. A. Brooks, The relationship between matter and life, *Nature*. **406** (2001) 409–414.
2. A. Church, An unsolvable problem in elementary number theory, *The American Journal of Mathematics*. **58** (1936) 345–363.
3. E. Csuhaĵ-Varjú, J. Dassow, J. Kelemen and Gh. Păun, *Grammar Systems – A Grammatical Approach to Distribution and Cooperation* (Gordon and Breach, London, 1994).
4. E. Csuhaĵ-Varjú, J. Kelemen, A. Kelemenová and Gh. Păun, Eco-grammar Systems - a grammatical framework for studying life-like interactions, *Artificial Life*. **3** (1997) 1–28.
5. E. Csuhaĵ-Varjú and A. Kelemenová, Team behaviour in eco-grammar systems, *Theoretical computer science*. **209** (1998) 213–224.

6. M. Giunti, Beyond computationalism, in *Proc. 18th Annual Conference of the Cognitive Science Society*, ed. G. W. Cottler (Lawrence Erlbaum, Mahwah, NJ, 1996), pp. 171–75.
7. G.T. Herman, G. Rozenberg, *Developmental systems and languages* (North-Holland, Amsterdam/American Elsevier, New York, 1975)
8. J. Kelemen, May embodiment cause hyper-computation? in *Advances in Artificial Life, Proc. ECAL '05*, eds. M. S. Capcarrere et al. (Springer Verlag, Berlin, 2005), pp. 31–36.
9. J. Kelemen and A. Kelemenová, A subsumption architecture for generative symbol systems, in *Cybernetics and System Research '92*, ed. R. Trappl (World Scientific, Singapore, 1992), pp. 1529–1536.
10. J. Kelemen and A. Kelemenová, A grammar-theoretic treatment of multiagent systems, *Cybernetics and Systems*. **23** (1992) 621–633.
11. A. Kelemenová, Timing in colonies, in *Grammatical models of multi-agent systems*, eds. G. Păun and A. Salomaa (Gordon and Breach, London, 1999), pp. 136–143
12. A. Kelemenová, Bounded life resources in colonies, in *XIV Tarragona Seminar on Formal Syntax and Semantics, Report 23/02 Research group on mathematical linguistics*, (Universitat Rovira i Virgili, 2002) 24 pp.
13. A. Kelemenová and E. Csuhaaj-Varjú, Languages of colonies, *Theoretical Computer Science*. **134** (1994) 119–130.
14. G. Rozenberg and A. Salomaa, *Cornestones of undecidability* (Prentice Hall, New York, 1994)
15. G. Rozenberg and A. Salomaa, (eds.), *The Handbook of Formal Languages*. 3 volumes, (Springer-Verlag, Berlin, 1996).
16. A. Sloman, The irrelevance of Turing machines to AI, in *Computationalism - New Directions*, ed. M. Scheutz (The MIT Press, Cambridge, Mass., 2002), pp. 87–127.
17. M. Stannett, Hypercomputational models, in *Alan Turing - Life and Legacy of a Great Thinker*, ed. C. Teuscher (Springer Verlag, Berlin, 2004) pp. 135-157.
18. A. M. Turing, On computable numbers, with an application to the entscheidungsproblem, in *Proc. London Mathematical Society* **42** (1936) 230–265; corrections in **43** (1937) 544–546.
19. A. M. Turing, Computing machinery and intelligence, *Mind*. **59** (1950) 433–460.
20. Š. Vavrečková, Properties of eco-colonies, in *Information Systems and Formal Models 07*, eds. A. Kelemenová et al. (Silesian University, Opava, 2007), pp. 235–242.
21. D. Wätjen, Function dependent teams in eco-grammar systems, *Theoretical Computer Science*. **306** (2003) 39–53.