

LOOKING FOR INTELLIGENT RECONFIGURABLE SIMULATION

Tudor Niculiu^B, Anton Manolescu^B and Sorin Cotofana^D
Universitatea POLITEHNICA din Bucuresti, ETI Dept.
Splaiul Independentei 313, 060042 Bucuresti, România
tudor-razvan@ieee.org
Delft University of Technology, EE Dept.
Mekelweg 42600, GA Delft, The Netherlands
S.D.Cotofana@ewi.tudelft.nl


KEYWORDS

Conscience, Hierarchy Types, Simulated Intelligence.

ABSTRACT

Simulation relates function to structure. Researching intelligence by simulating it to get to intelligent simulation demands the study of essential abstract structures: the human mind, the different types of hierarchies, and the simulation as relation between static and dynamic structures. Intelligence and Faith, as any other dichotomies, can converge together to integration, or can destroy each other if they are not associated. Conscience is the link. Both intelligent simulation and the simulation of intelligence demand transcending the present limits of computability toward simulability by an intensive effort on extensive research to integrate essential mathematical and physical knowledge guided by philosophical goals. We extend the reconfigurability to the simulation itself. First, by a self-aware simulation, for that we build a knowledge hierarchy corresponding to the simulation hierarchy, we get a self-control of the simulation process. Then, by expressing both simulation and knowledge hierarchies in the reference system of the basic hierarchy types (classes, symbols, modules), which is derived from the main partition of our Real Life (Beauty, Truth, Good), we aim to create the context for a self-organization of the simulation. So we try to model the conscience for simulating the intelligence, and then to reach for intelligent simulation.

ARGUMENT

Faith-Intelligence are  in our life (Way, Truth, Life).

Intelligence = (consciousness, adaptability, intention) and *Faith* = (inspiration, intuition, imagination) are complementary parts of the human mind, separated by *Conscience* = (consciousness, inspiration).

Reality is not confined to *Nature*, as the cardinal of the discrete IN is less than the one of the continuous IR, but *Reason*, which is natural, as $|IN| = |Q|$, is dense in the Reality, as IR is the analytical closure of the (discrete) rationales. This shows that neither pure reason-based adaptability nor pure intuition can approach Reality without being integrated by conscience and communicating by intention and imagination.

The natural limit of complexity is caused by the essentially sequential approach, whereby the real limit of computability results from the discreteness of our reason, when considered context-free in our mind. The first idea is to consider/ remember that reality is more than nature, as the continuum of IR is more powerful than the discrete universe of IN. The second analogy is that integer beauty is not enough to comprehend the Reality. The third argument is that reason is less than our real thoughts, as the cardinal of Q is \aleph_0 .

Although Q is dense in R , so pure reason could converge to reality, the complexity problem limits the computability. The essential limit of the discrete computability, as of the computable intelligence, results from the self-reference, demanded by the integration of level and metalevel needed for consciousness. A hierarchical type is necessary to represent conscious knowledge.

Even if for the moment other essential aspects of the faithful intelligence can neither be constructive or intuitive, they should not be neglected. For example, there are much more real things than those reasonably imagined, although between any two real numbers there is a rational one (not intuitive). And we know that if there is no cardinal between that of the countable sets and that of the continuous ones, then there exists no other logical value than true and false, what simply hurts the human in his love for nuances. This can be avoided only if we believe (not constructive) that an intermediary level between Natural Reason and Reality exists, as the wises think (Plesu 2003):

There are angels between humans and God.

The historical experiment of the pure reason had sense in our evolution, but should have ended two centuries ago, when geniuses, as Beethoven, Byron, Hegel, Gauß, Goethe, Kant, Mozart, Napoleon, Schiller, more synchronized than ever, proved the power of integrating

- faith with intelligence by conscience,
- art, science and construction by metaphysics,
- human and humanity, by their individual work that no one thinks to surpass alone.

Human thoughts can not be explained or handled by adaptability-based reason, even if this masters non-determinism or parallelism.

Reason has to extend to intelligence in the context of faith. An intuitive way is to integrate consciousness, then intention and imagination to intelligence, and then to extend the research towards inspiration and intuition. The power of abstraction is the real measure for the human mind. Turning abstraction into comprehensive construction could be the aim of humanity, the unique *God* for different cultures of free humans.

Freedom is understood necessity

Georg Friedrich Wilhelm Hegel reminded us of the real sense of freedom, so we have to recall our conscience to reintegrate our mind and to remember that society has to assist humans to live among humans, not to consider that humans just have to work for the society. An operating system serves the autonomous programs, both for the function of the hardware and for development of the software. The society has to assure health and education for everyone, and encourage search and research for any conscious human. Society is only the memory of the past, and the manager of the present problems to live together in respect of the others on the way to understand each other, building us together to more essential beings for an integrated existence.

RECONFIGURATION

Applying *Divide et Impera et Intellige* to hierarchy types reveals their comprehensive constructive importance based on structural approach, symbolic meaning, object-oriented representation. Formal hierarchical descriptions contribute to a theoretical kernel for self-organizing systems. A way to begin is hierarchical simulation. A way to confirm is the object-oriented reconfigurable simulation.

Reconfigurable computing architectures complement existing alternatives of spatial custom hardware and temporal processors, combining increased performance and density over processors, with flexibility in application. The experimented ways to reconfigurable design are Field-Programmable Gate Arrays for circuits (Rabaey 1997) and reconfigurable networks for systems (Miller et al. 1993).

Our project (Niculiu and Cotofana 2003) extends the reconfigurability to the simulation itself. Intelligent self-organization needs consciousness to control adaptability for reconfiguration. Towards a self-aware simulation to control the simulation process we build a knowledge hierarchy corresponding to the simulation hierarchy, and by expressing both simulation and knowledge hierarchies in the reference system of the basic hierarchy types we create the context for a self-organization of the simulation. We try to reach this goal integrating hierarchical intelligent simulation into nanotechnological realization (Goldstein and Budiu 2001)

Recursive reconfiguration of the simulation process, at any hierarchy level, is allowed by different strategies that alter one of the technique/ model/ method if one of the imposed

properties is not fulfilled after applying a technique, using a model and suitable methods for evaluation and reconfiguration. The process repeats for the initial description or the one resulted from prior (insufficient) improvement. This calls for an intelligent control system that assists/ automates the reconfiguration. The techniques use hardware-software model templates, whose methods are recursively handling the different components in the system's description. Measurement functions control the continuation process of the reconfiguration, what suggested bringing reconfiguration in the context of software and hardware, as the strategies can be expressed object-oriented/ categorical and developed/ understood mathematically.

HIERARCHICAL SIMULATION

The research on cosimulation inspires the study of essential abstract structures: human mind, different hierarchy types, and simulation - as relation between static and dynamic structures, or even, at a higher abstraction level, between structure and function (Niculiu and Cotofana 2004). Towards this goal we put in correspondence three triplets of concepts of different collaborating domains: *hierarchy types* (class, symbol, structure), simulation abstractions (syntax, semantics, pragmatics), basic philosophy: (Beauty, Truth, Good). More points of view can confirm a selection of the essential things to begin marching on the way. Further necessary correspondences to approach and goal partitions of the human evolution are explained.

Simulation = (representation, goal).

Representation is a 1-to-1 mapping from the universe of systems (objects of simulation) to a hierarchical universe of models (a representation can be inverted). A model must permit knowledge and manipulation, so it has two complementary parts/ views: description and operation. In a formal approach models correspond to classes and specifications to instances.

Understanding and construction should use correspondent hierarchy types, i.e., a reflexive kind of abstraction has to be expressed by the *knowledge hierarchy type*. Knowledge and construction hierarchies cooperate to integrate design and verification into simulation; object-oriented concepts are symbolized to handle data and operations formally; structural representation of behavior manages its realization. A hierarchical approach is needed to handle both knowledge and metaknowledge. Hierarchy types open the way to simulate intelligence as adaptable consciousness by integrating the system and the metasystem. Hierarchy is the syntax of abstraction.

Hierarchy is a network that can represent any mathematical structure type (algebraic, topological, order). Hierarchies are leveled structures, which represent different domains. A level is an autonomous mathematical structure, containing abstract/ concrete entities, linked by level scoped relations.

Abstraction relates the levels: this induces an order relation between levels, partial, concerning entities, and total, regarding the levels. Beyond the hierarchical point of view, the system can be formalized as an autonomous domain, structured by metahierarchical relations, to build a level in a higher order hierarchical system. Hierarchical structures exhibit two complementary processing strategies: top-down and bottom-up. Coexistent interdependent hierarchies structure the universe of models for complex systems, e.g., hardware/ software ones. They belong to different hierarchy types, defined by simulation abstraction levels, autonomous modules, classification, symbolization and different grades of knowledge abstraction.

Abstraction and hierarchy are semantic and syntactical aspects of a unique fundamental concept, the most powerful tool in systematic knowledge. The (hierarchy, abstraction) concept is a particular form of Divide et Impera et Intellige: hierarchy results of formalizing abstraction. There are different kinds of abstraction that need different types of hierarchy. Most abstractions are simplifying, what is compulsory for complex object-systems. Correspondent to the construction/ simulation hierarchy type, a knowledge hierarchy type has to formalize reflexive abstraction in order to enable intelligent simulation.

Classes abstract the form, symbols the contents, and partitions simplify the approach. All these assist the simulation hierarchy to construct, verify, optimize and test, being managed completely by discrete formalisms/ simulations of the pure reason. The knowledge hierarchies promise to integrate discrete into continuous, on the way to extend pure reason to less constrained thinking. Hierarchies of different types correspond to the kind of abstraction they reflect (\uparrow the abstraction goal):

- Class hierarchy (\uparrow concepts) \leftrightarrow virtual framework to represent any kind of hierarchy, based on form-contents, modularity, inheritance, polymorphism.
- Symbol hierarchy (\uparrow metaphors) \leftrightarrow stepwise formalism for all kind of (hierarchy) types.
- Structure hierarchy (\uparrow strategies) \leftrightarrow stepwise managing of all (hierarchy) types on different levels by recursive autonomous block decomposition.
- Construction hierarchy (\uparrow simulation) \leftrightarrow simulation (design/ verification/ optimization/ testing) framework of autonomous levels for different abstraction grades of description.
- Knowledge hierarchy (\uparrow theories) \leftrightarrow reflexive abstraction, aiming that each level has knowledge of its inferior levels, including itself. This hierarchy type offers a way to model conscience.

Metaphor is a popular instance of abstraction. God is the absolute abstraction. And if we remember that liberty is understood necessity we can detail the metaphorical thesis:

God is the evolution goal of our faithful intelligence.

We can reduce abstraction to its simplifying types: classes, symbols, modules, and construction, hoping to get to the absolute liberty, i.e., considering God, the simplest item of the Reality, totally unconstrained. But we can simulate/ construct/ live/ work associating a knowledge hierarchy to everything we do, aiming to understand constructively the *most complex absolute necessity, defining God*. The power of abstraction is human's gift to surpass the natural limits, extending pure reason to real intelligence. As any other dichotomy pair, faith and intelligence can evolve convergent to integration, or can destroy one another if they are not linked together constructively.

Divide et Impera et Intellige has three parts as *alle guten Dinge sind drei*. Mathematics develops from three basic structure types, usually integrating them: algebra, order, and topology. We divided our existence in three collaborating parts: arts, sciences, and technology, correspondent to our world of beauty-loving ideas, our world of truth-searching efforts, and our (presently exaggerated) world of good-aiming constructions.

1. Mathematics - the most accessible art (Hofstadter 1979) - discovers and studies (Bourbaki 1966) types of structures: (algebra, topology, order), correspondent to (construction, orientation, understanding) as example of correct and complete integration, to be followed by science and technology. *Art is for art*, so it's defining itself, looking for the Beauty.
2. Physics - the paradigmatic science - should integrate its fundamental forces in a theory (Traub 1999), and all natural and social sciences - as chapters, leading them to really apply mathematics. Social sciences study a universe, as complex and non-deterministic as the natural one, so mathematics is at least as important to them as for natural sciences. An integrated science would also better inspire mathematics. Science raises the fear and the research inspired by it to more abstract domains, i.e., it is defined, as *Fear of God*, looking for the Truth.
3. Engineering has to be closely related to mathematical approach and integration of parts, not only to mathematical techniques, as to scientific courage and multiple views, not only to scientific results (Ciaffaglione and diGianantonio 2001). As reality contains the abstract ideas, even if physics could explain everything discretely, the power of continuum can not be forgotten; analog engineering should not be neglected in modeling and simulation. Paying attention only to the Good in our life, is most dangerous, as this part of the Reality, called *mental world* (Penrose 1989), is defined by its complement, so it is not better than this, if not closely constrained by Art and Science.

The classical activities in complex systems simulation that regard different levels of the construction or knowledge hierarchy, can be expressed symbolically, represented object-oriented and simulated structurally.

Constructive type theory permits formal specification and simulation, generating an object satisfying the specification. Complex simulation needs consistent combination of mathematical domains and an intelligent compromise between consistence and completeness. Intelligence simulation implies a hierarchical approach of different types. Any application of it can be imagined as an educational system to discover models for conscience and understanding. The formalism for hierarchy types is the category theory (Ageron 2001). Different domains permit a unified formalization in the theory of categories, and a unified representation using object-oriented templates. Simulation should remain correct, with extended requirements for the object-system, regarding complexity, optimization and (sequential/ parallel) competence for different domains. The hierarchical principle, applied to knowledge and simulation, (locally) bounds the complexity, by problem decomposition, and assures (almost) correct-by-construction design and efficient (design-adapted) verification.

HIERARCHICAL COSIMULATION

Cosimulation of coexistent domains is an important step for collaborative specialization, the next step to Intelligé after Divide et Impera and an essential need before approaching conscience modeling. Testability is the technological correspondent of sincerity, which is essential for intelligence and communication.

Hardware-Software Cosimulation

The hardware-software cosimulation of complex systems is imposed by the lack of compatibility or optimality associated with the initial hardware/ software partition of a design, and by the inefficiency of the design-verification cycle in the context of a fixed partition (Niculiu et al. 2002). To unify simulation methodologies, we started from the results of different research directions: object-oriented hardware/ software description, formal verification of software/ hardware, automated synthesis of hardware systems. A unified representation for hardware and software allows techniques from one domain to be applied to the other domain. Therefore, a representation based on abstraction and object-orientation, used primarily for software, is employed for the hardware domain as well. Also, existing software techniques, such as those used for verification of abstract data type implementations, can be used for hardware.

Knowing the features (mandatory: abstraction, hierarchy, encapsulation, modularity, message passing + optionally: typing, concurrence, persistence) that characterize an object-oriented language, they also make sense from the perspective of hardware modeling and simulation. Object-oriented specification of models can be based on general systems theory, what makes this approach applicable in all domains. The designed framework permits self-organizing. It offers at any abstraction level of the simulation hierarchy: system description in a commonly used language extended

for parallelism by synchronization items; automatic learning-based hardware/ software partition of the description; consistent communication between heterogeneous parts and with the exterior; simulation of the whole system during any design phase. Data abstraction can be used to represent hardware. A class corresponds to a set of elements with common static and dynamic characteristics. Thus, a hardware component can be treated as class containing state along with a collection of associated operations that can manipulate this state. For example, a register can be viewed as a class with the operations read and write. The contents of the register correspond to its state, which can be accessed and manipulated using the operations read and write, respectively.

Software engineering utilizes data decomposition to refine (derive implementations for) abstract data types. When modeled as data abstractions, hardware elements can also be refined using this decomposition technique. Generic types result from the ability to parameterize with types a software element, such as procedure or data type. This makes programs more general. The template concept, that realizes it in C++, can be applied to hardware components that act as containers, e.g., registers, register files.

Digital-Analog Cosimulation

The essential difference between analog and digital simulation paradigm is induced by that between the mathematical structures their models are based on algebraic for digital versus analytical for analog. In view of intelligent simulation the whole intelligence has to be simulated, i.e., conscience along with adaptability. The discrete parts of simulation, e.g., a sequence of decisions/ stimuli for simulation, do not easily match the continuity of the analog part.

Usually, the difficulty of analog simulation is avoided by defining an auxiliary representation domain, intermediate between the behavioral and the structural, where the problem is decomposed into topology selection and dimension computation. The first process is discrete and the second one is continuous over a restricted problem space. Object-oriented representation lends itself for this complementary form-content instance.

But, topology selection would be more systematic if continuous modifications of the form were possible, and dimension computing is more efficient if symbolic algebraic methods are used. We searched the compromise between simulation algebra and analog analysis in three directions: defining upper levels of abstraction for the algebraic laws governed analog, modeling analog simulation in algebraic-analytical structures (of functional analysis) or association of analytical syntax to the analog simulation process (Zhong and Weihrauch 2003). All these approaches are suited to object-oriented *Analog Hardware Description Language* (AHDL).

Thermal-Analog Simulation

The development of Computer Aided Design (CAD) procedures for microsystems imposes the simulation of thermal phenomena as secondary effects to the main, analog - electronic, mechanical, optical, or chemical - ones. As the microsystem components are modeled in an AHDL the models can be enhanced with temperature dependence and power generation estimation. Moreover models for environment and packaging conditions can be added as well. AHDL models permit direct simulation of the microscopic thermal transfer, and qualitative simulation-oriented representation of second order effects. Consequently, different physical domains, described by isomorphic analog laws, can be simulated in a unique representation (Zeigler et al. 2000).

Electro-, hydro-, thermodynamics, or circuit theory can be expressed with through-across concepts governed by dual topological laws for continuity and compatibility. AHDL enables a direct physical simulation of heat conduction, alternative to discrete heat equation: if only the first order relation representing Fourier's hypothesis is expressed in the model, the integration and discretization are realized by topological constraints that characterize AHDL structures. This suggests the idea:

Simulation is computer-oriented theory.

Behavioral Adaptable Design for Testability

Design-for-testability (DFT) must suit the behavioral specification of today's complex system design. Referring to high-level synthesis, DFT can operate before, while or after it. The first choice permits the intervention of an intelligent agent for adapting the DFT technique, model or method to the particular design. We call it behavioral adaptable design-for-testability: it improves the testability, measured with adequate methods, direct on the behavioral specification or aided by special representations, that have to permit returning to the behavioral description after improving the testability of the system to be designed.

The results are general enough to be valid for systems, either hardware, software or hard/ soft. The most used DFT techniques are Scanning, Built-In Self-Test and Test Point Insertion. They can be applied at the different levels of the design hierarchy (behavior, Register Transfer Level - RTL, logic) and can be combined.

We began with Partial Scan applied to the autonomous blocks of the behavioral HDL specification, but the other techniques can contribute to improve the testability of the behavioral specification or the way to this goal. The Partial Scan problem is the selection of the scan registers following a strategy to find an optimum testability - complexity compromise; this is better shaded by analog computing (Blum et al. 1998).

Memory elements - registers (arrayed flip-flops)/ flip-flops/ latches (non-clocked flip-flops) - are represented in behavioral hardware descriptions by variables or signals.

Variables are description objects local to processes/ subprograms, used to store intermediate values between sequential statements, characterized by free assignment.

Signals are permanent description objects to link concurrent elements: components/ processes/ concurrent assignments, demanding synchronized assignment, declared locally - within architecture, block or other declarative region, or globally - in extended package. In the context of a process that is synchronized by a clock signal, in a behavioral description, signals implicated in simple/ multiple signal assignment generate memory during synthesis.

An analogous rule can be formulated for variables: Inside a process, a variable that must hold values between iterations of the process implies memory elements. A variable that is set but not used between synchronization statements infers memory; a variable that is read before being assigned also infers memory.

The context is not restrictive, as all concurrent statements are equivalent to processes (excepting direct instantiation). For called subprograms the rules of memory inference can be deduced directly: pure functions (no side effects) do infer memory - while procedures (side effects) do not. All types of hierarchies are implied in this approach: design abstraction levels, block structure, class framework, symbolization and knowledge hierarchies.

We combined Partial Scan methods to optimize the order to add memory elements to the scan chain, at behavioral level. An adaptable interface assures the translation, in both senses, from behavioral hard/ soft description to a structural representation of the required behavior. The partial-scan selection uses a knowledge base to generate the weighted directed graph (flip-flops, combinational paths) and to return to text the differences caused by transformation for testability improvement.

The rules of correspondence between description object (signal/ variable) assignments and registers, and those to translate the data flow in the behavioral specification to weighted arcs in the graph counterpart and to combine different testability measures in node weights, guide the first step. The second step is solved by incrementing rules for the hardware-software description.

Partial-scan needs for the return translation a pointing scheme for the scanned objects among signals/ variables of the behavioral specification. This is managed by an adequate data structure in HDL. In principle, flip-flops are selected for scan, but when a register is used in parallel, it is candidate entirely for scan. The variables/ signals inferring memory are testability-related sorted to select incrementally the scan elements that will be eventually mapped to the scan register.

HIERARCHICAL INTELLIGENT SIMULATION

Essential relations are extracted from non-formalized research on the human mind before searching conscience models enabling intelligent simulation.

Faith and Intelligence are ☺ in our life (Way, Truth, Life)
 Faith = (Inspiration, Intuition, Imagination)
 Intelligence = (Consciousness, Adaptability, Intention)
 Conscience = (Consciousness, Inspiration)

Conscience builds the non-deterministic interface between unconscious faith and the conscious intelligence.
 // Concept = (interface, kernel, messenger).

Human = human (Humanity);
 human \in Faith \times Intelligence \rightarrow Faith \times Intelligence;
 Humanity = (humans Set, evolution-oriented Structure).
 evolution \in (Hunger, Fear, Love)
 \times (Technology, Science, Art)
 \rightarrow (Technology, Science, Art).
 Mathematics \subset Art = Human :: beauty-oriented
 activity (Science, Technology).
 Physics = (natural \cup social) Science
 = Human :: truth-oriented activity (Art, Technology).
 Technology = Human :: good-oriented
 activity (Art, Science).
 Simulation \in Behavior \times Structure \Leftarrow Knowledge;
 Knowledge \Leftarrow Intelligence :: information();
 Imagination \Leftarrow | Intuition - Consciousness |;
 Intention \Leftarrow | Inspiration - Adaptability |;
 Adaptability \Leftarrow simplifying_Abstraction (Imagination);
 Consciousness \Leftarrow reflexive_Abstraction (Intention);

The relations defining informally the class Human are oversimplified in order to move towards intelligent simulation. Although we claim they are intuitive and hope they are inspired, to begin, we neglect the essential but too primitive to understand intuition and inspiration, so formalizing the simplifying abstractions mainly by the simulation hierarchy type, and the reflexive abstraction by the knowledge hierarchy type:

Conscience = knowledge (simulation (Conscience))

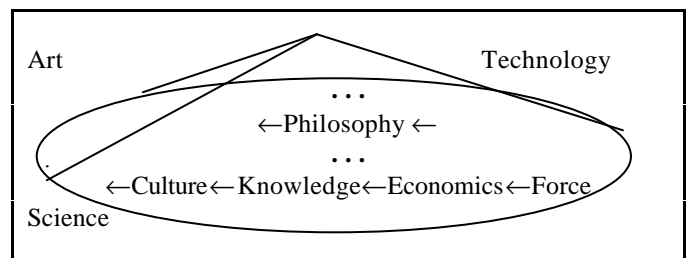
The fixed-point relation suggests to model conscience by association of a knowledge level to any hierarchical level of the simulation process. To solve the problem we build a metric space where knowledge^oconstruction is a contraction - the elements implied in the construction get closer to one another in the formal understanding of the formal construct. Considering the general relations between the parts of the faith-assisted intelligence:

Conscience = knowledge (intention (Inspiration, simulation (imagination (Intuition, Conscience))))

Mathematics contains structures that suggest to be used for self-referent models. The richest domain in this sense is functional analysis, which integrates algebra, topology and order (Kolmogorov 1977):

- contractions and fixed points in metric spaces,
- reflexive normed vector spaces,
- inductive limits of locally convex spaces,
- self-adjoint operators of Hilbert spaces,
- inversable operators in Banach algebra.

Evolution is linked to the initial design of mental faculties for surviving of the whole system, but also to the space-time context for communication between intelligent agents. Conscience is self-awareness of individual faith and intelligence, as well as of the relation to the local context (society) and to the global one (Universe/ Reality). To appear it needed self-knowledge, what could have resulted from community conscience featured by an eternal human structure, e.g., from the past: shepherds, farmers, sailors, Africans, Amerindians, ... Each individual recognized himself in his cohabitants, being most adaptable and having a lot of intuition. The common measure evolution



implies the construction of correspondingly intelligent agents to manage the lower stages and to concentrate on the higher ones. E.g., industry enabled the mechanization in agriculture preparing for the concentration on economics. Evolution is a multiple Divide et Impera et Intellige for conscience, generating the components lacking of the mind at start, then assisted by them:

- individual-social-universal conscience (subjective-contextual-objective) \rightarrow *inspiration* -
- space-time (structure-behavior) \rightarrow *imagination* -
- discrete-continuous (natural-real) \rightarrow *intention* -
- beauty-truth-good (art-science-technology).

The convergence process of evolution demands struggle against time, with structure as ally. Structure is sometimes too conservative, so it has to be reconfigured, at abstract levels, e.g., a plan, as at concrete ones. Conscience needs continuous feedback, not only discrete recurrence. Social and individual conscience are mostly divergent nowadays, i.e., we only performed Divide et Impera, neglecting *et Intellige*. It's high time to correct this! Intelligence in evolution is the faculty to transform (analyze/ synthesize/ modify) abstract/ natural/ artificial objects, and representations, in the correspondent worlds of arts, sciences and technologies, especially hierarchical reflexive:

ideas about ideas, how to get to ideas, objects to transform objects, representations on representations, how to build/understand representations.

Recurrence is confined to discrete worlds, while abstraction is not. This difference suggests searching for understanding based on mathematical structures that order algebra into topology. Recurrence of structures and operations enables approximate self-knowledge, with improved precision on higher levels of knowledge hierarchies. A continuous model for hierarchy levels, without losing the hierarchy attributes, would offer a better model for conscience and intelligence. A possible interpretation of knowledge hierarchies is: real time of the bottom levels (corresponding to primary knowledge/ behavior/ methods) is managed at upper levels (corresponding to concrete types/ strategies/ models) and abstracted on highest levels (corresponding to abstract types/ theories/ techniques).

A topology on the space of symbolic objects permits grouping items with common properties in classes. A dynamically object-oriented internal representation results, that can be adapted to the different hierarchy types. Topological concepts, as neighborhood, or concepts integrating mathematical structures, as closure, can be applied in verification and optimization, for objects and classes.

The simulation environment prepares a framework for representing entities and relations of the system to be simulated as general knowledge about the simulated universe. Knowledge-hierarchy-oriented architecture, both at environment and simulation component level, ensures flexibility of the framework realization, by defining it precisely only in the neighborhood of solved cases. For representation, this principle offers the advantage of open modeling. The user describes models, following a general accepted paradigm that ensures syntactic correctness, leaving the meaning to be specified by user-defined semantic functions that control the simulation.

E.g., a module in an unfinished design can be characterized by constraints regarding its interaction to other modules; the constraints system is a model, open to be interpreted, thus implemented, differently, adapting to criteria in a non-monotonic logic.

Let $(U, \{H_i \in S_h\})$ be a universe structured by different hierarchies H_i and S_h the set of hierarchies defined on it.

$$H = (\text{Rel}_{\text{eq}}, \{(\text{Level}_j, \text{Structure}_j) \mid j \in S_l\}, \text{Rel}_{\text{ord}}, \{A_j \mid j \in S_l\})$$

H is a generic hierarchy, S_l the set of hierarchy levels, Rel_{eq} the equivalence relation generating the levels, Structure_j the structure of level j , Rel_{ord} the (total) order relation defined on the set of hierarchy levels, and $A_j \subset \text{Level}_{j-1} \times \text{Level}_j$, $j \in S_l$ the abstraction relation. U is a category, e.g., containing Hilbert spaces with almost everywhere-continuous functions as morphisms, enabling different ways to simulate self-awareness.

A hierarchical formal system can be defined. Considering self-adjoint operators as higher-level objects of the knowledge hierarchy, these levels can approach self-knowledge in the context of knowledge about the inferior levels as of the current one, and having some qualitative knowing about the superior levels. The self-knowledge raises as the abstraction corresponding to the hierarchy level.

1. $(U, \{H_i \in S_h\})$, $\text{card}(U) > \aleph_0$ // hierarchical universe
2. $\Sigma = F \cup L \cup A \cup K$ // functional objects
 - $F = \{f \mid f \in U^* \rightarrow U\}$ // global functions
 - $L = \{f \mid f \in \text{Level}_j^* \rightarrow \text{Level}_j\}$ // level structures
 - $A = \{f \mid f \in \text{Level}_j^* \rightarrow \text{Level}_{j+1}\}$ // abstractions
 - $K = \{f \mid f \in \text{Level}_j^* \times \text{Level}_{j+1} \rightarrow \text{Level}_{j+1}\}$ // knowledge abstractions
3. $I = \Sigma^* \cap R$ // initial functions
4. $R = \{r \mid r \in \Sigma^* \times R^* \rightarrow \Sigma \times R\}$ // transformation rules.

The correspondence problem, i.e., associating the knowledge hierarchy to the simulation hierarchy, is managed by natural transformations over the various functors of the different hierarchies regarding the simulated system. To complete the simulation of the intelligence's components, intention is first determined by human-system dialogue.

The alternative ways followed to extend the computability concept are suggested by approaches known from German literature, which is philosophy-oriented, trying to express essential ideas that link to the unconscious part of our mind. They respectively concentrate on the mental world of the good managed by technology, the physical world of the truth researched by science, and Plato's ideal world of abstractions discovered by arts.

1. Faust (Johann Wolfgang von Goethe): heuristics - risking competence for performance, basing on imagination, confined to the mental world.
2. Das Glasperlenspiel (Hermann Hesse): unlimited natural parallelism - remaining at countable physical suggestions, so in the Nature.
3. Der Zauberberg (Thomas Mann): hierarchical self-referential knowledge - needing to conciliate the discrete structure of hierarchy with the continuous reaction, hoping to open the way to Reality.

CONCLUSIONS

Conscience simulation demands transcending the present limits of computability, by an intensive effort on extensive research to integrate essential physical and mathematical knowledge guided by philosophical goals. Formalizing hierarchical descriptions, we create a theoretical kernel for self-organizing systems. A way to begin is hierarchical reconfigurable cosimulation. Applying Divide et Impera et Intellige to hierarchy types, using the formalism of categories, reveals their comprehensive constructive

importance based on structural approach, symbolic meaning, object-oriented representation. Further than modeling conscience to simulate intelligence we will be searching to comprehend inspiration, using Lebesgue measure on differentiable manifolds and non-separable Hilbert spaces. Perhaps even mathematics will have to develop more philosophy-oriented to approach intuition.

Simulability is computability by the power of continuum.

There are enough positive signs for this from analog electronics, control systems, mechatronics. Real progress towards this way of computation needs unrestricted mathematics, integrated physics and thinking by analogies. Evolution implies the separation of faith and intelligence, so we have to better understand both, integrating them to human wisdom, to be divided further to get more human. Metaphorically phrased, our searches and researches should have as axioms:

God is unique.
Uncountable are His ways.
Hierarchical are His plans, so we hope.

REFERENCES

- Ageron, P. 2001. "Limites inductives point par point dans les categories accessibles". *Theory and Applications of Categories* 7, No. 1, 313-323.
- Blum, L.; F. Cucker; M. Shub; and S. Smale. 1998. *Complexity and Real Computation*, Springer, Heidelberg/ NewYork.
- Bourbaki, N. 1966. *Éléments de Mathématique. Théorie des Ensembles. Structures*. Hermann, Paris.
- Ciaffaglione, A. and P. diGianantonio. 2001. "Constructive Real Numbers". *Proceedings of TYPES. Lecture Notes in Computer Science* 1961, 2277-2292.
- Goldstein, S.C. and M. Budiú. 2001. "Nanofabrics: Spatial Computing using Molecular Electronics". *Proceedings of the 28th Annual Symposium on Computer Architecture* (Göteborg, Sweden, Jun.17-19). IEEE, Piscataway, NJ. , 178-191.
- Hofstadter, D. 1979. *Gödel, Escher, Bach: an Eternal Golden Braid*. Basic Books Inc., New York.
- Kolmogorov, A. and S. Fomin. 1977. *Analyse Fonctionnelle*. MIR, Moskow.
- Miller, R.; V.K. Prasanna-Kumar; D. Reisis; and Q.Stout. 1993. "Parallel Computations on Reconfigurable Meshes." *IEEE Transactions on Computers* 42, No.6, 678-692.
- Niculiu, T. and S.D. Cotofana. 2004. "Hierarchical Simulation of Intelligence", *Proceedings of the 16th Conference on Systems Research, Informatics and Cybernetics* (Baden-Baden, Germany, Jul.29-Aug.5). The International Institute for Advanced Studies on Systems Research and Cybernetics, Windsor, Canada, keynote, 45-51.
- Niculiu, T. and S.D. Cotofana. 2003. "Hierarchical Reconfigurable Simulated Intelligence Templates", In *Proceedings of the International Conference on Intelligent Systems and Control*. (Salzburg, Austria, Jun. 22-25). The International Association of Science Technology and Education for Development, Calgary, Canada , 39-44.
- Niculiu, T.; C. Aktouf; and S.D. Cotofana. 2002. "High-Level Intelligence oriented Simulation". In *Proceedings of the International Semiconductor Conference* (Sinaia, Romania, Oct.8-10). IEEE, Piscataway, NJ. , 381-385.
- Penrose, R. 1989. *The Emperor's New Mind*. Oxford University Press, Oxford.
- Plesu, A. 2003. *About angels*. Humanitas, Bucharest,
- Rabaey, J. 1997. "Reconfigurable Computing". In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing* (München, Germany, Sep.22-24). IEEE, Piscataway, NJ. , 127-132.
- Traub, J.F. 1999. "A Continuous Model of Computation". *Physics Today*.5, No.5, 39-43.
- Zeigler, B.; H. Praehofer; and T. Kim. 2000. *Theory of Modeling and Simulation*. Academic Press, Oxford.
- Zhong, N. and K. Weihrauch. 2003. "Computability Theory of Generalized Functions". *Journal of Automated Computer Machinery*.50, No. 6, 574-583.

BIOGRAPHIES

TUDOR NICULIU is Professor at the Electronics, Telecommunications and Information Technology Faculty of the University POLITEHNICA in Bucharest and Senior Researcher at the Center for New Electronic Architectures of the Romanian Academy. He is looking for hierarchical integration of different domains, as: hardware-software, discrete-continuous, electrical-non-electrical, in order to simulate intelligence to understand it and to apply it to intelligent simulation. Therefore he formalizes hierarchies. He wants to contribute at comprehension and surpassing of the computability limits, functionally or structurally. Since 19991 he teaches and researches at the same institution (PhD 1995, MS 1985). Before he was Senior Researcher at the R&D Institute for Electronic Components in Bucharest, researching and designing hierarchical simulation of integrated circuits. He also studied Mathematics at the University of Bucharest (MA 1994).

ANTON MANOLESCU is Professor at the Electronics, Telecommunications and Information Technology Faculty of the University POLITEHNICA in Bucharest. He researches, teaches and masters Phd students on Analog Integrated Circuits and Technologies. He published more than 100 articles in international journals and conference proceedings. He has 3 invention patents and a prize of the Romanian Academy for his outstanding activity.

SORIN COTOFANA is Professor at the EE Dept., Delft University of Technology. He received the MS degree in Computer Science from the POLITEHNICA University of Bucharest, Romania, and the PhD degree in Electrical Engineering from Delft University of Technology, The Netherlands. He worked for a decade with the Research & Development Institute for Electronic Components in Bucharest, working on structured design of digital systems, design rule checking of integrated circuits' layout, logic and mixed-mode simulation of electronic circuits, testability analysis and image processing. His research interests include: computer arithmetic, custom computing machines, embedded systems, parallel architectures, logic design, computational geometry, neural networks, and CAD.