Resonance Thinking and Inductive Machine Learning

Yves Kodratoff, Marta Franova
LRI, UMR8623 du CNRS & INRIA Saclay
Bât. 660, Orsay, France
e-mail: yvekod@gmail.com, mf@lri.fr

***Abstract*—This paper presents one of the symbiotic parts deemed necessary to complete our theory of computer systems design devoted to incomplete domains, here called Cartesian Systemic Emergence (CSE). CSE is dealing with one version of the concept of (semi)-automated creativity. We call systems implementing this kind of creativity Symbiotic Recursive Pulsative Systems (SRPS). SRPS are intended to contribute to solving real-world problems in incomplete domains requiring control and prevention. CSE is concerned with strategic aspects of the conception of such SPRS. Each component of a SPRS has to be symbiotically linked to all the other components. This requirement is not very usual in Computer Science, hence we have to introduce notions that are not yet present in scientific vocabulary. This paper is devoted to the most important features of one particular way of thinking present in CSE. We call it 'Resonance Thinking' (RT). RT takes care of generating and handling experiments during CSE. We explain that RT causes the complexity of CSE to be analogous to Ackermann's function computation complexity.**

Keywords—Cartesian Systemic Emergence; Symbiotic Recursive Pulsative Systems; Resonance Thinking; Computer-based and Human-based creativity; Systems Design.

I. INTRODUCTION

The need for Symbiotic Recursive Pulsative Systems (SRPS) design raised during our search for a solution of a complex real-world application, namely automatic construction of recursive programs in incomplete domains [4] [7]. For simplicity, we refer to it as Program Synthesis (PS). Our aim in PS has been to tackle incompleteness and informal specifications that represent problems dealt with neither in classical PS nor in system design approaches [7] [13]. Incompleteness and informal specifications required the introduction of a new model and on-purpose methods in general system design. We call Cartesian Systemic Emergence (CSE) this new theory. CSE handles strategic aspects of the design and particular evolutive improvement of SRPS. The construction and desired improvements have to guarantee control and prevention in the system. Resonance Thinking (RT) introduced in this paper takes care of generating and handling experiments during CSE.

In [9], we describe several facets of CSE, namely tackling underspecified information, on-purpose invention instead of manipulating a specific search space, and formulating fruitful experiments. RT, as a symbiotic part of CSE, possesses also these facets and describes them in a more precise way, even though a formal description has still to be worked out. Since the illustrations of RT in PS are very complex, we shall re-use here the toy example presented in [9]. The purpose of this paper is four-fold: to

- describe particularities of RT taking place in CSE;
- illustrate this method on a toy example nevertheless dealing with a problem that many innovative researchers may have to face;
- explain that the complexity of RT is similar to a computation process of Ackermann's function;
- propose several new strategies relative to System Design.

The paper is organized as follows. Section II presents fundamental notions necessary for understanding CSE and RT. Section III recalls the notion of CSE. Section IV presents the RT problem formulation intertwined with an example. Finally, Section V describes several challenges that RT offers to implementation of a theory of creative thinking.

II. FUNDAMENTAL NOTIONS

The goal of CSE is to formalize strategic aspects of human creation of *informally* specified *symbiotic systems* in *incomplete domains* following our *pulsation* model. This formalization is performed in order to prepare fundamentals for designing automated tools that help to perform this complex task. In this section, we recall four terms by which this goal is expressed and that will be used also in our presentation of RT, namely

- informal specification,
- symbiosis,
- incompleteness, and
- pulsation.

Informal specification of a system that has to be constructed is a description of this system in terms that are not yet exactly defined and that, when considered out of a particular context, may even seem absurd. These terms, in which the specification is expressed, will evolve during the system construction. In other words, depending on some constraints and opportunities that will arise during the construction, the meaning of the terms used in the starting specification will evolve and will make a part of the solution. The initial ambiguity of terms is eliminated by the provided solution. We might say that notions used in an informal specification are of evolutive and flexible character. Their evolution will also bring an exact specification of the context to be considered.

In the context of CSE, the notion of informal specification needs to be completed by differentiating the notions of formalized and formal specification. *Formalized specification* is an intermediary state in the progress from informal to formal specification. It consists in a collection of basic working definitions and basic tools that seem plausibly pointing out a successful completion process, even though some inventive steps may still be needed to complete the tools so as giving their final form to the working definitions. *Formal specification* then consists in the complete solution represented by the working system, the methodology of the functioning as well as of the construction of the system. These all are needed in order to be used in further evolutive improvement.

As far as *incompleteness* is concerned, from a practical point of view, we know that full reality is unknown. What we may know at a given time can be formalized by an incomplete system. From a decision point of view, it is well-known that incompleteness constitutes a large drawback [12]. Incompleteness, however, is not at all a drawback for the practical purpose of solving real-world problems that are asking for some kind of innovation. This is due to the fact that, from a construction point of view, incompleteness brings a freedom for technological ingeniosity, resulting in possible new technological inventions. Since informal specification contains terms that are not exactly defined, a particular informal specification points out to a context that can be represented by an incomplete environment. CSE can then be seen not only as a construction process for a system in its informally specified initial environment but also as a fruitful strategy for a progressive completion of this environment.

By *symbiosis* we understand a composition of several parts which is vitally separation-sensitive. By vital separation-sensitivity we mean that eliminating one part leads to the destruction or to a non-recoverable mutilation of the other parts and of the whole composition. This means that the widely used divide and conquer strategy is not at all suitable when creating and extending symbiotic systems. We can also say that analysis and synthesis are inappropriate tools when creating and observing symbiotic systems. Symbiosis is therefore different from synergy that is a mutually profitable composition of the elements that are not destroyed nor mutilated by a separation.

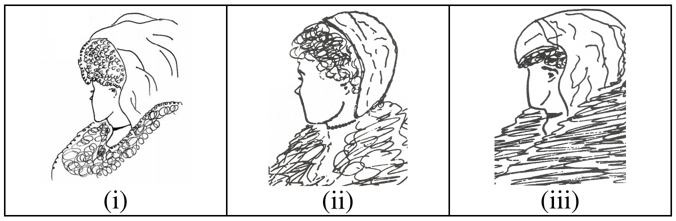

Figure 1. Example of pictorial symbiosis.

In Figure 1, (i) can be seen as a symbiosis of (ii) and (iii). Here, we need to point out that symbiotic parts do not necessarily need to overlap in the final symbiotic object. They may have a symbiotic, and maybe invisible, intersection that makes their whole symbiotic. From a pragmatic point of view, symbiosis of a system is embodied by the interdependence of all notions and parts of this system. We have illustrated this, in [6], on the example of Natural Numbers defined by Peano's axioms.

Pulsation is a model for construction and evolutive improvement of incomplete systems that are concerned with the factors of control and prevention. In other words, pulsation provides a rigorous framework for the completion process of incomplete systems. This model is described in [8]. It relies on our particular handling of Ackermann's function. We shall recall now the features of its handling that will be referred to, later in the paper.

Let 'ack' be Ackermann's function defined, as in [17], by its standard definition, i.e.,

$$\text{ack}(0,n) = n+1 \qquad (1)$$

$$\text{ack}(m+1,0) = \text{ack}(m,1) \qquad (2)$$

$$\text{ack}(m+1,n+1) = \text{ack}(m,\text{ack}(m+1,n)). \qquad (3)$$

Since ack is a non-primitive recursive function, thus by definition of non-primitive recursion, it is a particular composition of an infinite sequence of primitive recursive functions. In similarity to the infinite sequence, which is used – in [11] – to construct Ackermann's function, the evolutive improvement (i.e., pulsation), relies on a construction of a potentially infinite sequence of systems that might, in an ideal world, be used to construct a global 'Ackermann's system' that contains all of these systems. In our work, by pulsation we thus understand a progressive construction of a potentially infinite sequence of incomplete theories $T_0, T_1, \ldots, T_n, T_{n+1}, \ldots$ such that $T_i \subset T_{i+1}, T_i \neq T_{i+1}$ (for i = 0, 1, 2, …) and such that an infinite limit of this sequence represents an ideal, complete system. In addition, each T_i is practically complete in the sense that, from a practical point of view, it covers an exploitable formalization. (Think, for instance, of the incompleteness of natural numbers [12] but their practical completeness in our everyday use.) Pulsation does not reduce to one particular step in this sequence. This follows from that pulsative systems are formalized progressively and potentially indefinitely. Pulsation is a model that does not describes how the particular systems in this sequence are constructed. This is the role of Cartesian Systemic Emergence [9].

III. Cartesian Systemic Emergence

As said above, CSE goal is formalizing strategic aspects of human creation of informally specified symbiotic systems in incomplete domains. In this section, we recall two paradigms that play a fundamental role in CSE and that will be referred to in Section IV. The first paradigm can be represented formally by the formula

$$\forall \text{ Problem } \exists \text{ System solves(System,Problem)}. \qquad (4)$$

The second one can be represented by the formula

$$\exists \text{ System } \forall \text{ Problem solves(System,Problem)}. \qquad (5)$$

There are two main differences between these two paradigms. The first difference is that, in (4), each problem or a class of problems related to a system can have its own solution while in (5) a unique, universal solution is looked for. The first paradigm leads to a library of particular heuristics, while the second paradigm results in a single universal method. CSE is concerned with the pulsative construction of a system that verifies (5).

As presented in [9], the main features of CSE are thus as follows:

- It works with an informally specified goal.
- It handles incompleteness.
- It takes into account symbiosis and pulsation.
- It generates experiences.
- It oscillates between the paradigms (4) and (5) in order to reach a solution described by (5).

To our best knowledge, there is no other work on simultaneously solving the problems addressed by these features of CSE. This explains why we need here so many new concepts and mechanisms. Moreover, because we deal with symbiosis, pulsation and informal specifications, CSE has to be considered in the framework of Cartesian Intuitionism and not in the framework of Newtonian Science. In [7], we explain in more details that the main keywords of Newtonian Science are

- exactness
- formal systems and tools justified in a logical way
- methods of demonstration reduced to some axioms and rules of inference
- decision and undecidability.

In contrast to this, as pointed out in the same paper, the main keywords of Cartesian Intuitionism are

- realization and ingeniosity
- systems and tools justified in an epistemological way
- methodology of construction taking into account also ‘Cartesian Intuition’ (i.e., a symbiotic composition)
- handling incompleteness in a constructive way.

This means that Cartesian Intuitionism has its own, we might say ‘pragmatic’, notion of rigor that enables, during the research and development stages, relying on methods and tools that do not verify the strict criteria of Newtonian Science. This non-conformity to logical criteria and a kind of ‘rigorous freedom’ will become clear in the next sections.

It happens that the process of construction of informally specified symbiotic systems is very difficult to describe exactly and in its full generality. Our CSE attempts to tackle the task of its description. In [9], we present a general, even though yet informal, scheme for CSE based on the method called Constructive Matching formula construction (*CM-formula construction*) which is used in PS (introduced in [3]). We shall refer here to it as CSE-scheme.

IV. Resonance Thinking

RT is a method for solving problems represented by paradigm (5). It takes care of generating and handling experiments in the process of CSE.

In order to take hold of RT complexity, it is necessary to keep in mind that CSE and RT are designed for the creation of systems that have to provide control and prevention. Therefore, the criterion of security is strongly involved already in the system’s creation. Such a particular security follows from fulfilling four precepts of Descartes’ method [2], p. 120:

a) “Carefully avoid precipitate conclusions and preconceptions.
b) Divide each of the difficulties into as many parts as possible and as may be required in order to resolve them better.
c) Suppose some order even among objects that have no natural order of precedence.
d) Make enumerations so complete and so comprehensive, so we can be sure of leaving nothing out.”

Note that these four rules represent also four fundamental (symbiotic) facets of CSE in this order:

a') Pulsative Thinking, i.e., taking care of security, control and prevention [11].
b') Metamorphic Thinking, i.e., taking care of resulting epistemological equivalence between paradigm (5) and particular CSE-handling paradigm (4).
c') Symbiotic Thinking, i.e., taking care of construction of a symbiotic system.
d') RT, i.e., taking care of generating and handling experiments.

As one can realize while trying to give an *exact* description of old-young lady picture given in Figure 1.a, a description of a one part in a symbiotic composition (such as ‘old lady’ in Figure1.c) is not a simple task. Indeed, an exact description of the old lady part in Figure 1.a would imperatively require explicit references to young lady part of Figure 1.a. Therefore, in this paper, we do not intend yet to provide a complete description of RT, because we first need to describe in more details Metamorphic Thinking (MT) and Symbiotic Thinking (ST).

We shall present RT and its basic notions with the help of a toy example used in [9] for description of CSE. In comparison to examples provided by PS framework, this example is simpler and could illustrate many other scientific fields than PS-research does. The problem presented here concerns conveying a new original scientific knowledge in such a way that its essential content and creative potential are preserved by the next generations. This is not a trivial problem as already pointed out in the past [1] [2]. Our experience confirms that, for new knowledge relative to creation and extension of symbiotic recursive systems, this problem remains relevant today also.

Since CSE and RT handle incompleteness it is only natural that procedures and notions of CSE come out through a progressive evolution from informal specifications to formal specifications.

A. *Specification of a toy example*

In this section, we present our example illustrating CSE and RT. Let us suppose that René is a founder of a novel scientific theory with a high pulsative potential. Referring back to many unpleasant experiences of the past founders, he needs to ask himself how to build some ‘works’ able to convey the full complexity of his new theory while immediately preventing a degradation of its pulsative potential. In a more formal way, René must solve a problem informally specified as:

$$\exists \text{works}\ \forall \text{disciple conveys(René, works)}\ \&\ \text{conveys(works,disciple)} \Rightarrow \text{essential\_of(René)} = \text{essential\_of(disciple)} \qquad (6)$$

Note that this problem has the same logical structure as the second paradigm presented in the form (5). Specification (6) is an informal specification. As said above, this means that the notions that appear in (6) are not defined in a rigorous way. They are only specified in an informal way in terms of some non-formal criteria (i.e., a kind of underspecified constraints). This means that a solution 'works' for (6) has to emerge simultaneously with suitable formalizations (thus, the final definitions) of notions that occur in (6). In the following, we shall denote by D_t the set of (initially underspecified) sentences specifying 'to convey' and by D_e the set of (initially underspecified) sentences specifying 'essential_of'. These two sets evolve in the process of CSE and RT towards a more rigorous final form. For simplicity of presentation, we do not involve such an evolution in our notation.

In [9], we mention that, in order to solve (6), there is a particular switch to a framework of experiences described by the formula

$$\forall \text{disciple}\ \exists \text{works conveys(René, works)}\ \&\ \text{conveys(works,disciple)} \Rightarrow \text{essential\_of(René)} = \text{essential\_of(disciple)} \qquad (7)$$

This formula represents the paradigm (4). We have explained above that there is a difference between solving (4) and (5), and this obviously applies to their instances (6) and (7). In general, in order to be fruitful and justified, a switch from (5) to (4) has to rely on what we call Metamorphic Thinking (MT). Roughly speaking, MT takes care of a rigorous, epistemologically and pragmatically justified transformation of paradigm (5) into the context of paradigm (4). Our paper [9] gives its illustration in the field of program synthesis from specifications. A more detailed description of MT is presently under development. Note that we call *oscillation* the process of switching between these two paradigms.

In other words, MT provides a switch from (6) to (7) that is useful in order to generate experiences generating, within the framework of (7), some hints and inspiration for solving (6). These hints and inspirations represent temporary (see precept (a)) underspecified constraints that enlarge the already existing set of underspecified constraints. In order to generate such inspiring experiences, while considering (7), from the set of all disciples, we chose a finite number of disciples $d_0, d_1, \ldots, d_n$ that seem highly different so that each of them seems to need a different 'works'. Note that this step implicitly embodies the above precept (b). We shall call *representatives* these disciples. In other words, our experience shows us that challenging experiences are needed to obtain some inspirations contributing to a solution of (6) in the framework of paradigm (5). Note that we order these disciples in a numbered sequence just for the presentation purposes. This will be useful when describing recursive procedures that handle this finite set of disciples.

Very roughly speaking, in order to solve a problem represented by paradigm (5), it might seem possible to replace MT from paradigm (5) to (4) by a symbiotic composition of a set of solutions for carefully chosen representatives of universally quantified elements of this paradigm. A drawback of such a description lies in considering a lone symbiotic operation (i.e., one action), while RT, through precepts (a), (b), (c) and (d) requires performing a great number of interdependent symbiotic compositions, as will be described below.

Recall that the two operators 'conveys' and 'essential_of' are here specified informally only by some set of sentences that represent informal descriptions (i.e., underspecified constraints) relative to these notions. Thus, we shall replace these notions by their informal descriptions. Above, we have denoted by D_t the set of sentences specifying 'to convey' and by D_e the set of sentences specifying 'essence_of'. Therefore, (7) writes as

$$\forall \text{disciple}\ \exists \text{works}\ D_t(\text{René, works})\ \&\ D_t(\text{works,disciple}) \Rightarrow D_e(\text{René}) = D_e(\text{disciple}) \qquad (8)$$

Let us consider (8) for each particular d_i, i.e.,

$$\exists \text{works}\ D_t(\text{René, works})\ \&\ D_t(\text{works}, d_i) \Rightarrow D_e(\text{René}) = D_e(d_i) \qquad (9)$$

In [9], we show that a solution for (9) can be found for each d_i by following CSE-scheme and oscillating between paradigms (4) and (5). This solution consists of a concrete value w_i for 'works' and of less informal descriptors $D_{t,i}$ and $D_{e,i}$. We shall call this solution $Sol_i = \{w_i, D_{t,i}, D_{e,i}\}$. Due to a careful oscillation between paradigms (4) and (5), w_i and the descriptors $D_{t,i}$, $D_{e,i}$ refine 'works' and the operators 'to convey' and 'essential_of' in (6). These resulting refinements 'resonate' within paradigm (5) framework. By their resonating, we mean that during the experimentation process, we feel that they might, probably after some 'judicious adaptations', be applied also to other instances of 'disciple'.

B. Resonance Thinking

RT relies heavily on what Merriam-Webster Dictionary considers as resonance: a quality that makes something personally meaningful or important to someone. RT thus involves the ability to create and explore personally meaningful or important relations in the process of generating and handling experiences.

Procedurally, RT is based on two procedures of which we cannot here provide a detailed description, since it relies on other CSE symbiotic facets, not yet introduced ones (namely, MS and ST mentioned above). We shall therefore concentrate on explaining the role of these procedures. The first procedure will be called *topological symbiosis* (noted *ts*) and it is also a primitive operation for the second procedure. The second procedure is called *complementary topological symbiosis* (noted *cts*). Both these procedures require creativity in developing symbiotic systems. In this paper, we describe the way these procedures work: they are therefore to be handled, for the time being, by a creative human person. The following description of the role of *ts* and *cts* will illustrate some of the challenges that *ts* and *cts* have to tackle.

1) Topological Symbiosis and ILP

Our argumentation relies besides on some results recently obtained in the field of Inductive Machine Learning (see the relevant references in [15] and [16]) that proved a first instance of what a pioneer in ML, Donald Michie, thirty years ago called "Ultra-Strong Learning" (see [14]) that we dubbed as U-SL. The strategy of U-SL relies on four essential steps. Steps A. and B. are based on machine power, and steps C. and D. will be introduced later in the 'discussion' section of this paper. These two Muggelton et al. papers elaborate on a description of how Prolog programs might be understood or misunderstood by someone in the process of learning this programming language. A side remark may illustrate the depth of U-SL concept: in U-SL: this kind of learning 'unites' machine and human learning in a way that will become clear later.

Step A. Generating knowledge in the form of new predicates that have not been aforehand provided to the system. An efficient way to achieve this goal has been presented in [15] where the system makes use of a controlled pattern matching of the higher order knowledge, provided in the form of meta-knowledge handled by a meta-interpreter. New knowledge is obtained by proving that a meta-goal is valid on a selected set of true examples. Note that this procedure is not submitted to our constraints of symbiosis and pulsation.

In our presentation, Muggleton's Step A. can be seen as a partial instance of what we call here *ts*, the role of which is to create new relationships induced from the data. Since our examples deal with the field of inducing notions and programs from an incomplete specification (which imposes symbiosis and pulsation), we still need to define a *ts* adapted to this very complex task. This explains also why we have to provide a coherent description of *ts* before being able to implement it.

Step B. Once new rules a found during Step A., [16] makes use of these rules in order to select a set of significant examples. This creativity could work in a random way, generating a random mixture of examples illustrating both the old rules and the new generated one. In the context of U-SL, these examples are generated in such an order as to constitute the 'background knowledge' provided to a human learner. Thus, some selection among the possibly generated rules has to be done in order to be sure to obtain a 'significant' background knowledge.

In our presentation, Muggleton's step B. can be seen as a partial instance of what we call here *cts*. The goal of *cts*, similarly to step B., is to select a set of examples in order to complete or to enlarge the new knowledge initially generated by *ts*. However, in difference with step B., *cts* generates random examples because this provides a greater probability of generation of new (useful or missing) knowledge. This is coherent with our choice of a set of disciples for which solving (9) is rather difficult. We have mentioned above that this leads to a necessity of a greater creativity and thus leads more efficiently to practical completeness of resulting system, here 'works' as in (6). At this stage, we already can acknowledge that steps A. and B. may be used as an inspirational model for programming the main procedural features of *ts* and *cts*.

Now that we have provided more intuitive understanding to what are *ts* and *cts*, we can illustrate the strategy we will use in order to implement them through a description of our 'typically symbiotic' examples : 'two different women', 'Peano's axioms' and 'René's disciples'.

2) On symbiosis in RT

We need to point out here two particular features of *ts*. The first one concerns the character of possible "mutilations" performed by *ts* and the second one concerns its goal.

Let us recall first the above two different women given in Figure 1.b and Figure 1.c. The essential difference between these two figures can be expressed by the term 'age'. Indeed, the woman in Figure 1.c looks old and the woman in Figure 1.b looks young. We might say that the goal of topological symbiosis is here to 'merge' these two figures so that the descriptor 'age' has simultaneously two values, namely 'young' and 'old'. In the ML field, this operation is similar to now a classical one called Predicate Synthesis. Obviously, Figure 1.a is a solution for this task. We say that Figure 1.a is a symbiotic composition of Figure 1.b and Figure 1.c. We can however see that that the original figures Figure 1.b and Figure 1.c have been 'mutilated' to satisfy the requirement of this goal. For instance, the eye of old woman in Figure 1.c becomes an ear for young woman projection in Figure 1.a. This pictorial example, however, is too simple for illustrating an essential feature of symbiotic relations: the fact that we are starting with underspecified pieces of the puzzle (i.e., the axioms and constraints available at the start and which fail to solve our problem).

In [6], we used the example of Peano's axioms that are (also) symbiotic since, by deleting one of its axioms, the reduced set of axioms is either non sense or leads also to other interpretation structures (such as the set of Perfect Women in [5]). This example exhibits an explicit degradation due the presence of a set of notions and constraints that obviously became underspecified when one of Peano's axioms is deleted. This shows that symbiosis manifests itself not so much as 'merging' contradictory facets of the considered system, but as constructing an emergent vitally separation-sensitive interdependence (i.e., symbiosis) of parts of the system.

3) On generating experiments in RT

We are going to describe *ts* and *cts* in the framework of René's example. At this stage, we suppose that (9) for d_0 is already solved following the CSE-scheme providing the solution Sol_0 for d_0. Sol_0 represents a 'temporary' solution for d_0. By 'temporary' we mean that this solution will still have to be approved or modified by RT. We can now come to *ts* and *cts*. Similarly, for other disciples d_1, …, d_n, we will obtain Sol_1, Sol_2 and so on. We assume here that the solutions are obtained in a particular 'linear' way, one after another. This 'linear' way looks as follows.

Once Sol_0 is constructed, a 'temporary' solution Sol_1 for d_1 is constructed ('temporary' in the same way as Sol_0 is a 'temporary' solution for d_0). Note that, in order to concentrate on the problem at hand, both these constructions may lead to new experiences and thus, *they modify the initial environment* by refining the informal notions of our definition (6) of our problem. For the sake of simplicity, we do not describe explicitly below this evolution of environment, though we take it into account by calling it a 'feedback' when we use it.

Now, suppose that we solved the problem for the first disciple. Before starting solving the problem for the next one, we try to take into account the informal specifications present in (6). This try amounts to an attempt to 'merge' the solutions Sol_0 and Sol_1 using topological symbiosis *ts*, i.e., we try to achieve their symbiotic composition that resonates (as explained in Section 4) with the informal specifications in (6). We shall denote this process by $ts(\mathrm{Sol}_0,\mathrm{Sol}_1)$.

If solving $ts(\mathrm{S}_0,\mathrm{S}_1)$ fails, i.e., we cannot find relevant refinements, we keep in mind the feedback obtained while constructing Sol_0 and Sol_1, as well as the failure reasons of $ts(\mathrm{S}_0,\mathrm{S}_1)$. This failed step will have to be redone later while relying on some inspirations that may rise while finding the solutions for the next disciples. If this process fails, or leads to an infinite number of repetitions, the problem will have to be considered as a challenge for one of the next pulsation steps.

If the process $ts(\mathrm{Sol}_0,\mathrm{Sol}_1)$ succeeds, both solutions are temporarily approved. Then, keeping in mind all the feedback obtained, a solution of (9) for d_2 is constructed. One might suppose that this process may continue linearly as suggested by its beginning, as we just have seen. However, recall that we work in an environment that requires control and prevention. Therefore, in this environment, we rely strongly on the above four precepts. This means that generating complementary experiments for topological symbiosis of solutions constructed is necessary. We call *complementary topological symbiosis* (noted *cts*) this procedure for generating new experiments.

Roughly speaking, *cts* is a particular generation process (defined with help of *ts*) for creating experiments. The goal of these complementary experiments is to provide inspirations for further refinement for underspecified notions and constraints. Similarly to computation of ack (see [10]), in the process of generating experiments (via *ts*) for Sol_m and Sol_n, i.e., while 'computing' $cts(\mathrm{Sol}_m,\mathrm{Sol}_n)$, the operation $ts(\mathrm{Sol}_i,\mathrm{Sol}_j)$ for other solutions Sol_i and Sol_j is performed several times.

Let us denote by $ts_1(\mathrm{Sol}_i,\mathrm{Sol}_j)$ the solution of the first computation, by $ts_2(\mathrm{Sol}_i,\mathrm{Sol}_j)$ the second computation, and so on. It is important to point out that $ts_p(\mathrm{Sol}_i,\mathrm{Sol}_j)$ and $ts_q(\mathrm{Sol}_i,\mathrm{Sol}_j)$ in this sequence of computations may carry two different feedbacks. Indeed, each inner step of *cts* (i.e., evaluating $cts(\mathrm{Sol}_m,\mathrm{Sol}_n)$), may bring new refinements, constraints as well as it may point out to missing knowledge or second-order notions and procedures. The procedures *ts* and *cts* have to insure that not only reasonable and achievable solutions are obtained but that a possibility of future evolutions are guaranteed while properly handling prevention and control.

The procedures *ts* and *cts* are, in our case, presently performed by a human mind. This means that human mind can rely on relevant creativity in order to decrease the number of repetitions. In consequence, even though *ts* and *cts* are not simple, CSE and RT are not overwhelming tasks for human performers. However, they may be overwhelming for a human observer even in this simplified form. For instance, an observation of the computation steps of ack(3,2) before its ending does suggest that the process leads nowhere because it seems to loop. The same holds for an external observer of CSE. This is why we believe that further research is necessary to give a reasonable formula for performing *cts* by machine.

V. Discussion

It is interesting to check if steps C. and D. of Muggleton's approach to U-SL could serve as an Ariadne's golden thread for us to develop a similar computer-aided explanation of some features of our project.

In section IV.B, we already presented our possible interactions with U-SL Steps A and B. Its Steps C and D both try to measure how much students have been able to understand the way Prolog computes.

Step C. Remember that the students receive a set of rules during Step B. In Step C., the students are offered different programs at different levels of abstraction, i.e. more or less general clauses. They have to understand the relations between the clauses and the examples given.

Step D. The teachers create a questionnaire that checks if the students understood or not the information provided at Step B.

It has been observed that both success and failure provide information about the way the students manage their understanding of Prolog. Successful students provide information on some of their unexpected own way to handle Prolog (i.e., hints at handling it in a creative way), and the various failure cases provide hints to possible repair procedures within incorrectly structured Prolog knowledge. More than delivering a mark of value to the students' learning ability or to the teachers' teaching one, this U-SL approach rather provides clues about how to improve these abilities.

This successful trend of Machine Learning research opens us to some hope that human-based creation of programs from badly or incompletely specified may benefit of the U-SL attitude, as follows.

Example 1., relative to René's goal. Each solution of (9) will generate at least one improved 'works'. We could then, similarly to U-SL steps C. and D., organize a kind of consultation between René himself (the 'teacher') and each particular disciple (the 'student'). In this case, René would test whether his (so far) constructed partial 'works' brings to his potential disciples a correct comprehension of his fundamental notions.

Example 2., relative to Symbiosis among the components of a system. In section IV.B, we have shown that pictorial Symbiosis provides clues for handling symbiosis. As we have seen, symbiosis is better defined by its "vitally separation-sensitive interdependence" among the components, as symbiotic Peano's axioms illustrate. As far as we know, teaching the recognition and handling of separation-sensitive interdependent systems, a skill necessary to creative programmers, does not exist yet. The research presented here provides a few clues of how it could be formalized. A tight collaboration with specialists in Cognitive Sciences should enable us to provide a large enough battery of symbiotic and non symbiotic systems so that, mimicking US-L, we could unravel the deep features of systemic symbiosis, a necessary, if not sufficient, condition to safely handle creativity.

Example 3., relative to Oscillation. Oscillation has been introduced in section III where we underlined the difference between problems of the type (4): ∀ Problem ∃ System solves(System,Problem) and those of the type (5): ∃ System ∀ Problem solves(System,Problem). While exposing "René disciples" example, we used the switch between these two problems by replacing formula (6) by (7). Understanding the nature of a switch from a "∀ ∃" problem to a "∃ ∀" one is by itself not easy, and it is even more difficult to realize that the oscillation from one to the other may lead to a solution respecting the four basic requirements expressed at the beginning of section II. We think that a strategy *à la* U-SL may constitute a tool favoring the understanding of the importance of the shift proposed here.

VI. Conclusion

Cartesian Systemic Emergence is intended to become an implementable system design theory for Symbiotic Recursive Pulsative Systems. In this paper, we have introduced one of its symbiotic features, namely Resonance Thinking. RT takes care of generating and handling experiments during the creation process of symbiotic systems specified, at the start, by an informal specification. RT is very complex, since it has to deal with the requirements of control and prevention, as well as with the process of 'shrinking' the incompleteness in accordance with the pulsation model. We have described also a particular work in Inductive Machine Learning, namely U-SL, which seems to provide a fruitful inspiration for the final implementation of two main procedures of RT (topological symbiosis and complementary topological symbiosis).

RT is only one of four symbiotic features of CSE. We have already presented a basis for Pulsative Thinking, namely the Pulsation model [11]. We are currently working out on its last two features, Symbiotic Thinking and Metamorphic Thinking.

By its symbiotic character, the system design theory proposed by CSE differs from the contemporary approaches to system design (see [13]). However, it does not compete with those approaches. It completes their modular considerations by considerations that are suitable for handling symbiotic pulsative systems.

Acknowledments

We thank Michèle Sebag and Yannis Manoussakis for their moral support.

References

[1] F. Bacon, Novum Organum, P.U.F, 1986.

[2] R. Descartes, "Discourse on the Method," in. R. Descartes, translated by J. Cottingham, R.Stoothoff, D. Murdoch, Philosophical Writings of Descartes, vol. 1, Cambridge University Press, 2006, pp. 111-151.

[3] M. Franova, "CM-strategy: A Methodology for Inductive Theorem Proving or Constructive Well-Generalized Proofs," in A. K. Joshi ed., Proc. of the Ninth International Joint Conference on Artificial Intelligence, 1985, pp. 1214-1220.

[4] M. Franova, "An Implementation of Program Synthesis from Formal Specifications", in Y. Kodratoff, ed., Proceedings of the 8th European Conference on Artificial Intelligence, Pitman, 1988, pp. 559-564.

[5] M. Franova, "The Role of Recursion in and for Scientific Creativity," in R. Trappl ed., Cybernetics and Systems 2010, Proc. of the Twentieth European Meeting on Cybernetics and System research, Austrian Society for Cybernetic Studies, 2010, pp. 573-578.

[6] M. Franova, "A Cartesian Methodology for an Autonomous Program Synthesis System," in M.Jäntti, G. Weckman eds., proc. of ICONS 2014, The Ninth International Conference on Systems, ISBN, 978-1-61208-319-3, 2014, pp. 22-27.

[7] M. Franova, "Cartesian versus Newtonian Paradigms for Recursive Program Synthesis," International Journal on Advances in Systems and Measurements, vol. 7, no 3&4, 2014, pp. 209-222.

[8] M. Franova and Y. Kodratoff, "A Model of Pulsation for Evolutive Formalizing Incomplete Intelligent Systems," in L. van Moergestel, G. Goncalves, S. Kim, C. Leon eds., INTELLI 2017, The Sixth International Conference on Intelligent Systems and Applications, ISBN, 978-1-61208-576-0, 2017, pp. 1 - 6.

[9] M. Franova and Y. Kodratoff, "Cartesian Systemic Emergence - Tackling Underspecified Notions in Incomplete Domains," in O. Chernavskaya, K. Miwa eds., Proc. of COGNITIVE 2018, The Tenth International Conference on Advanced Cognitive Technologies and Applications, ISBN, 978-1-61208-609-5, 2018, pp. 1-6.

[10] M. Franova, "Trace of computation for ack(3,2)", https://sites.google.com/site/martafranovacnrs/trace-of-computation-for-ack-3-2, retrieved 2019.01.21.

[11] M. Franova and Y. Kodratoff, "Cartesian Systemic Pulsation – A Model for Evolutive Improvement of Incomplete Symbiotic Recursive Systems," International Journal On Advances in Intelligent Systems, vol 11, no 1&2, 2018, pp. 35-45.

[12] K. Gödel, "Some metamathematical results on completeness and consistency, On formally undecidable propositions of Principia Mathematica and related systems I, and On completeness and consistency," in J. van Heijenoort, From Frege to Godel, Harvard University Press, 1967, pp. 592-618.

[13] M. Levin, Modular system design and evaluation. Springer, 2015.

[14] D. Michie, "Machine learning in the next five years," Proceedings of the third European working session on learning, Pitman, 1988, pp. 107-122.

[15] S. Muggleton, D. Lin and A. Tamaddoni-Nezhad, "Meta-interpretive learning of higher-order dyadic datalog, predicate invention revisited," Machine Learning 100, 2015, pp. 49-73.

[16] S. Muggleton, U. Schmid, C. Zeller, A. Tamaddoni-Nezhad and T. Besold, "Ultra-Strong Machine Learning, comprehensibility of programs learned with ILP," Machine Learning 107, 2018, pp. 1119-1140.

[17] A. Yasuhara, Recursive Function Theory and Logic, Academic Press, New York, 1971.