— vordenker-archive —

Rudolf Kaehr

(1942-2016)

Title

Towards Abstract Memristic Machines

Archive-Number / Categories

2_53 / K11

Publication Date

2010

Keywords

Memristics, Memristive Systems, Memristor, Morphogrammatics, Konogrammatics, Polycontextiurality

Disciplines

Cybernetics, Computer Sciences, Systems Architecture and Theory and Algorithms, Self-Referentiality

Abstract

Memristic machines are time-tensed machines of the nanosphere. Their definition and their rules are not covered by ordinary logic, arithmetics and semiotics, basic for a theory of abstract automata. The difference to classical concepts of machines to tensed, i.e. memristive machines is elaborated. As an attempt to develop memristive machines, basic constructs from morphogrammatics are applied.

Properties of retro-gradeness (antidromicity), self-referentiality, simultaneity and locality (positionality) of operations as they occur in kenogramamtic and morphogrammatic basic operations, like the successor operations, 'addition' and 'multiplication' have to be realized on all levels of operativity in memristive systems.

Hence, the tiny memristive properties of time- and history-dependence for kenomic successors are presented for all further operations, like "addition" (coalition), "multiplication", "reflection", etc. Morphogrammatics will be further developed in Part II of the paper.

A new framework for design and analysis for memristive systems, i.e. memristics, shall be sketched as a complex methodology of Morphogrammatics, Diamond Category Theory, Diagrammatics and Nanotechnology.

Citation Information / How to cite

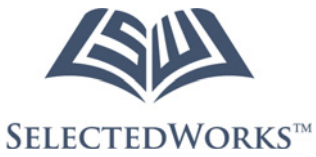
Rudolf Kaehr: "Towards Abstract Memristic Machines", www.vordenker.de (Sommer Edition, 2017) J. Paul (Ed.),
URL: http://www.vordenker.de/rk/rk_Towards-Abstract-Memristic-Machines_2010.pdf

Categories of the RK-Archive

2010

Towards Abstract Memristic Machines

Rudolf Kaehr



Towards Abstract Memristic Machines

Outline of morphogrammatics as a formal model for tensed machines

Rudolf Kaehr Dr.phil.[@]

 ISSN 2041-4358

Abstract

Memristic machines are time-tensed machines of the nanosphere. Their definition and their rules are not covered by ordinary logic, arithmetics and semiotics, basic for a theory of abstract automata. The difference to classical concepts of machines to tensed, i.e. memristive machines is elaborated. As an attempt to develop memristive machines, basic constructs from morphogrammatics are applied.

Properties of retro-gradeness (antidromicity), self-referentiality, simultaneity and locality (positionality) of operations as they occur in kenogramamtic and morphogrammatic basic operations, like the successor operations, 'addition' and 'multiplication' have to be realized on all levels of operativity in memristive systems.

Hence, the tiny memristive properties of time- and history-dependence for kenomic successors are prsented for all further operations, like "addition" (coalition), "multiplication", "reflection", etc. Morphogrammatics will be further developed in Part II of the paper.

A new framework for design and analysis for memristive systems, i.e. memristics, shall be sketched as a complex methodology of Morphogrammatics, Diamond Category Theory, Diagrammatics and Nanotechnology.

Time- and history-dependence in history

Rechnen heisst:

Aus gegebenen Angaben nach einer Vorschrift neue Angaben bilden. (Konrad Zuse)

„Soll die hierdurch bedingte Vieldeutigkeit begreiflich werden, so dürfen wir nicht den jeweiligen Zustand allein ins Auge fassen. Wir müssen auch die Zwischenzustände beachten, die der Körper bis zur Rückkehr in den anfänglichen Zustand durchläuft, und sie für das veränderte Verhalten verantwortlichmachen. Dies führt dazu, die Änderungsweise eines lebendigen Körpers ganz allgemein durch seine früheren Zustände bedingt zu denken." G. F. Lipps, Mythenbildung und Erkenntnis, Leipzig u. Berlin 1907, S. 263 (Zitiert nach: Karl Faigl, Ganzheit und Zahl, Jena 1926, Herdflamme Bd. 2)

„Wenn ich aber jetzt ein kybernetisches System bauen will, das mindestens Spuren oder Grade der Selbstreferenz zeigt, so setzt eine solche Selbstreferenz voraus, dass das betreffende System eine innere Zeit hat, d.h., dass es auf einen früheren Zustand seinerselbst zurückblicken kann. Auf das Früher kommt es an, also auf das Zeitmoment. In diesem Fall genügt die einfache Alternative nicht mehr, dass etwas so oder nicht so ist." Gotthard Gunther

1. Memristors and non-trivial machines

1.1. Trivial vs. non-trivial machines

"*Devices whose resistance depends on the internal state of the system.*" (Kim)

The cybernetician Heinz von Foerster introduced the distinctions of *trivial*, *non-trivial* and *recursively* operating non-trivial machines. Those distinctions got a wide use in cybernetics and in the theory of learning systems. Connection to the common theory of abstract machines had been studied too.

Instead to go primarily into the classic theory of automata to study the possibility of memristic machines, I will focus on this handy distinction of trivial, non-trivial and autopoietic machines.

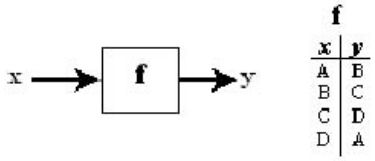
Nevertheless it should be noticed that von Foerster models of machines are formal models in the sense of mathematical and programming systems and not in any way connected with the possibilities of emulations by material systems, like it is possible now with memristive systems. History-dependence is use in computing sense of recursive functions and not in the sense that the "hardware" (XX) in itself is history-dependent in its behaviour.

Furthermore, see Gordon Pask's approach to cybernetics:

"Life and intelligence lie somewhere in the conflict between closed, unique, construction and open, shared, interaction. Between a specific material fabric, and a general conceptual/functional organization."

Some criteria
Leading criteria of comparison shall be:
history-dependent vs. history-independent
determined vs indetermined
predictable vs. unpredictable
programmed vs. self-organized
simulation vs. emulation

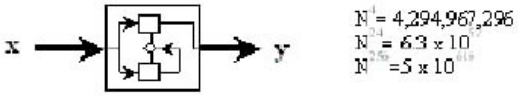
Trivial Machines:
A trivial machine is modeled by the set-theoretic function *f* with the properties:
(i) Synthetically determined;
(ii) History independent;
(iii) Analytically determined;
(iv) Predictable.



Non-Trivial Machines:
In contrast to trvial machines, non-trivial machines have an internal state computed by *z'*:
(i) Synthetically determined;
(ii) History dependent;
(iii) Analytically indeterminable;
(iv) Unpredictable.

Driving function: y = F(x, y)
State function: z' = Z (x, z)
Z = internal state.

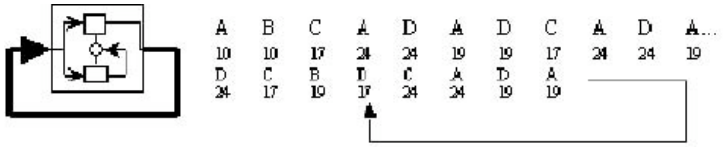
N^4= 4,294,967,296



(i) *Read* the input symbol x.
(ii) *Compare* x with z, the internal state of the machine.
(iii) *Write* the appropriate output symbol y.
(iv) *Change* the internal state z to the new state z'.
(v) *Repeat* the above sequence with a new input state x'.

Further explanations of non-trivial machines: example [1]

Recursively Operating Non-Trivial Machine:
"Computing EigenValues, Eigen-Behaviours, Eigen-Operators, Eigen-Organizartions, etc..."



http://www.cybsoc.org/heinz.htm

What kind of machines are memristive systems representing?
Amazingly, it will turn out that within the possibilities of memristive systems very different types of machines are constructible.
Common to all three types of von Foerster's machine is that they are dealing with data, information, objects, etc., and not with their own conceptual domains and definitions. Hence, even if they are classified as second-order machines, they are not in any sense interactional, reflectional or interventional.

Because memristive behaviors occur in different physical media, a dissemination over disjunct media to realize polycontextural emulations of memristors appears quite natural. This fact might contribute, together with the mem-properties of memristive devices to realize polycontextural machines.

Both, properties, the mem-property, allowing the realization of chiastic self-referentiality and the localization-property doesn't exist in the framework of von Foerster's second-order machines.

The 'property' of localization (or positionality) obviously has nothing in common with localization theories of Sir John Eccles and his engrams or Pribram's hologram theory of memory. Localization in the context of polycontextural studies of computation and memory refers to different contexture. It is well known, that the notion "contexture" and the concept of a multitude of distributed and mediated, i.e. disseminated contextures has no equivalent in classical theories.

Both kinds of machine, trivial and non-trivial, are depending on a pre-given data set of a single general domain.Their languages (Chomsky-Hierarchy) are presuming a pre-given alphabet (sign repertoire), which is, in general, stable and is not changing during computation. In contrast, autopoietic co-creative machines are not dealing with signs therefore they don't presume an alphabet. Their beginnings are changing depending of their usage. Unfortunately, this point never got a clarification in the second-order cybernetic literature.

1.2. Memristors as trivial machines

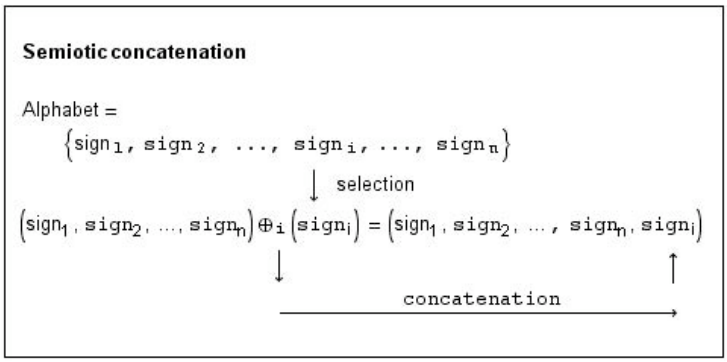
1.2.1. Memristors, semiotics and categories

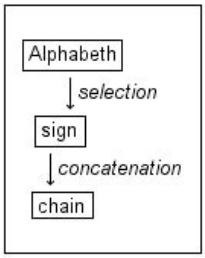
Also memristors are not genuinely representing trivial machines because their *"resistance depends on the internal state of the system"*, while trivial machines are *"history independent",* memristors might be *used* to construct, nevertheless, trivial machines.
Because the behavior of memristors might be interpreted as a *material implication*, and material implication plus the value F are building a logically complete set for propositional connectives, and propositional logic is additionally to the completeness of the junctional set and its axiomatization, *decidable*, memristors are able to perform decidable, i.e. trivial machines, too.

Also the point that memristors are passive elements in contrast to the active time-clocked transistors has to be considered in more detail it will not necessarily change the base of the argument.

Semiotics
Semiotically, trivial machines are based on the operation of *concatenation,* which is the basic construct underlying state transitions. Concatenation is an abstract 'addition' without any possibility to refer to past semiotic events. The mechanisms of feedback loops are not escaping this restriction of concatenation but accelerating it as a function in time.

The concatenation mechanism for sign-sequences consists in a two-level action: *selection* of a sign out of the pre-given sign repertorire (alphabet) and a '*linear addition*' of the selected sign to the existing chain of signs.





This concept of semiotics might be simplified without loss to a semiotic system with one element only, {|}. And the empty sign to mark the blank between signs. Then, all different alphabetic elements are produced by the one-element system as abbreviations. Say, | = a, || = b, etc.
The first rule R_1 of this stroke calculus is introducing a stroke, |, the second rule R_2 says, if there are *n* strokes produced a further stroke might be added, *n*|, the third rule R_3 states the iterativity of the second rule. Hence, there is no need to refere to a "history" of the stroke production. There are obviously some crucial implications involved which are not mentioned in the context of a stroke claculus. Funny enough, all that is repeated,

nevertheless, with George Spencer's Calculus of Indication.
www.thinkartlab.com/pkl/media/SKIZZE-0.9.5-TEIL%20A-ARCHIV.pdf

stroke calculus
Alphabet = { | }:
R_1 : $\Longrightarrow$ |
R_2: $n \Longrightarrow n|$
R_3: iteration of R_2

Category theory
Trivial machines are perfectly modeled by category-theoretical approaches. Classical category theory is studying the constructions based on composition. Composition is interpreted generally as a serial connectivity (Abramsky, Coecke), while an additional operator, like yuxtaposition, is understood as *parallel* connectivity. Both, composition and yuxtaposition, nevertheless, are defined strictly within the paradigm of a-temporal and "history independent" structures and operativity.

This *"history-independency"* is axiomatized in category theory by the axioms of associativity and identity for composition of morphisms. Both axioms are not referring in their definition *retro-grade* to previous events.

Identity: $I_A \circ f = f = f \circ I_A$
Associativity: $(f \circ g) \circ h = f \circ (g \circ h)$

Category Theory and Computer Programming (Eds., David Pitt, Samson Abramsky et al, 1985, Springer LNCS 240) gives a definitive introduction, overview and programatics for future studies.

The axiom of associativity guarantees the *"history-independent"* charactor of the operation of *composition*. But this is necessarily secured by the first axiom, the identity of $I_A \circ f$ and $f \circ I_A$as f . There is no such property like *retro*-grade reference for categorical composition, and for yuxtaposition too.

In a interesting remark, John Baez points to the fact that non-commutative systems got studied in category theory, and elsewhere, but there seems to be not much interest for non-associative formalisms for category theory. Non-associativity might be well studied for general algebras but not as categories.

An up-to-date approach of modern category theory is presented by Peter Selinger:[22]
In contrast to the memristor-papers, this important research report is *free* accessible at:
http://www.mscs.dal.ca/~selinger/papers/graphical.pdf

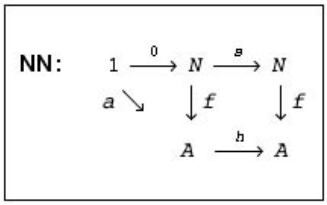
Finite State Machine
- A deterministic finite state machine or *acceptor* deterministic finite state machine is a quintuple (Σ, S, s0, δ, F), where:
- Σ is the input *alphabet* (a finite, non-empty set of symbols).
- S is a finite, non-empty set of *states*.
- s_0 is an *initial* state, an element of S.
- δ is the state-*transition* function
- F is the set of *final* states, a (possibly empty) subset of S.

The state-transition function is "*"time- and history-independent"* as it should be by definition. It is a morphisms from the *initial* to the *final* states based on the input alphabet delivering its out-put.

Concatenation in poly-categories
The simplest concatenation system is the system of natural numbers NN. Category-theoretically, this system is introduced as a commutative graph with (0, s) for the numbers and (a, h) for the model. Such a system is based or anchored on an *initial* object "0" as the starting point of the linear succession. This is stated in the *recursive* formula:

$$f(0) = a$$
$$f(n+1) = h(f(n))$$

A *categorification* of the recursive formula is introduced by the following steps.
For short, if *0: 1 --> N, s: N --> N* and *a: 1 --> A, h: A --> A* are morphism and are producing a diagram that commutes then the object defined with this graph are the natural number system NN. The unique morphism, which make the diagram commute is the morphism f.



This definition or construction is categorical because it is defining the interactions, i.e. morphisms, between the objects, and is not defining natural numbers as elements of a set with specific properties. It defines the natural number system, NN, as a correlation (morphism) between the numerical *system*, (N, s), and a *model* (a, h) of the numerical system "(N, s)".

The question arises: How can we compare two natural number systems NN? Each NN is categorically defined as an interaction of a number system, (0, s), and as a model (a, h) of this number system. Hence, a comparison of NN_1 and NN_2 has to consider both of each system, the number system and the model system. Obviously, this kind of comparison is a comparison between the number systems as such and not a comparison of special cases belonging to the ultimate, i.e. *"the one and only one"* NN.

This is naturally generalized from number systems "NN" to *sign systems* "SS" in general.
Therefore, a simple modeling of the comparison shall happen in a polycontextural monoidal context. If we are considering the result of the comparison of SS_1 and SS_2 the comparison is producing super-additively a new SignSystem SS_3 with the inscription of the comparison as such. Obviously, the mechanism of comparison is not an isomorphism between a sign system and its 'model' or a functorial mapping but a mechanism of mediation with its super-additivity between contexturally different systems.

What's the matter with this tedious formula? If we want to compare two machines, for convenience, two trivial machines based on the concatenation of signs, it is the first thing we should know: How are they defined in regard to their behavior, in contrast to their set-theoretical properties? The answer, at least a first step to it, is constructed with the help of polycontextural monoidal categories as shown below.

All that is, without doubt, simply a beginning to get some *orientation* and *directions* for further studies.

$$\mathcal{U}^{(3)} = \left(\mathcal{U}_1 \amalg_{1.2.0} \mathcal{U}_2\right) \amalg_{1.2.3} \mathcal{U}_3,$$

$$\mathcal{U}_1 \bigcap \mathcal{U}_2 \bigcap \mathcal{U}_3 = \emptyset$$

$$\text{SignSys} = \left(\text{N}, 0, \text{s}\right), \ \text{Model} = \left(\text{A}, \text{a}, \text{h}\right)$$

$$\left[\mathcal{U}_i = \left\{\text{SignSys}_i, \text{Model}_i\right\}, \ i = 1, 2, 3\right] :$$

$$\left(\begin{array}{c} \left(\begin{array}{c} \left(\text{SignSys}_1 \circ_{1.0.0} \text{Model}_1\right) \\ \amalg_{1.2.0} \\ \left(\text{SignSys}_2 \circ_{0.2.0} \text{Model}_2\right) \end{array}\right) \\ \amalg_{1.2.3} \\ \left(\text{SignSys}_3 \circ_{0.0.3} \text{Model}_3\right) \end{array}\right) = \left(\begin{array}{c} \left(\begin{array}{c} \text{SignSys}_1 \\ \amalg_{1.2.0} \\ \text{SignSys}_2 \end{array}\right) \\ \amalg_{1.2.3} \\ \text{SignSys}_3 \end{array}\right) \circ_{1.2.3} \left(\begin{array}{c} \left(\begin{array}{c} \text{Model}_1 \\ \amalg_{1.2.0} \\ \text{Model}_2 \end{array}\right) \\ \amalg_{1.2.3} \\ \text{Model}_3 \end{array}\right)$$

1.2.2. Mediated trivial machines

Also the scheme of interchangeability is rather simple, it allows a quite complex combinatorics in its applications. A few example shall be exposed. Nevertheless, these exercises and the examples above, are just a first step towards a diamond-theoretic formalization and are not yet demonstrating much of this new approach to modeling and formalization as prerequisite of implementation and emulation of new ways of computing. Especially the symmetric relation between system and model has to be questioned and extended to an interplay of symmetric and asymmetric relations. The reasons to study such abstract relationships is motivated by the requirements of a theory of multi- and poly-layered crossbar constructions for memristive systems.

Composition of two sign systems

$\text{SignSys} = (N, 0, s)$, $\text{Model} = (A, a, h)$

$\mathcal{U}_2 = \{\text{SignSys}_2, \text{Model}_2\}$

$\mathcal{U}_1 = \{\text{SignSys}_1, \text{Model}_1\}$

$\mathcal{U}_1 \bigcap_{1.2} \mathcal{U}_2 = \emptyset$

$\mathcal{U}^{(2)} = \mathcal{U}_1 \amalg_{1.2} \mathcal{U}_2$:

$$\begin{bmatrix} \text{Model}_1 & \text{Model}_2 \\ \text{SignSys}_1 & \text{SignSys}_2 \end{bmatrix} :$$

$$\begin{pmatrix} \text{Model}_1 \\ \amalg \\ \text{SignSys}_1 \end{pmatrix} \circ \begin{pmatrix} \text{Model}_2 \\ \amalg \\ \text{SignSys}_2 \end{pmatrix} = \begin{pmatrix} \text{Model}_1 & \circ & \text{Model}_2 \\ & \amalg & \\ \text{SignSys}_1 & \circ & \text{SignSys}_2 \end{pmatrix}$$

Mutual cross-exchange of sign systems and models

$\mathcal{U}_2 = \{\text{SignSys}_2, \text{Model}_2\}$

$\mathcal{U}_1 = \{\text{SignSys}_1, \text{Model}_1\}$

$\mathcal{U}_1 \bigcap_{1.2} \mathcal{U}_2 = \emptyset$

$\mathcal{U}^{(2)} = \mathcal{U}_1 \amalg_{1.2} \mathcal{U}_2$:

$$\begin{bmatrix} \text{Model}_1 & \text{Model}_2 \\ \text{SignSys}_1 & \text{SignSys}_2 \end{bmatrix} :$$

$$\begin{pmatrix} \text{Model}_1 \\ \amalg \\ \text{SignSys}_1 \end{pmatrix} \lozenge \begin{pmatrix} \text{Model}_2 \\ \amalg \\ \text{SignSys}_2 \end{pmatrix} = \begin{pmatrix} \text{Model}_1 & \circ & \text{Model}_2 \\ & \square & \\ \text{SignSys}_1 & \circ & \text{SignSys}_2 \end{pmatrix}$$

$\amalg$: mediation between contextures
$\circ$: composition of morphisms
$\lozenge$: cross-interchange between levels
$\square$: $(\lozenge, \amalg)$, mediated cross-interchange
$=$: equivalence

Cross-Iteration
Composition of three sign systems

$\text{SignSys} = (N, 0, s)$, $\text{Model} = (A, a, h)$

$\mathcal{U}_i = \{\text{SignSys}_i, \text{Model}_i\}$, $i = 1, 2, 3$

$\mathcal{U}_1 \bigcap_{1.2} \mathcal{U}_2 \bigcap_{2.3} \mathcal{U}_3 = \emptyset$

$\mathcal{U}^{(3)} = (\mathcal{U}_1 \amalg_{1.2} \mathcal{U}_2) \amalg_{1.2.3} \mathcal{U}_3$:

$$\begin{bmatrix} \text{Model}_1 & \text{Model}_2 & \text{Model}_3 \\ \text{SignSys}_1 & \text{SignSys}_2 & \text{SignSys}_3 \end{bmatrix} :$$

$$\begin{pmatrix} \text{Model}_1 \\ \amalg \\ \text{SignSys}_1 \end{pmatrix} \lozenge \begin{pmatrix} \text{Model}_2 \\ \amalg \\ \text{SignSys}_2 \end{pmatrix} \lozenge \begin{pmatrix} \text{Model}_3 \\ \amalg \\ \text{SignSys}_3 \end{pmatrix}$$

$$\begin{pmatrix} \text{Model}_1 & \circ & \text{Model}_2 & \circ & \text{Model}_3 \\ & \lozenge & & \lozenge & \\ \text{SignSys}_1 & \circ & \text{SignSys}_2 & \circ & \text{SignSys}_3 \end{pmatrix}$$

Accretion
Composition of two mediated sign systems

$$\left(\begin{array}{c}\left(\begin{array}{c}\text{Model}_{2.2}\\ \amalg_2\\ \text{SignSys}_{2.1}\end{array}\right)_2\\ \amalg_{1.2}\\ \left(\begin{array}{c}\text{Model}_{1.2}\\ \amalg_1\\ \text{SignSys}_{1.1}\end{array}\right)_1\end{array}\right) \circ_{1.2} \left(\begin{array}{c}\left(\begin{array}{c}\text{Model}_{2.2}\\ \amalg_2\\ \text{SignSys}_{2.1}\end{array}\right)_2\\ \amalg_{1.2}\\ \left(\begin{array}{c}\text{Model}_{1.2}\\ \amalg_1\\ \text{SignSys}_{1.1}\end{array}\right)_1\end{array}\right) =$$

$$\left(\begin{array}{c}\left(\left(\begin{array}{c}\text{Model}_{2.2}\\ \amalg_2\\ \text{SignSys}_{2.1}\end{array}\right) \circ_2 \left(\begin{array}{c}\text{Model}_{2.2}\\ \amalg_2\\ \text{SignSys}_{2.1}\end{array}\right)\right)_2\\ \amalg_{1.2}\\ \left(\left(\begin{array}{c}\text{Model}_{1.2}\\ \amalg_1\\ \text{SignSys}_{1.1}\end{array}\right) \circ_1 \left(\begin{array}{c}\text{Model}_{1.2}\\ \amalg_1\\ \text{SignSys}_{1.1}\end{array}\right)\right)_1\end{array}\right)$$

1.2.3. Additivity vs. Super-additivity of compositions

A composition of finite state-machines is additive, linear and associative.
A combination of memristic systems, designed as polycontextural and morphogrammatic machines, is super-additive, tabular, and poly-associative.

Deterministic finite-state machine (DFA)

"DFAs are one of the most practical models of computation, since there is a trivial linear time, constant-space, online algorithm to simulate a DFA on a stream of input. Given two DFAs there are efficient algorithms to find a DFA recognizing:

* the *union* of the two DFAs
* the *intersection* of the two DFAs
* *complements* of the languages the DFAs recognize.

DFAs are equivalent in computing power to nondeterministic finite automata." (WiKi)

Polycategorical concatenation of automata

$$\mathcal{U}^{(3)} = \left(\mathcal{U}_1 \amalg_{1.2.0} \mathcal{U}_2\right) \amalg_{1.2.3} \mathcal{U}_3,$$

$$\left(\mathcal{U}_1 \bigcap\nolimits_{1.2} \mathcal{U}_2\right) \bigcap\nolimits_{1.2.3} \mathcal{U}_3 = \emptyset$$

$$\text{DFA}_{i.1} = \left(Q, \Sigma, \delta, q_0, F\right)_{i.1},$$

$$\text{DFA}_{j.2} = \left(Q, \Sigma, \delta, q_0, F\right)_{i.2}, \quad i = 1, 2, 3$$

$$\mathcal{U}_i = \left\{\text{DFA}_{i.1}, \text{DFA}_{i.2}\right\}, \quad i = 1, 2, 3:$$

$$\left(\begin{array}{c}\left(\begin{array}{c}\left(\text{DFA}_{1.1} \circ_{1.0.0} \text{DFA}_{1.2}\right)\\ \amalg_{1.2.0}\\ \left(\text{DFA}_{2.1} \circ_{0.2.0} \text{DFA}_{2.2}\right)\end{array}\right)\\ \amalg_{1.2.3}\\ \left(\text{DFA}_{3.1} \circ_{0.0.3} \text{DFA}_{3.2}\right)\end{array}\right) = \left(\begin{array}{c}\left(\begin{array}{c}\text{DFA}_{1.1}\\ \amalg_{1.2.0}\\ \text{DFA}_{2.1}\end{array}\right)\\ \amalg_{1.2.3}\\ \text{DFA}_{3.1}\end{array}\right) \circ_{1.2.3} \left(\begin{array}{c}\left(\begin{array}{c}\text{DFA}_{1.2}\\ \amalg_{1.2.0}\\ \text{DFA}_{2.2}\end{array}\right)\\ \amalg_{1.2.3}\\ \text{DFA}_{3.2}\end{array}\right)$$

1.2.4. Products of DFAs

Products of DFAs are of special interests for a theory of atomata.

1.3. Memristors as non-trivial machines

1.3.1. History-dependence

Obviously, memristors are not representing trivial machines because their *"resistance depends on the internal state of the system"*, while trivial machines, mono-contextural as well as 'polycontextural', are *"history independent"*.
Also memristors have a strict distinction of input and output functionality, their out-put is depending not only on the input and the definition of the function but also on the history of the former input/output relation. Hence they are *"history dependent"*.
Are memristors therefore non-trivial?
Trivial machines have additionally to their history-independence a "predictable" behavior, non-trivial machines behave "unpredictable". Is this true for memristors? If yes, in which sense is the behavior of memristors unpredictable?
The behaviors of memristors is interpretable in two distinct ways: 1. as digital, and 2. as analog.

Unpredictability
"Ionized atomic degrees of freedom *define the internal state of the device.*''
History
"New possibilities in the understanding of neural processes using memristive memory devices whose response depends on the whole dynamical history *of the system."* (Kim, New Scientist, 2009)

Further characterizations of *"time- and history-dependence"* of memristive behavior and its modeling by kenomic operators gets strong support by my previous studies towards a new paradigm of computation and a *"Theory of Living Systems"*, making use of retro-grade iter/alterability. [4]

In other words, repetition, i.e. iterability as retrograde recursivity is involved in *self-referentiality*, *transparency*, *memory* and *history*, and *evolution* of objects (morphograms). Until now, only the retro-grade and self-referential aspect of time- and memory-depending actions in memristive systems had been in focus.

Di Ventura et al, *Putting memory into circuit elements: memristors, memcapacitors and meminductors*
"Equivalently, the memristor relates the current to the voltage, but unlike its traditional counterpart, its resistance, upon turning off the power source, depends on the integral of its entire past current waveform. In other words, it has memory of past states through which the system has evolved. [...]
A simple reason for this is that at the nanoscale the dynamical properties of electrons and ions strongly depend on the history of the system, at least within certain time scales. Therefore, many devices at these length scales retain partial memory of the electron and ion dynamics."
http://physics.ucsd.edu/~diventra/PointofViewDPC6.pdf

It seems that a lot of the wordings and conceptual developments for 'memristive' systems has been repeatedly done long ago. What is new today is a better mathematical apparatus (polycontextural category and diamond theory) and, crucially, the discovery of memristive behaviors in nanoelectronics (memristor, memdevices).

Memristive properties are depending, according to Di Ventura, "on the integral of its *entire past* current waveform", "has *memory of past* states through which the *system* has evolved", "*history of the system*, at least within certain time scales" and therefore "retain partial *memory* of the electron and ion dynamics".

In this study I will restrict myself to the aspects of *self-referentiality* and morphogrammatic *evolution* of minimal time- and history-dependent memristive systems as it appears as retro-grade and evolutive patterns of morphogrammatics, therefore omitting aspects of *positionality*, *locality* and *transparency*. Aspects of *transparency* are connected with the autonomy of a system as a whole. Transparency of a computational system is not reasonably accessible for information processing systems. It needs an additional abstraction from the complexity of informational processes offered by a morphic abstraction.
Together with the mentioned features, the fact that nano-devices are *localized*, i.e. are taking a position in a positional matrix, has to be emphasized too. System-theoretic notions are not covering features (or principle) of localization and positionality.

Memristors are in fact "*assemblies of nanoparticles*" (Kim, 2009), therefore, their behavior is not "analytically" pre-defined, their behavior, depending on the contextual history of the system, i.e. the position in a memristive matrix, has to be interpreted. This might happen as an abstraction of identity, producing a predictable binarity of values, or it might be interpreted analogously, producing a non-predictable set of values.
The concept of a single or a collection of memristors as nano-technological devices has therefore its machine-theoretic description by the concept of a non-trivial machine.

5

Because functional complete logical functor-sets are representing trivial machines, the characterization of memristors as non-trivial machines might be in conflict with the understanding of memristors as *material implications* together with the functional completeness of a logic with implication and a negative constant, {IMP, F}. Complete junctional sets in logic theories are decidable, hence, their behavior is predictable.
This characterization by logically complete sets is only halve the story of the possible behaviors of memristors.

Notes from the *Variety*

A finite state machine has a state but not a memory of a state.
A memristive machine has a state of a state, i.e. a *meta-state* as a memory, therefore a memristic machine is not a finite state machine.
A meta-state always can be taken as a simple state in the sense that a reduction from an as-abstraction to an is-abstraction is directly possible because the necessary informations are stored in the meta-state. From *"x as y is z"* there is an easy way to *reduce* it to *"x is x"*. Such a reduction of a second-order system to a first-order system is nevertheless losing the essential features of the reduced system.

A memristive machine, then, is a machine with a tensed time, while finite-state machines are not tensed machines. Their temporality is of first-order, memristic time is of second-order, i.e. an interpretation of a state of a state.
Todays interpretation of memristors as memory devices in an ANN is reducing the possibility of second-order learning to simple first-order learning as trained adaption.

Memristors and states
In a further analysis the conceptual levels of the constructions become more clear:
1. Zero-level : the RIC-elements are of zero-level because they don't have a state.
2. First-level as machines: States of a state-machine based on RIC-elements are elementary states.
3. First-level as elements: Memristors, or mem-elements, are devices with a state. Hence they replace first-level finite-state machines.
4. Second-level: States of memristive machines are states that have a state, i.e. a memrory of a state.

Hence, RIC-elements have no state, mem-elements *have* a state and mem-machines have a state of a state.

Memristive iterability
Therefore, an iteration of an electronic action for non-memristive devices is a history-independent action. While a repetition of an action for a memristitive device is history-dependent and therefore changing its character in time. Non-,meristive repetitions are conceptually well modeled by arithmetic or semiotic successor operations, i.e. by concatenation based on a pre-given alphabet. Repetition in non-memristive systems is stable, i.e. monoton.
Memristive iterability is modeled by kenomic operators of change based on monomorphies. Here again, iterability is alterability too.
The complexity of time- and history-dependence for memristors is very minimal but significantly different from semiotic concatenation. Each repetition or each new run might change the dynamics between first- and second-order characteristics in memristive devices.

6

Morphogrammatics offers a complex theory of history-dependend changes of morphograms, i.e. of iter-/alterability.[6]

1.3.2. Kenomic modeling

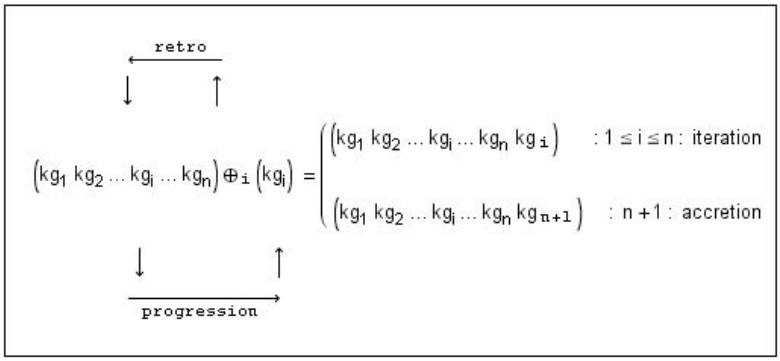
In contrast to semiotic concatenation, kenomic evolution has to be considered as retro-grade, depending on the history of its occurrences. As developed before in several papers, kenomic evolutions are not framed by *initial* and *final* (terminal) objects.

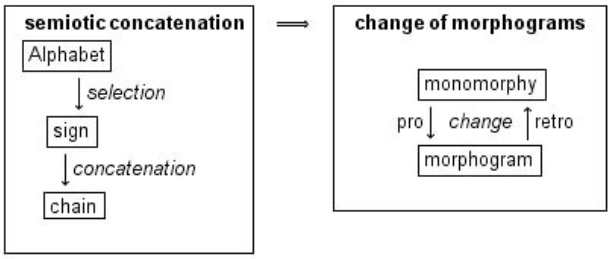
"In contrast to morphogramatic evolution, kenogrammatic 'concatenation' still relies to some degree on the *linear* order of its kenoms. But there is no need anymore for a pre-given alphabet, and concatenation itself is only one of the elementary operations of change. Further operations are *chaining* and different kinds of *fusion*. Without a pre-given alphabet the risk has to be taken to develop change out of the encountered kenogram sequences only. With that the abstractness of the semiotic concatenation is surpassed. There is not only no alphabet given, but the kenoms involved are semiotically indistinguishable. The operation of concatenation is defined by an *interaction* with the encountered kenogram sequence. Its range is determined by the occurring kenoms of the sequence which remains itself still untouched by the process of concatenation. Hence, kenogrammatic concatenation is not defined in an abstract way but *retro-grade* to the encountered kenomic pattern." (Kaehr, Morphogrammatics of Change)

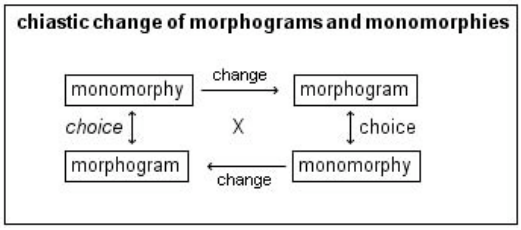
Such *retro-grade* recursion to develop kenomic progressions is a necessary condition to develop a *"history-dependent"* mechanism of change on a pre-semiotic level of inscription without the presumption of a pre-given sign-repertoire and its restrictions of atomicity, linearity and identity.

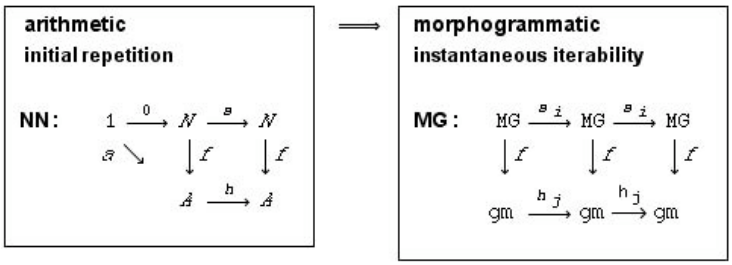
Hence, what is repeated and involved at new into a calculation is not a data from an external (re)source (sign repertoire, environment) but the result of a former mark (activity), which is *remembered* (retained) at that locus. Therefore, the repeated mark is a memorized constellation of a former activity. It is thus a mark of a mark what defines a history-dependent mark.
As a consequence, an external observer can't predict the outcome of a kenomic concatenation. The concepts of input and output are losing their relevance. Hence, the lack of predictability is not to confuse with a gain of stochastics or propabilistics of non-deterministic machines.

1.3.3. Morphogrammatic prolongations









First aspect: iteration
Given a morphogram MG, which is always a localized pattern in a kenomic matrix, a *prolongation* (successor, evolution) of the morphogram is achieved with the successor operator s_i. To each prolongation a further prolongation is defined by the iterated application of the operator s_i.

The morphogrammatic succession $\left(MG \xrightarrow{s_i} MG\right)$ is founded by its model $\left(gm \xrightarrow{h_j} gm\right)$ and the morphism *f*, guaranteeing the commutativity of the construction.
As a third rule, the iterability of the successor operation is *arbitrary,* which is characterised by the commutativity of the diagram. Hence, the conditions for a (retrograde) recursive formalisation are given.

Second aspect: anti-dromicity
Each prolongation is realized simultaneously by an iterative *progression* and an *antidromic retro-gression*. That is, the operation of prolongation of a morphogram is defined retro-grade by the possibilities given by the encountered morphogram. A concrete prolongation is selecting out of those possibilities its specific successions. All successions are to be considered as being realized at once.

Third aspect: simultaneity and interchangeability
This simultaneity of different successions defines the range of the prolongation. This definition of morphogrammatic prolongation is not requiring an alphabet and a selection of a sign out of the alphabet. Hence, the concept of morphogrammatic prolongation is defined by the two aspects of iteration and antidromic retro-gradeness of the successor operation. The simultaneity of the prolongations is modeled by the interchangeability of its actions.

Fourth aspect: diamond characterization of antidromicity
Both aspects together, repeatability and antidromicity with its simultaneous and interchangeable realizations, are covered by the diamond-theoretic concept of combination of operations and morphisms, i.e. composition and saltisition, between morphogramatic prolongations.

Diamond of the MG – successor rule

$$\frac{\left(MG \xrightarrow{s_j} MG\right) \Diamond \left(MG \xrightarrow{s_j} MG\right)}{\left(MG \xrightarrow{s_i} MG\right) \circ \left(MG \xrightarrow{s_j} MG\right) \;\middle|\; \bar{MG} \xleftarrow{\bar{h}} \bar{MG}}$$

Diamond of the MG – successor system

$$\begin{array}{ccccc} [MG] & \xrightarrow{s_i} & \{MG\} & \xrightarrow{s_i} & \{MG\} \;\middle|\; \bar{MG} \xleftarrow{\bar{s}} \bar{MG} \\ \downarrow f & & \downarrow f & & \downarrow f \qquad \uparrow f \\ [gm] & \xrightarrow{h_j} & \{gm\} & \xrightarrow{h_j} & \{gm\} \;\middle|\; \bar{gm} \xleftarrow{\bar{h}} \bar{gm} \end{array}$$

Diamond MG – successor system

$$\begin{array}{ccc} [\{MG\} & \xrightarrow{s_i} & \{MG\}] \;\middle|\; \bar{MG} \xleftarrow{\bar{s}} \bar{MG} \\ \downarrow f & & \downarrow f \qquad \uparrow f \\ \{gm\} & \xrightarrow{h_j} & \{gm\} \;\middle|\; \bar{gm} \xleftarrow{\bar{h}} \bar{gm} \end{array}$$

2. Morphogrammatics

2.1. A new framework for memristics

A new framework of design and analysis shall be proposed.

Properties of retro-gradeness (antidromicity), self-referentiality, simultaneity and locality (positionality) of operations as they occur in kenogramamtic and morphogrammatic basic operations, like the successor operation (function) have to be realized on all levels of operativity in memristive systems.

Hence, the tiny memristive properties of time- and history-dependence for kenomic successors have to be developed for all further operations, like “addition" (coalition), "multiplication”, “reflection”, etc.

Morphogrammatics, Diamond Category Theory, Diagrammatics, Memristics, Nanotechnology.

2.1.1. Electronics

Monotony

Each activity in a non-memory-dependent system happens, iterates, successively without being reflected by prior activities in a linear monotony.
A run through a RCL-circuit might iteratively be repeated, ideally, without changing its predefined conditions.

Composition of networks
Behaviors of RCL-circuit networks are structurally based on serial and parallel compositions.

An important consideration in non-linear analysis is the question of *uniqueness*. For a network composed of linear components there will always be one, and only one, unique solution for a given set of boundary conditions. This is not always the case in non-linear circuits.

2.1.2. Memristics

Antidromic iteration

Each activity in a memory-dependent system happens, iterates or accreates, successively in strict dependence of the prior activities of the whole circuit.
A run through a RCL-circuit is iteratively reflecting or learning, ideally, its new and changing conditions, i.e. states of procedurality.
Hence, the retro-grade activity of the progression might be well modeled with the basic structures of kenomic and morphogrammatic operations.

Combinations, compositionality
Behaviors of RCL-circuits are structurally based on *serial* and *parallel* compositions. What are the memristic analogies? Coalitions, combinations and products of morphograms might model features of memristive compositionality. Such compositionality is understood as radically memristive and shall not be reduced to the topic of applications of memristors as elements in classical circuits.
Thus, the logical interpretation of memristors as a material implication is just a start and is not yet reflecting on the holistic patterns of memristive behaviours.

If memristors are intepreted in the memristic framework as second-order elements, a change of metaphors could help to develop a simple theory of series and parallel compositions for memristors. Instead of “second-order’ elementsis shall be understood as 2-layered basic memristive elements.

Hence, by diamondization, the rules for series and parallel compositions for memristors might be added to the classical rules as their second layer, i.e. as their structural environment. The daisy chain-coupled circuits gets coupled with its neighbors.

Structured antidromicity
As a new step in the modeling of memristive activities by the application of morphogrammatics it seems to be necessary to understand the change of memristance (memductance, memcapacitance) as a *structured* process. Hence, patterns of second-order features have to be studied in memristics. Until know, this appears as a speculation because there are not yet any experimental results and descriptions available about such a topic.

Presupposing the existence of structured memristive antidromicity as a pattern of second-order activities, a morphogrammatic modeling in the framework of *monomorphies* follows quite naturally as a consequence of the retro-grade and holistic structure of morphograms.

Framework of a memristic research program

$$\text{Memristics} = \begin{pmatrix} \text{Morphogrammatics} & \longleftrightarrow & \text{poly} - \text{Category} \\ \updownarrow & \times & \updownarrow \\ \text{Memristive systems} & \longleftrightarrow & \text{Diagrammatics} \end{pmatrix}$$

2.2. A very first approximation for non-structured memristive elements

By diamondization, the rules for series and parallel compositions of memristors might be added to the classical rules of series and parallel compositions as their second layer, i.e.as their structural environment, covered formally by diamond-theoretic notions. Again, these are conceptual speculations, first attempts to understand memristive systems and their possible technology.

Again, a memristor *has* a state,that means, a memristor *is* not a state.
This says it clearly enough that a memristor is no a state. Its behavior is not characterized to be a state but to *have* a state. This as a result of an interaction with the memristor.
What means *having* a state in the case of a memristor?
It was shown clearly enough that a memristor is a specific resistor having a state. That’s why it is described as resistor with memory, called memristor.

If a memristor is to be described or defined by its resistance and in parallel by its memory of that resistance, we obviously have to deal with a device that has two functionalities. One as a resistor, another as a memory of that resistance.

It is clear too, that such a characterization as a two-fold functionality is not properly understood as a *superposition* of one function over the other. It is not simply a memory of a resistance or a resistance of a memory. Such a successive and hierarchical interpretation is useful for a simulation of the behavior of a memristor only. But it is not adequate to describe the characterization of the behavior of a memristor as such.

A proper understanding of a memristor as having a state has to take into account the simultaneity of both functions: the resistance and the memristance of the memristor. Both are defined as two levels or layers of a

second-order device. Such a simultaneity and parallelism puts both levels into a heterarchical yuxtaposition.

'Memristance is a property of an electronic component. If charge flows in one direction through a circuit, the resistance of that component of the circuit will increase and if charge flows in the opposite direction in the circuit, the resistance will decrease. If the flow of charge is stopped by turning off the applied voltage, the component will 'remember' the last resistance that it had, and when the flow of charge starts again the resistance of the circut will be what it was last active."
"In other words, a memristor is 'a device which bookkeeps the charge passing its own port'" (Stanley Williams)

"v = R[q(t)]i
The meaning of this equation is that the charge flowing through the memristor dynamically changes the internal state of the memristor making it a nonlinear element."

Again, bookkeeping is a parallel activity, happening simultaneously to the incoming 'bookings'.

2.2.1. Phenomenological description

change of flow $(\text{resistor})_1 \longrightarrow$ recording new state $(\text{storage})_1$
$\downarrow \qquad\qquad \downarrow$
change of flow $(\text{resistor})_2 \longleftarrow$ retrieving kept state, $(\text{memory})_1$
$\downarrow \qquad\qquad \downarrow$
change of flow $(\text{resistor})_2 \longrightarrow$ recording new state $(\text{storage})_2$
$\downarrow \qquad\qquad \downarrow$
change of flow $(\text{resistor})_3 \longleftarrow$ retrieving kept state, $(\text{memory})_2$

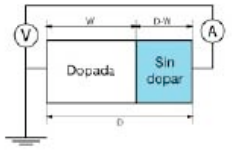
$$\text{memristor} = \text{chiasm}(\text{change}, \text{resistance}, \text{recording}, \text{retrieving})$$

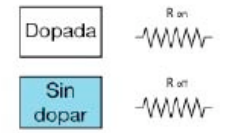
$$\text{memristor} = \begin{bmatrix} (\text{change}) & \longrightarrow & (\text{resitance}) \\ \updownarrow & X & \updownarrow \\ (\text{retriving}) & \longleftarrow & (\text{recording}) \end{bmatrix} = \begin{bmatrix} (\text{resistance}) & \xrightarrow{\text{changing}} & (\text{state}) \\ \updownarrow \text{iteration} & X & \updownarrow \text{receiv} \\ (\text{memory}) & \xleftarrow{\text{retriving}} & (\text{storage}) \end{bmatrix}$$

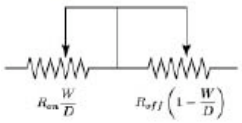
The phenomenological result corresponds the structuration pattern " *proemiality* " (open chiasm):

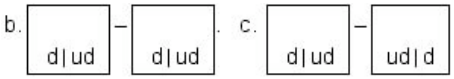
$$\boxed{\begin{matrix} A \longrightarrow B \\ X \\ D \longleftarrow C \end{matrix}} \Longrightarrow \boxed{\begin{matrix} \text{state} \longrightarrow \text{record} \\ X \\ \text{state} \longleftarrow \text{retrival} \\ X \\ \text{record} \longrightarrow \text{retrival} \end{matrix}} \Longrightarrow \boxed{\begin{matrix} \text{state} \longrightarrow \text{record} \\ \Updownarrow \\ \text{retrival} \longrightarrow \text{state} \\ \Updownarrow \\ \text{record} \longrightarrow \text{retrival} \end{matrix}}$$

2.2.2. Physical description



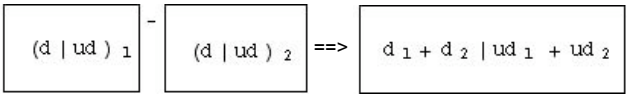




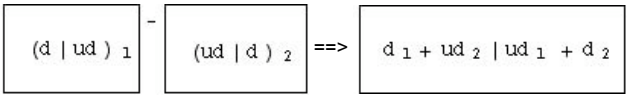


"FIG. 2:
a: Schematic of a memristor of length *D* as two resistors in series. The doped region (TiO_{2-x}) has resistance $R_{ON}w/D$ and the undoped region (TiO_2) has resistance $R_{OFF}(1-w/D)$. The size of the doped region, with its charge +2 ionic dopants, changes in response to the applied voltage and thus alters the effective resistance of the memristor.
b: Two memristors with the same polarity in series. *d* and *ud* represent the doped and undoped regions respectively. In this case, the memristive effect is retained because doped regions in both memristors simultaneously shrink or expand.
c: Two memristors with opposite polarities in series. The net memristive effect is suppressed."
Yogesh N. Joglekar and Stephen J. Wolf, The elusive memristor: properties of basic electrical circuits, 2009
http://arxiv.org/pdf/0807.3994v2

b. M + M = M



c. M + M = R



A physical description of a memristor is not yet the description of its behavior. A memristor defined as a programmable non-volatile double-resistor device is not yet describing its behavioral history-dependency. This quality is mostly mentioned after the physical description of a memristor as an appendix, i.e.an additional interpretation of the working of a physical memristor.

Again,

"The primary property of the memristor is the memory of the charge that has passed through it, reflected *in its effective resistance M(q).*" (Joglekar)

2.2.3. Some more concrete electronic modeling

1. Different polarity η

$$M_T(q) = \left(R_{0,1} - \eta \frac{\Delta R_1\, q(t)}{Q_0}\right) + \left(R_{0,1} + \eta \frac{\Delta R_2\, q(t)}{Q_0}\right) \qquad (1)$$

$$M_T(q) = \left(R_{0,1} + R_{0,2}\right) - \eta\left(\Delta R_1 - \Delta R_2\right) q(t) / Q_0 \qquad (2)$$

where:
– $R_{0,i}$ for $i = 1, 2$ is the effectie memristance of the memristor M_i at time $t = 0$;
– η is the polarity of the memrstor which can be + 1 or − 1;
– $\Delta R_i = R_{off,i} - R_{on,i} \simeq R_{off,i}$

Null

$$M = \left(R \middle| \overleftarrow{T}\right) \equiv (R, \Delta R)$$

$$M_T(q) = M_1 + M_2 \Longrightarrow M_1 + M_2 \neq M_2 + M_1$$

$$\left(R_{0,1} + R_{0,2}\right) - \eta\left(\Delta R_1 - \Delta R_2\right) q(t) / Q_0 \neq$$
$$\left(R_{0,2} + R_{0,1}\right) - \eta\left(\Delta R_2 - \Delta R_1\right) q(t) / Q_0$$

" It is clear that the overall behaviour of these memristors
depends on the relationship between ΔR_1 and ΔR_2 or simply $\alpha = \frac{\Delta R_1}{\Delta R_2}$ "

(F. Merrikh – Bayat, 24.08 .2010)

$\Delta R_i = R_{off,i} - R_{on,i} \simeq R_{off,i}$

$R_{off,i}$: maximum,

$R_{on,i}$: minimum

$\left(R_{0,1} + R_{0,2}\right) = \left(R_{0,2} + R_{0,1}\right)$

$\left(\Delta R_1 - \Delta R_2\right) \neq \left(\Delta R_2 - \Delta R_1\right)$

$\alpha = \frac{\Delta R_1}{\Delta R_2} \neq \frac{\Delta R_2}{\Delta R_1}$, i.e. $\alpha \neq \bar{\alpha}$.

2. Same polarity η

$M_r(q) = \left(R_{0,1} + R_{0,2}\right) - \eta\left(\Delta R_1 + \Delta R_2\right) q\left(t\right) / Q_0$

$M_r(q) = M_1 + M_2 \Longrightarrow M_1 + M_2 \neq M_2 + M_1$

$\left(\Delta R_1 + \Delta R_2\right) = \left(\Delta R_2 + \Delta R_1\right)$, but $\alpha \neq \bar{\alpha}$

Therefore, for same and different polarity :

$M_r(q) = M_1 + M_2 \Longrightarrow M_1 + M_2 \neq M_2 + M_1$.

In the following I will give a first complementary approach to the physical approach to define *conceptually* behavioral traits of memristors and memristive systems.

2.2.4. Memristive series and parallel circuits

Series

$M = \left(R \middle| \overleftarrow{r}\right)$: Memristor as a 2 – layered resistor with resistance R and $\overleftarrow{r}$ as retrograde memristance.

$R_{total} = R_1 + R_2 + \ldots + R_n$

$\left(R \middle| \overleftarrow{r}\right)_{total} = \left(R \middle| \overleftarrow{r}\right)_1 + \left(R \middle| \overleftarrow{r}\right)_2 + \ldots + \left(R \middle| \overleftarrow{r}\right)_n = M_{total}$

$$M_{total} = \left(R_1 + R_2 + \ldots + R_n\right) \middle| \left(\overleftarrow{r}_1 + \overleftarrow{r}_2 + \ldots + \overleftarrow{r}_n\right)$$

Memristive crossbars are constructed by series of memristors.

Parallel

$$\frac{1}{R_{total}} = \frac{1}{R_1} + \frac{1}{R_1} + \ldots \frac{1}{R_1}$$

For R_1 and R_2 :

$$M_{total} = \frac{\left(R \middle| \overleftarrow{r}\right)_1 \left(R \middle| \overleftarrow{r}\right)_2}{\left(R \middle| \overleftarrow{r}\right)_1 + \left(R \middle| \overleftarrow{r}\right)_2}$$

$$M_{total} = \frac{R_1 R_2 \middle| \overleftarrow{r}_1 \overleftarrow{r}_2}{R_1 + R_2 \middle| \overleftarrow{r}_1 + \overleftarrow{r}_2}$$

Diamondization of serial composition

$$\text{diam}\left(R_1 + R_2\right) = R_1 + R_2 \,\Big|\, \overleftarrow{r_4}$$

$$\text{diam}\left(M_1 + M_2\right) =$$

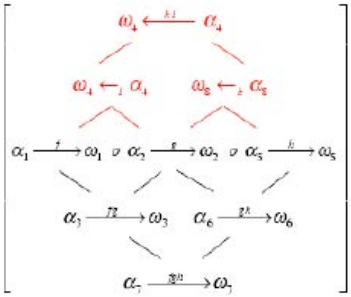
$$\text{diam}\left(\left(R\,\Big|\,\overleftarrow{r}\right)_1 + \left(R\,\Big|\,\overleftarrow{r}\right)_2\right) = \left(R\,\Big|\,\overleftarrow{r}\right)_1 + \left(R\,\Big|\,\overleftarrow{r}\right)_2 \,\Big|\, \overleftarrow{r_4}$$

$$\text{diam}\left(M_1 + M_2\right) = \left(R_1 + R_2 \,\Big|\, \overleftarrow{r}_1 + \overleftarrow{r}_2\right) \,\Big|\, \overleftarrow{r_4}$$

$$\text{diam}_{\text{ser}}\left(M_1 + M_2\right) = \left(R_1 + R_2 \,\Big|\, \overleftarrow{r}_1 + \overleftarrow{r}_2\right) \,\Big|\, \overleftarrow{r_4}$$

Series with saltitions

Numbering of diamond subsystems with jump-operation (saltisition ||) between subsystem 8 and 4. Number of subsystem (k, j) = 9



$$\text{diam}\left(M_1 + M_2 + M_3\right) = \left(R_1 + R_2 + R_3 \,\Big|\, \overleftarrow{r}_1 + \overleftarrow{r}_2 + \overleftarrow{r}_5\right) \,\Big|\, \left(\overleftarrow{r_4} + \overleftarrow{r}_8\right) \,\|\, \overleftarrow{r_9}$$

$$\text{diam}\left(M_1 + M_2\right) = \left(R_1 + R_2 \,\Big|\, \overleftarrow{r}_1 + \overleftarrow{r}_2\right) \,\Big|\, \overleftarrow{r_4}$$

$$\text{diam}\left(M_2 + M_3\right) = \left(R_2 + R_3 \,\Big|\, \overleftarrow{r}_2 + \overleftarrow{r}_5\right) \,\Big|\, \overleftarrow{r_8}$$

$$\Longrightarrow \text{diam}\left(M_1 + M_3\right) = \left(R_1 + R_3 \,\Big|\, \overleftarrow{r}_1 + \overleftarrow{r}_5\right) \,\Big|\, \overleftarrow{r_9}$$

$$\text{diam}\left(M_1 + M_2 + M_3\right) =$$

$$\left(R_1 + R_2 + R_3 \,\Big|\, \overleftarrow{r}_1 + \overleftarrow{r}_2 + \overleftarrow{r}_5\right) \,\Big|\, \left(\overleftarrow{r_4} + \overleftarrow{r}_8\right) \,\|\, \overleftarrow{r_9}$$

Diamodization of parallel composition

$$\text{diam}_{\text{par}}\left(M_{\text{total}}\right) =$$

$$\frac{R_1 R_2 \,\Big|\, \overleftarrow{r}_1\, \overleftarrow{r}_2}{R_1 + R_2 \,\Big|\, \overleftarrow{r}_1 + \overleftarrow{r}_2} \,\Big|\, \overleftarrow{r_4}$$

$$\text{diam}_{\text{par}}\left(M_{\text{total}}\right) =$$

$$\frac{M_1 M_2}{M_1 + M_2} \,\Big|\, \overleftarrow{r_4}$$

2.2.5. Arithmetic of linear electronic elements

It sounds probably strange to speak about an arithmetic of electronic elements.
Formally, series and parallel circuits, but also mixed combinations, are behaving arithmetically in the kind they are adding elements together. This is not surprising because electronic circuits are logically equivalent to restricted Boolean algebras.

Addition is *linear* (monotone)
$(((R_1 + R_1) + \ldots) + R_n)$

Idempotence
$R_1 + R_1 = R_1$

Succession of R's are following abstractly like natural number one the other without any need to relay on the previous succession. The same holds for a modeling with the concatenation operation on signs.

Addition is *commutative*
$R_1 + R_2 = R_2 + R_1$

Addition is *associative* and is not in any case involved into superadditivity:
$(R_1 + R_2) + R_3 = R_1 + (R_2 + R_3)$

2.2.6. Mono-structured memristive addition

Things are changing dramatically for non-linear additions of electronic elements, like memristors and other memdevices. Memristive addition is based on a retrograd reliance of the preceding events.

This might be modeled and studied as an addition of memristors or as a repeted application of a single memristor. What is of interest is the "*history-dependence*" of iteration, i.e. the retrograde character of succession. This kind of addition is super-additive and its value is retro-grade depending on the preceding state of the previous event.

Again,

"The memristor came later because it's inherently nonlinear. Why? A linear memristor is just a linear resistor, since we can differentiate the linear relationship p = Mq and get p' = Mq'. But if p' = Mq' for a nonlinear function f we get something new:

$$p' = f'(q)\, q'$$

So, we see that in general, a memristor acts like a resistor whose resistance is some function of q. But q is the time integral of the current q'. So a nonlinear memristor is like a resistor whose resistance depends on the time integral of the current that has flowed through it! Its resistance depends on its history. So, it has a "memory" - hence the name "memristance"." (John Baez)

Heterarchization of superposition

$p' = f'(q)\, q' :$

$$p' = \begin{pmatrix} f'(q) \\ f'(q)\, q' \end{pmatrix} \Longrightarrow \begin{pmatrix} f'(R) \\ f'(R)\, r' \end{pmatrix} \Longrightarrow \begin{pmatrix} M \\ (M, r) \end{pmatrix} = \text{Memristor}$$

$f'(q) : M$

$f'(q)\, q' : (M, r)$

$$p' = f'(q)\, q' \Longrightarrow \begin{pmatrix} f'(M) \\ f'((M)\, r) \end{pmatrix}$$

Non – Monotony

$$(((M_1 + M_1) + \ldots) + M_n) \Longrightarrow (((M_1 r_1 + (M_2 r_2) r_1 + \ldots + (M_n r_1) \ \ldots r_n)))$$

$$M_1 + M_1 = : M_1 + M_1 r_1$$

Non-Commutativity
Hence, commutativity is resolved and abolished for memristive addition.

$$M_1 + M_2 \neq M_2 + M_1$$

In words:
For $M_1 + M_2$:
M_1 is having a value "r_1", M_2 is having its value "r_2" too, but it depends additionally on the value of M_1.
$M_1 r_1 + M_2 r_2 = M_1 r_1 + (M_2 r_2) r_1$

For $M_2 + M_1$:
$M_2 r_2 + M_1 r_1 = M_2 r_2 + (M_1 r_1) r_2$

Therefore, a combination of memristors is replicating (retrieving, fetching) the inner state of the repeated memristor with the succiding memristor or the succeding memristive behavior. This reflects the history-dependence of memristive behaviours.

non – commutativity for memristive addition

$$M_1 r_1 + (M_2 r_2) r_1 \neq M_2 r_2 + (M_1 r_1) r_2$$

Special case:
$$\text{For } r_1 = r_2 \Longrightarrow M_1 + M_2 = M_2 + M_1 \; M_1 r_1 + (M_2 r_1) r_1 \neq M_2 r_1 + (M_1 r_1) r_1.$$

Non – Associativity

$$(M_1 + M_2) + M_3 \neq M_1 + (M_2 + M_3)$$

$$(M_1 r_1 + (M_2 r_2) r_1) + (M_3 r_3) r_2) r_1) \neq (((M_1 r_1) r_2) r_3) + (((M_2 r_2) r_3) + (M_3 r_3))$$

2.2.7. Categorical modeling as yuxtaposition and as mediation

Bifunctoriality of memristive addition with iteration

$$\text{BIFUNCT}^{(2)}_{\begin{bmatrix} r_1 & r_2 \\ M_1 & M_2 \end{bmatrix}} : \begin{pmatrix} M_1 \\ \otimes \\ M_2 \end{pmatrix} \circ \begin{pmatrix} r_1 \\ \otimes \\ r_2 \end{pmatrix} = \begin{pmatrix} M_1 \circ (r_1, r_2) \\ \otimes \\ M_2 \circ (r_2, r_1) \end{pmatrix}$$

$$\forall M, r \in \text{Universe } \mathcal{U}_i, i \in \mathcal{N}, \; i = 1$$

$$M_1 r_1 + M_2 r_2 = M_1 r_1 + (M_2 r_2) r_1$$
$$M_2 r_2 + M_1 r_1 = M_2 r_2 + (M_1 r_1) r_2$$

$$(M_1 \otimes r_1) \circ (M_2 \otimes r_2) = (M_1 \circ M_2) \otimes (r_1 \circ r_2) : \text{Bif}_{\text{sym}}$$

$$(M_1 \circ M_2) \otimes (r_1 \circ r_2) = (M_1 \otimes r_1) \circ (M_2 \otimes (r_2 \square r_1)) : \text{Bif}_{\text{asym}}$$

$$(M_1 \otimes r_1)\,[\circ\square]\,(M_2 \otimes r_2) = (M_1 \circ M_2) \otimes (r_1 \circ (r_2 \square r_1))$$

BIFUNCT $^{(2)}_{\begin{bmatrix} r_1 & r_2 \\ M_1 & M_2 \end{bmatrix}}$:

$$\begin{pmatrix} r_1 \\ \otimes \\ M_1 \end{pmatrix} [\circ\square] \begin{pmatrix} r_2 \\ \otimes \\ M_2 \end{pmatrix} = \begin{pmatrix} r_1 \circ (r_2 \square r_1) \\ \otimes \\ M_1 \circ M_2 \end{pmatrix}$$

$\forall$ M, r $\in$ Universe $\mathcal{U}_i$, i $\in \mathcal{N}$, i = 1

Interchangeability of memristive addition with replication

The formula of combination of memristors, $(M_1 \otimes r_1) \circ (M_2 \otimes r_2) = (M_1 \circ M_2) \otimes (r_1 \circ (r_2 \square r_1))$ gets a direct modeling with categorical notions.

A combination of memristors, $(M_1 r_1 + M_2 r_2)$, is replicating (retrieving, fetching), $(r_2 \square r_1)$, the inner state r_1 of the repeated memristor $(M_1 r_1)$ with the succeeding memristor $(M_2 r_2)$, i.e. the succeeding memristive behavior of $(M_2 r_2)$, resulting in the memristive combination $(M_1 r_1 + (M_2 r_2)\, r_1)$.

This is category-theoretically modeled with the composition operation ($\circ$) for the combination of memristors and the yuxtaposition ($\otimes$) for the two-leveled constitution of the memristors as (M r), while the retrieving (fetching) operation is modeled by the replication operation ($\square$) for $(r_2 \square r_1)$.

The modeling might happen in two modi of strictness:
- *first* a bifunctoriality with the yuxtaposition of two domains of a *multi-sorted* category with a single universe $\mathcal{U}$, and
- *second* by a mediation of *discontextural* domains $\mathcal{U}_{1,2,3}$ of a polycontextural category with a mediated polyverse $\mathcal{U}^{(3)}$ and super-additivity between $\mathcal{U}_1$ and $\mathcal{U}_2$.

1 – category with yuxtaposition for memristors

$$\text{BIFUNCT}^{(2)}_{\begin{bmatrix} M_2 & r_2 \\ M_1 & r_1 \end{bmatrix}} : \begin{pmatrix} r_1 \\ \otimes \\ M_1 \end{pmatrix} \circ\square \begin{pmatrix} r_2 \\ \otimes \\ M_2 \end{pmatrix} = \begin{pmatrix} r_1 \circ (r_2 \square r_1) \\ \otimes \\ M_1 \circ M_2 \end{pmatrix}$$

$\forall$ M, r $\in$ Universe $\mathcal{U}_i$, i $\in \mathcal{N}$, i = 1

$\mathcal{U}_1 = \mathcal{U}_{1.1} \bigcup \mathcal{U}_{1.2}$

A polycontextural modeling of the behavior of memristive devices is of importance, if the domains are not belonging to a common universe of objects. Instead, the discontectural domains are mediated, keeping the difference of both domains.

Super – additivity of a 3 – category with replication for memristors

$m = 3,\ n = 2$

$$\begin{bmatrix} r_1 & - & r_3 \\ M_1 & r_2 & - \\ - & M_2 & M_3 \end{bmatrix}:$$

$$\left(\begin{pmatrix} r_1 \\ \sqcup_{1.2.0} \\ M_1 \\ \sqcup_{1.2.3} \\ M_3 \end{pmatrix}\right)\begin{bmatrix} \circ_1 \ \square_1 & \\ & \circ_2 \\ \circ_3 \ \square_1 & \end{bmatrix}\left(\begin{pmatrix} r_2 \\ \sqcup_{1.2.0} \\ M_2 \\ \sqcup_{1.2.3} \\ r_3 \end{pmatrix}\right) = \begin{pmatrix} \begin{pmatrix} r_1 \ \circ_{1.0.0} \ (r_2 \square_1 r_1) \end{pmatrix} \\ \sqcup_{1.2.0} \\ \begin{pmatrix} M_1 \ \circ_{0.2.0} \ M_2 \end{pmatrix} \\ \sqcup_{1.2.3} \\ \begin{pmatrix} M_3 \ \circ_{0.0.3} \ r_3 \square_1 (r_2 \ r_1) \end{pmatrix} \end{pmatrix}$$

$\forall M, r \in \text{Universe}\ \mathcal{U}^{(3)} = \mathcal{U}_i, i \in \mathcal{N},\ i = 1, 2, 3$

$\mathcal{U}_1 \bigcap_{1.2} \mathcal{U}_2 \bigcap_{2.3} \mathcal{U}_3 = \phi$

$\mathcal{U}^{(3)} = (\mathcal{U}_1 \sqcup_{1.2} \mathcal{U}_2) \sqcup_{1.2.3} \mathcal{U}_3$

Chiasm

"The primary property of the memristor is the memory *of the charge that has passed through it,* reflected *in its effective* resistance *M(q)*." (Joglekar)

The conceptual modeling with chiasms is emphasizing the possibility of a continuation on a second-order level with the memorized value of the memristor. Hence, the double character of memristance as resistance and stored resistance, $(M \mid r)$, gets involved into a double functionality as a calculated and a calculating aspect.

It becomes clear that memristive behavior is not to be thematized on single memristors and their combinations but on memristive systems, i.e. memristive or memristic complexions, offering the conceptual and physical space for interchanging functionalities.
The following conceptual hint, which is ab/using some electronic formulas, shouldn't be confused with an engineering approach.
The chiasm is mediating two functionalities of a 2-complexion of interacting memristors.
One functionality is: $R_{0,1} \rightarrow \Delta R_1 <==> R_{0,2} <-> R_{0,1}$,
the other is: $R_{0,2} \rightarrow \Delta R2 <-> \Delta R_1 <==> R_{0,2}$,
both together are defining a feedback loop conceptualized as a chiasm between resistance and memristance of two instances.

This might be the place to promote the idea, again, that memristics should start with memristive complexions.
A single memristor is then a special case, separated and isolated from the memristive complexion.
Series and parallel circuits of memristors are not yet defining memristive complexions.

Hence, the minimal, still conceptual, conditions for memristive complexions might be defined by at least 3 memristive functionalities, 2 for the relation $\dashrightarrow$ and 1 for relation $\Longleftrightarrow$. It might be necessary to implement the cross-relation with 2 more memristors.

$\chi(M, r)$ ==> $\boxed{\begin{matrix} M_1 \longrightarrow r_1 \\ \times \ \Updownarrow \\ r_2 \longleftarrow M_2 \end{matrix}}$ ==> $\boxed{\begin{matrix} R_{0,1} \longrightarrow \Delta R_1 \\ \times \ \Updownarrow \\ \Delta R_2 \longleftarrow R_{0,2} \end{matrix}}$

$$M_\tau(q) = (R_{0,1} + R_{0,2}) - \eta(\Delta R_1 - \Delta R_2)\ q(t) \,/\, Q_0$$

$$\text{MemR}_1 = \left(M_1 \middle| r_1\right)$$

$$\text{MemR}_2 = \left(M_2 \middle| r_2\right)$$

$$\chi\left(\text{MemR}_1, \text{MemR}_2\right) = \begin{pmatrix} M_1 \circ r_1 \\ \Diamond \\ M_2 \circ r_2 \end{pmatrix} = \begin{pmatrix} M_1 \\ \Diamond \\ M_2 \end{pmatrix} \circ \begin{pmatrix} r_1 \\ \Diamond \\ r_2 \end{pmatrix}$$

$$M_\chi\left(q\right) = \chi\left(M\left(q\right)\right) = \left(M\left(q\right)_1, M\left(q\right)_2\right) = \begin{pmatrix} R_{0,1} \circ \Delta R_1 \\ \Diamond \\ R_{0,2} \circ \Delta R_2 \end{pmatrix} = \begin{pmatrix} R_{0,1} \\ \Diamond \\ R_{0,2} \end{pmatrix} \circ \begin{pmatrix} \Delta R_1 \\ \Diamond \\ \Delta R_2 \end{pmatrix}$$

Chiastic memristance

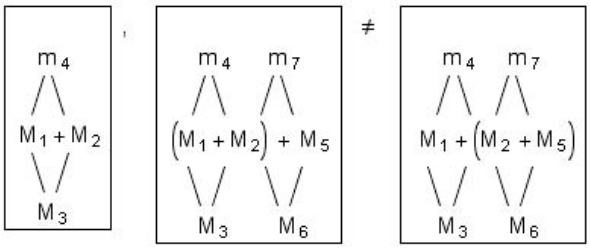
$$M_\chi\left(q\right) = \begin{pmatrix} R_{0,1} \circ \Delta R_1 \\ \Diamond \\ R_{0,2} \circ \Delta R_2 \end{pmatrix} = \begin{pmatrix} R_{0,1} \\ \Diamond \\ R_{0,2} \end{pmatrix} \circ \begin{pmatrix} \Delta R_1 \\ \Diamond \\ \Delta R_2 \end{pmatrix}$$

Revised formula:

$$M^2_\chi\left(q\right) = \chi\left(R_{0,1} + R_{0,2}\right) - \eta\left(\chi\left(\Delta R_1 - \Delta R_2\right)\right) q_{1,2}\left(t_{1,2}\right) / Q_{0\,1,2}$$

Diamondization

$$\text{diam}\left(M_1 + M_2\right) = \left(M_1 + M_2\right) \Big| m_4$$



2.3. Poly-structured memristive compositions

Not only memristors are in fact "*assemblies of nanoparticles*" (Kim, 2009) but also memristive behavior is observable as structured "assemblies". Therefore, construction and study of single memristors is selective and just the beginning of the adventure. Structured assemblies of memristive behaviors might be understood as complexions of memristors and their memristances. Hence, memristance of a memristive system is conceived more as a field of interacting memristive agents then as a circuit of memristive elements.

This paragraph intends to give a very first glance into the idea of compositions of poly-structured memristive complexions.

As the leading formal approach to modeling, monomorphy based morphogrammatics is applied.

Structuration: Iter/alterability of monomorphies
A combination of the retro-grade mechanism of prolongation (succession) with the complexity of the retro-grade prolongated memristive complexion demand for the monomorphy-based morphogrammatics.
Instead of kenomic iteration/accretion, prolongation is retro-grade defined on the monomorphies of morphograms.
Combinations of *structured* memristive systems are demanding for morphogrammatics and its study of monomorphies.

Technically, this seems to be a step further in the concretization of memristics, i.e. a step from the memristor as an element to memristive systems as complexions.

This topic will be developed in a next paper *'Morphogrammatics of Memristic Machines'*.

2.3.1. Monomorphic evolvement

The morphogram [aa] gets a memristic interpretation by $[M|r_1 r_1]$ or $[M|r_{1.1}]$, [ab] then corresponds to $[M|r_1 r_2]$.
The same holds for [aba] ==> $[M|r_1 r_2 r_1]$ and [aab] ==> $[M|r_1 r_1 r_2]$.

$$\text{evol}([\text{abb}]) \Longrightarrow \begin{pmatrix} [[\text{abb}],[a]] \\ \amalg_{1.2} \\ [[\text{abb}],[b]] \\ \amalg_{1.2.3} \\ [[\text{abb}],[c]] \\ \amalg \\ [[\text{abb}],[\text{aa}]] \\ \amalg_{1.2} \\ [[\text{abb}],[\text{bb}]] \\ \amalg_{1.2.3} \\ [[\text{abb}],[\text{cc}]] \end{pmatrix} = \begin{pmatrix} \text{evol}_a([\text{abb}]) = [\text{abba}] \\ \amalg_{1.2} \\ \text{evol}_b([\text{abb}]) = [\text{abbb}] \\ \amalg_{1.2.3} \\ \text{evol}_c([\text{abb}]) = [\text{abbc}] \\ \amalg \\ \text{evol}_{\text{aa}}([\text{abb}]) = [\text{abbaa}] \\ \amalg_{1.2} \\ \text{evol}_{\text{bb}}([\text{abb}]) = [\text{abbbb}] \\ \amalg_{1.2.3} \\ \text{evol}_{\text{cc}}([\text{abb}]) = [\text{abbcc}] \end{pmatrix}$$

The morphogram [abb] is build by two monomorphies [a] and [bb]. Hence, the operation of evolvement “evol” has to be applied kenogrammatically on both monomorphies, i.e. evol(mg_1, mg_2) = (evol(mg_1), evol(mg_2))

The result of an evolution of [abb] is not considered as a *set* of results but as a *simultaneity* of resulting evolutions.
The patterns [[abb] [a], [[abb] [b]] and [[abb] [c]] are seen as disjunct and simultaneously produced. They represent 3 different patterns, morphograms, i.e. [abba], [abbb] and [abbc] as mono-form resuslts of the evolvement of the morphogram [abb] with the monomorhy [a]. A further evolvement of [abb] is based on the second monomorphy [bb] of [abb] and is delivering the 3 morphograms [abbaa], [abbbb] and [abbcc].
There are no other constellations possible in this framework of morphogrammatic modeling of the *evolvement* of the morphogram [abb]. Other specifications of evolvement are defining different results.

Recall, morphograms are invariant patterns of kenograms. Therefore, semiotic representations of the morphogram [abb] as [baa] or as [acc] or as [#%%] etc. are morphogrammatically equivalent.
Again, this shows the retro-grade definition of operations in morphogrammatics, i.e. morphograms are defined as evolvements from themselves, and are not depending on a external alphabet.

The simultaneity of the results holds for the monomorphic modeling of the complex memristor $[M|r_1 r_2 r_2]$ too.

Thus, an evolvement of complex memristance produces simultaneous results.

What is offered by an evolvement as a structural multitude of possibilities has not always to be realized at once. Depending on other criteria, say from a context, a decision for a single or a subsystem of possibilities might be realized.
From a conceptual point of view, all possibilities have to be developed formally. Because of the holistic character of morphograms, changes of morphograms are always inherently finite.

$$\text{evol}\left(\left[M\middle|r_1 r_2 r_2\right]\right) \Longrightarrow$$

$$\begin{pmatrix} \left[M\middle|r_1 r_2 r_2\right],\left[M\middle|r_1\right] \\ \amalg_{1.2} \\ \left[M\middle|r_1 r_2 r_2\right],\left[M\middle|r_2\right] \\ \amalg_{1.2.3} \\ \left[M\middle|r_1 r_2 r_2\right],\left[M\middle|r_3\right] \\ \amalg \\ \left[M\middle|r_1 r_2 r_2\right],\left[M\middle|r_1 r_1\right] \\ \amalg_{1.2} \\ \left[M\middle|r_1 r_2 r_2\right],\left[M\middle|r_2 r_2\right] \\ \amalg_{1.2.3} \\ \left[M\middle|r_1 r_2 r_2\right],\left[M\middle|r_3 r_3\right] \end{pmatrix} = \begin{pmatrix} \text{evol}_{r_1}\left(\left[M\middle|r_1 r_2 r_2\right]\right) = \left[M\middle|r_1 r_2 r_2 r_1\right] \\ \amalg_{1.2} \\ \text{evol}_{r_2}\left(\left[M\middle|r_1 r_2 r_2\right]\right) = \left[M\middle|r_1 r_2 r_2 r_2\right] \\ \amalg_{1.2.3} \\ \text{evol}_{r_3}\left(\left[M\middle|r_1 r_2 r_2\right]\right) = \left[M\middle|r_1 r_2 r_2 r_3\right] \\ \amalg \\ \text{evol}_{r_{1.1}}\left(\left[M\middle|r_1 r_2 r_2\right]\right) = \left[M\middle|r_1 r_2 r_2 r_1 r_1\right] \\ \amalg_{1.2} \\ \text{evol}_{r_{2.2}}\left(\left[M\middle|r_1 r_2 r_2\right]\right) = \left[M\middle|r_1 r_2 r_2 r_2 r_2\right] \\ \amalg_{1.2.3} \\ \text{evol}_{r_{3.3}}\left(\left[M\middle|r_1 r_2 r_2\right]\right) = \left[M\middle|r_1 r_2 r_2 r_3 r_3\right] \end{pmatrix}$$

2.3.2. Monomorphic multiplicative coalitions

Also morphograms $MG_1 = [ab]$, $MG_2 = [ab]$ are morphogrammatically equivalent and consisting of 2 monomorphies [*a*] and [interesting operations are possible, one , *a* multiplication of MG_1 with MG_2 is shown.

$$MG_1 = \left[ab\right],\ MG_2 = \left[ab\right]$$

$$\text{kmul}\left(\left[ab\right],\left[ab\right]\right) = \begin{pmatrix} \text{mul}\left(\left[ab\right],\left[ba\right]\right) = \left[abba\right] \\ \amalg \\ \text{mul}\left(\left[ab\right],\left[bc\right]\right) = \left[abbc\right] \\ \amalg \\ \text{mul}\left(\left[ab\right],\left[ca\right]\right) = \left[abca\right] \\ \amalg \\ \text{mul}\left(\left[ab\right],\left[cd\right]\right) = \left[abcd\right] \end{pmatrix}$$

Multiplication tables for kmul$\left(\left[a, b\right],\left[a, b\right]\right)$

kmul	*a*	*b*
1	*a*	*x*
2	*b*	*y*

kmul	*a*	*b*	*b'*	*b"*	*b'"*
a	*a*	*b*	*b*	*c*	*c*
b	*b*	*a*	*c*	*a*	*d*

$$\text{kmul}\left(\begin{bmatrix} a \\ b \end{bmatrix}, \begin{bmatrix} a \\ b \end{bmatrix}\right) = \prod\left(\begin{bmatrix} a & b \\ b & a \end{bmatrix}, \begin{bmatrix} a & b \\ b & c \end{bmatrix}, \begin{bmatrix} a & c \\ b & a \end{bmatrix}, \begin{bmatrix} a & c \\ b & d \end{bmatrix}\right)$$

Context – Rule for monomorphic multiplication

$\forall\, i \in \mathrm{Dec}(\mathrm{MG}_{i+1}),\ \mathrm{mg}_i \in \mathrm{MG}_1,\ \mathrm{mg}_i \in \mathrm{MG}_2$:

$$\mathrm{kmul}(\mathrm{MG}_1,\ \mathrm{MG}_2) \text{ iff } \begin{pmatrix} \mathrm{head}(\mathrm{mg})_i \neq \mathrm{head}(\mathrm{mg})_{i+1} \\ \mathrm{body}(\mathrm{mg})_i \neq \mathrm{body}(\mathrm{mg})_{i+1} \end{pmatrix}$$

Head = first kenom of *a* monomorphy
Body = the rest.

Order

numerical lexical order(MG) = $1 < 2 < 3 < \ldots$

alphabetical lexical order(MG) = $a < b < c < \ldots$

Context rules: kmul$([a, b], [a, b])$

1. Identity: $[a]\,x\,[a, b] = [a, b]$: head – and body – iteration, $\mathrm{head}_1 = \mathrm{head}_2$, $\mathrm{body}_1 = \mathrm{body}_2$

2. Diversity: $[b]\,x\,[a, b] =$

$[b, a]$: iterative component,

$[c, a]$: head – accretion, body – iteration,

$[b, c]$: head – iteration, body – accretion,

$[c, d]$: head – and body – accretion,

all accepting CRM: $\mathrm{head}_1 \neq \mathrm{head}_2$, $\mathrm{body}_1 \neq \mathrm{body}_2$

A morphogram *MG* gets decomposed, *Dec*, into its monomorphies mg_i wich get placed at the loci loc_i.

Position $\mathrm{MG}^{(m,n)}$

$\mathrm{MG}^{(m)}$	locus
$\mathrm{Dec}(\mathrm{MG}^{(m)})$	monomorphy
$\mathrm{Ken}(\mathrm{MG}^{(m)})$	kenom

Positionality of morphograms: < Position, Locality, Place > .
Position of the morphogram in *a* morphogrammatic system defined by emanation and evolution.
Locality of the monomorphies in *a* morphogram; loci are offering place for different monomorphies.
Monomorphies might be reduced to homogeneous patterns or they might keep some structuration.
Place of *a* kenom in *a* monomorphy depending on the length of the monomorphy.

$\begin{matrix} a & b \\ b & a \end{matrix}$	$\mathrm{loc}_1\ \mathrm{loc}_2\ \mathrm{loc}_3\ \mathrm{loc}_4$
Dec	$\mathrm{mg}_1\ \mathrm{mg}_2\ \mathrm{mg}_3\ \mathrm{mg}_4$
Ken	$a\ \ b\ \ a\ \ \varnothing$ $\varnothing\ \ b\ \ \varnothing\ \ \varnothing$ $x\ \ y\ \ z\ \ u$

$\begin{matrix} a & b \\ b & c \end{matrix}$	$\mathrm{loc}_1\ \mathrm{loc}_2\ \mathrm{loc}_3\ \mathrm{loc}_4$
Dec	$\mathrm{mg}_1\ \mathrm{mg}_2\ \mathrm{mg}_3\ \mathrm{mg}_4$
Ken	$a\ \ b\ \ c\ \ \varnothing$ $\varnothing\ \ b\ \ \varnothing\ \ \varnothing$ $x\ \ y\ \ z\ \ u$

$\begin{matrix} a & c \\ b & d \end{matrix}$	$\mathrm{loc}_1\ \mathrm{loc}_2\ \mathrm{loc}_3\ \mathrm{loc}_4$
Dec	$\mathrm{mg}_1\ \mathrm{mg}_2\ \mathrm{mg}_3\ \mathrm{mg}_4$
Ken	$a\ \ b\ \ c\ \ d$ $\varnothing\ \ \varnothing\ \ \varnothing\ \ \varnothing$ $x\ \ y\ \ z\ \ u$

Non – commutativity

$\text{kmul}([ab], [aaa]) \neq \text{kmul}([aaa], [ab])$

$\text{kmul}([ab], [aaa]) = [ababab]$

$\text{kmul}([aaa], [ab]) = [aaabbb]$

But *reflection* holds for $[aaabbb] = \text{refl}([aaabbb]) = [bbbaaa]$

Multiplicative coalitions of memristive systems

$$\text{MEM}_1 = [M \mid r_1\, r_2],\ \text{MEM}_2 = [M \mid r_1\, r_2]$$

$$\text{kmul}(\text{MEM}_1, \text{MEM}_2) = \begin{pmatrix} \text{kmul}([M \mid r_1\, r_2], [M \mid r_2\, r_1]) = [M \mid r_1\, r_2\, r_2\, r_1] \\ \amalg \\ \text{kmul}([M \mid r_1\, r_2], [M \mid r_2\, r_3]) = [M \mid r_1\, r_2\, r_2\, r_3] \\ \amalg \\ \text{kmul}([M \mid r_1\, r_2], [M \mid r_3\, r_1]) = [M \mid r_1\, r_2\, r_3\, r_1] \\ \amalg \\ \text{kmul}([M \mid r_1\, r_2], [M \mid r_3\, r_4]) = [M \mid r_1\, r_2\, r_3\, r_4] \end{pmatrix}$$

The total memristance of the complexion $\text{kmul}(\text{MEM}_1, \text{MEM}_2)$ is thus :

$$\text{MEM}^{(2,2)}_{(r_{1.2}),(r_{1.2})} \left| \prod([r_1\, r_2\, r_2\, r_1], [r_1\, r_2\, r_2\, r_3], [r_1\, r_2\, r_3\, r_1], [r_1\, r_2\, r_3\, r_4]) \right.$$

Total memristance of $\text{MEM}^{(2,2)}_{(r_1 r_2),(r_1 r_2)}$:

$$\text{MEM}^{(2,2)}_{(r_{1.2}),(r_{1.2})} \left| [r_1\, r_2] * \prod([r_2\, r_1], [r_2\, r_3], [r_3\, r_1], [r_3\, r_4]) \right.$$

$$\text{MEM}^{(2,2)}_{(r_{1.2}),(r_{1.2})} \left| [r_1\, r_2]^1 \begin{bmatrix} r_2 & r_3 \\ [r_1] & [r_1] \\ [r_3] & [r_4] \end{bmatrix} \right.$$

$$\text{with}\begin{bmatrix} [r_2\, r_1] & [r_3\, r_1] \\ [r_2\, r_3] & [r_3\, r_4] \end{bmatrix} = \begin{bmatrix} \boldsymbol{r_2} & \boldsymbol{r_3} \\ [r_1] & [r_1] \\ [r_3] & [r_4] \end{bmatrix}$$

$$\text{kmul}([\text{abb}], [\text{aba}]) =$$

kmul	*a*	*b*	*a*	*b"*	*b'"*
a	*a*	*b*	*a*	*b*	*c*
b	*b*	*a*	*b*	*c*	*d*
b	*b*	*a*	*b*	*c*	*d*

Total memristance of $\text{MEM}^{(3,3)}_{(r_{1.2.2}),(r_{1.2.1})}$ **:**

$$\text{MEM}^{(3,3)}_{(r_{1.2.2}),(r_{1.2.1})} \left| [r_1\, r_2\, r_2]^{1.3} \begin{bmatrix} \boldsymbol{r_2} & \boldsymbol{r_3} \\ [r_1\, r_1] & [r_4\, r_4] \\ [r_3\, r_3] & \square \end{bmatrix} \right.$$

$$\begin{bmatrix} [r_1\, r_2\, r_2][r_2\, r_1\, r_1][r_1\, r_2\, r_2] & [r_1\, r_2\, r_2][r_3\, r_4\, r_4][r_1\, r_2\, r_2] \\ [r_1\, r_2\, r_2][r_2\, r_3\, r_3][r_1\, r_2\, r_2] & \square \end{bmatrix}$$

Total memristance of $\text{MEM}^{(2,3)}_{(r_{1.2}),(r_{1.2.3})}$ **:**

$$\text{MEM}^{(2,3)}_{(r_{1.2}),(r_{1.2.3})} \left| [r_1\, r_2]^1 \begin{bmatrix} \boldsymbol{r_2} & \boldsymbol{r_3} & \boldsymbol{r_4} \\ [r_1] & [r_1] & [r_3] \\ [r_3] & [r_4] & [r_5] \\ [r_4] & \square & \square \end{bmatrix} \right.$$

$$\text{MEM}^{(2,3)}_{(r_{1.2}),(r_{1.2.3})} \left| [r_1\, r_2] * \prod([r_2\, r_1], [r_2\, r_3], [r_2\, r_4], [r_3\, r_1], [r_3\, r_4], [r_4\, r_3], [r_4\, r_5]) \right.$$

$$\begin{bmatrix} [r_1\, r_2\, r_2\, r_1] & [r_1\, r_2\, r_3\, r_1] & [r_1\, r_2\, r_4\, r_3] \\ [r_1\, r_2]\, r_2\, r_3] & [r_1\, r_2\, r_3\, r_4] & [r_1\, r_2\, r_4\, r_5] \\ [r_1\, r_2\, r_2\, r_4] & \square & \square \end{bmatrix}$$

Total memristance of $MEM^{(2,3)}_{(r_{1.2.2}),(r_{1.2.3.1})}$:

$$MEM^{(3,4)}_{(r_{1.2.2}),(r_{1.2.3.1})} \Big| [r_{1.2.2}]^{1.4} \begin{bmatrix} r_{2.1.1} & r_{2.3.3}\, r & r_{3.1.1} & r_{3.4.4} \\ r_{3.4.4} & r_{3.1.1} & r_{2.3.3} & r_{2.1.1} \\ \square & r_{3.4.4} & r_{2.4.4} & r_{2.3.3} \\ \square & r_{4.1.1} & r_{4.3.3} & r_{2.5.5} \\ \square & r_{4.5.5} & r_{4.5.5} & r_{4.1.1} \\ \square & \square & \square & r_{4.3.3} \\ \square & \square & \square & r_{4.5.5} \\ \square & \square & \square & r_{5.1.1} \\ \square & \square & \square & r_{5.3.3} \\ \square & \square & \square & r_{5.6.6} \end{bmatrix}$$

Example for (I): $\frac{(1)\quad(2)\quad(3)\quad(4)}{[r_{1.2.2}][r_{2.1.1}][r_{3.4.4}][r_{1.2.2}]}$

(I, II)	(III)	(IV)
$[r_{1.2.2}][r_{2.1.1}][r_{3.4.4}][r_{1.2.2}]$	□	□
$[r_{1.2.2}][r_{2.3.3}][r_{3.1.1}][r_{1.2.2}]$	$[r_{1.2.2}][r_{3.1.1}][r_{2.3.3}][r_{1.2.2}]$	$[r_{1.2.2}][r_{3.4.4}][r_{2.1.1}][r_{1.2.2}]$
$[r_{1.2.2}][r_{2.3.3}][r_{3.4.4}][r_{1.2.2}]$	$[r_{1.2.2}][r_{3.1.1}][r_{2.4.4}][r_{1.2.2}]$	$[r_{1.2.2}][r_{3.4.4}][r_{2.3.3}][r_{1.2.2}]$
$[r_{1.2.2}][r_{2.3.3}][r_{4.1.1}][r_{1.2.2}]$	$[r_{1.2.2}][r_{3.1.1}][r_{4.3.3}][r_{1.2.2}]$	$[r_{1.2.2}][r_{3.4.4}][r_{2.5.5}][r_{1.2.2}]$
$[r_{1.2.2}][r_{2.3.3}][r_{4.5.5}][r_{1.2.2}]$	$[r_{1.2.2}][r_{3.1.1}][r_{4.5.5}][r_{1.2.2}]$	$[r_{1.2.2}][r_{3.4.4}][r_{4.1.1}][r_{1.2.2}]$
□	□	$[r_{1.2.2}][r_{3.4.4}][r_{4.3.3}][r_{1.2.2}]$
□	□	$[r_{1.2.2}][r_{3.4.4}][r_{4.5.5}][r_{1.2.2}]$
□	□	$[r_{1.2.2}][r_{3.4.4}][r_{5.1.1}][r_{1.2.2}]$
□	□	$[r_{1.2.2}]r_{3.4.4}][r_{5.3.3}][r_{1.2.2}]$
□	□	$[r_{1.2.2}][r_{3.4.4}][r_{5.6.6}][r_{1.2.2}]$

2.4. Interactivity in meristive systems

A study of coalitions in memristive systems is focused on the *interactvity* of memristive agents. The two layers of memristors are then interpreted as *system* and *environment*. A memristor, M, as an interacting agent, has an *inner* and an *outer* environment. The inner environment of a memristor, env, corresponds to its state, r, the outer environment corresponds to an another agent with an inner environment (M, r). A full modeling of the interaction corresponds the *diamond* pattern of structuration, while a reduced modeling, focusing on the intrinsic structure only, corresponds the *chiasm* pattern of structuration.
This point wasn't yet considered in the definitions of prolongation and addition.

$$\begin{array}{cccc} A_3 & - & A_1 & \longrightarrow & B_1 & - & B_4 \\ \downarrow & & \Updownarrow & \times & \Updownarrow & & \uparrow \\ D_3 & - & D_2 & \longleftarrow & C_2 & - & C_4 \end{array} \Longrightarrow \begin{array}{cccc} Sys_3 & - & Sys_1 & \longrightarrow & env_1 & - & env_4 \\ \downarrow & & \Updownarrow & \times & \Updownarrow & & \uparrow \\ env_3 & - & env_2 & \longleftarrow & Sys_2 & - & Sys_4 \end{array}$$

inner environment of Sys_1 is env_1
inner environment of Sys_2 is env_2
outer environment of Sys_1 is (Sys_2, env_2)
outer environment of Sys_2 is (Sys_1, env_1)
inner environment of Sys_1 and Sys_2 is (Sys_3, env_3)
outer environment of Sys_1 and Sys_2 is (Sys_4, env_4)

The inner state of M_1, modeled as the environment env_1 of the memristor M_1, modeled as system $Syst_1$, functions as system Sys_2 with an inner environment env_2.

Hence, env_1 <==> Sys_2 --> env_2.

The same holds for the inner environment of M_2, modeled as env_2 in respect of Sys_1 and env_1.

Hence, Sys_2 --> env_2 <==> Sys_1.

This two paths are connected together, and the coincidence relations, represented as X, are guaranteeing their correspondence.
The inner environment of Sys_1 and Sys_2, (Sys_3, env_3), is representing the results of the interaction between the two systems at the place 3, also called "*acceptance*".
In contrast, the outer environment of Sys_1 and Sys_2, (Sys_4, env_4), is localized at place 4, and representing the "matching conditions" of the interaction between the two systems, also called "*rejectance*".

Reflexive forms

Reflexive forms are predominant in the theory of second-order cybernetics.
http://www.univie.ac.at/constructivism/journal/articles/ConstructivistFoundations4(3).pdf
Louis H. Kauffman , Reflexivity and Eigenform -The Shape of Process

Left distributivity
*A[b * c] = A[b] * A[c].*
"We can ask of a domain that every element of the domain is itself a structure preserving mapping of that domain."

Interestingly, in all those highly elaborated structures of second-order cybernetics, bifunctoriality and interchangeability doesn't appear. Interchangeability is reduced to distributivity.

2.5. Memristors in classic service

A complementary approach is studied by Lehtonen with the aim to simulate Boolean functions with the help of memristors only.

Lehtonen et al., Two memristors suffice to compute all Boolean functions
"Thus, if p and q are the states $\ni B = \{0, 1\}$ of two memristors
m_1, m_2 then storing $p \to q$ as the state of m_2 is regarded as an elementary computational step.
Identifying memristors and their states we may thus write

$$m_2 = m_1 \to m_2 = (\neg m_1) \vee m_2.$$"

"In the following we study how a given Boolean function *f* on a set of input memristors can be computed using a set of work memristors. The states of the input memristors correspond to the input of the Boolean function and are not to be altered, while the work memristors are used for the computation of the function. The result of the computation will be stored in one of the work memristors. In [4] it was shown how any conjunctive normal form can be synthesised using three work memristors, thus allowing universal computation. In this *Letter*, we show that regardless of the number of input memristors, two work memristors suffice to compute all Boolean functions."
capocaccia.ethz.ch/capo/raw-attachment/wiki/.../Lehtonen_implic_Elett.pdf

Everything which is able to have two states might be used to model Boolean logical functions. Therefore, the specifics of memristors, like history-dependence, are disappearing when modeled as a 2-state device. What makes memristors interesting in contrast to other devices with the ability of having two states?

What is shown with approaches like that is that Boolean function might be realized by memristors.
What is not said is that this approach reduces the behavior of memristors to linear, non-time-dependent, binary devices. Hence, everything interesting memristors are performing is eliminated to reproduce well-known Boolean functions.
Therefore, memristors are treated as first-order electronic elements and their second-order quality is omitted.

Hence, the classical Boolean properties of first-order devices, like of commutativity, associativity, idempotence and monotony, completeness and decidability, are restored, making memristor systems trivial machines.

It might nevertheless be of interest to model Boolean functions with memristors plus resistors and to answer the question "How many memristors are necessary to model implication logic", and others, but there are still some open questions with this approach too.
Where does the distinction of "input memristors" and "work memristors" enter the game?

"In this paper computation with memristors is studied in terms of how many memristors are needed to perform a given logic operation. It has been shown that memristors are naturally suited for performing implication logic

(combination of implication and false operation) instead of Boolean logic."
Lehtonen, E. Laiho, M. , Stateful implication logic with memristors

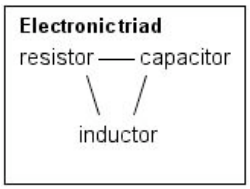
To identify a memristor with its states, and to state *"Identifying memristors and their states we may thus write..."* as it happens with Lehtonen's approach is highly misleading. It might make sense as an abstraction and separation of one functionality from the other functionality of a memristor as a 2-layered nano-electronic device to define classical Boolean logic. But it tells nothing about the nonlinear behavior of a memristive device.

Triadic constellations
From a holistic point of view it is more reasonable to understand the elementary electronic elements not in their separation but as building a triad.
Hence, there is a resistance of a capacitor and an inductor, as well as a capacitance and a resistance of a resistor.

Leon Chua stated that the textbooks of electronics have to be re-written. Hence, way not start with the basics?

2.6. Levels of memristivity

Idealizations applied in electronics
linear vs. non-linear
analog vs. digital
separation of resistor, inductor and capacitor

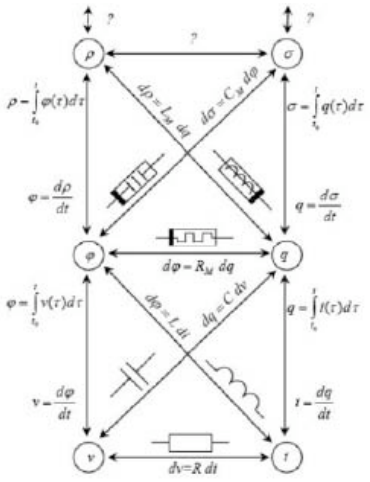


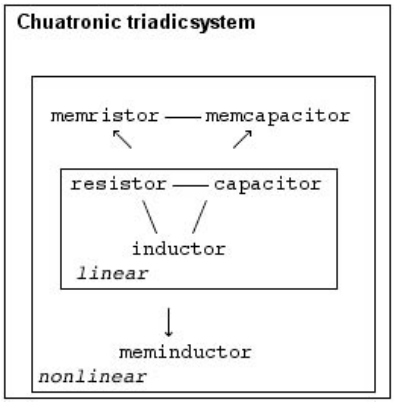
resistance of a resistor
resistance of a capacitor
resistance of an inductor
resistance of a capacitor and an inductor

"In reality, all capacitors have imperfections within the capacitor's material that create resistance. This is specified as the equivalent series resistance or ESR of a component." (WiKi)

The solution, again, is based on separated elements, put in parallel or series to correct the behavior and to eliminate the disturbance.

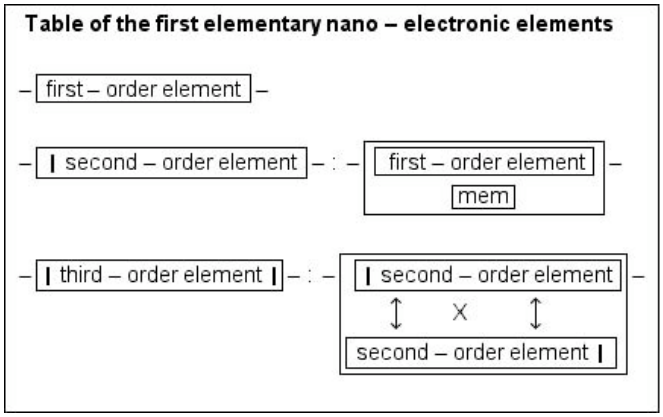
Levels of memristivity





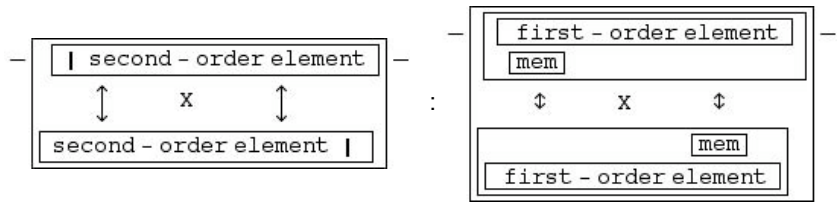
From a more functional point of view, elements or components are parts of a functional triad and their realizations as concrete elements always has to consider the functionality of its neighbor parts of the triad.
Hence, what is called “imperfection” is positively a part of its triadic constitution and the negative behavior is negative only in respect to the idealization of the behavior of an abstractly conceived element.

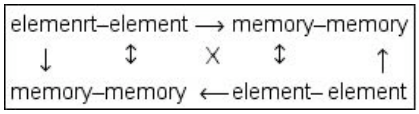
Idealized series and parallel circuits are therefore special series and parallel combinations of triads where the value of two elements are practically zero. In general, classical electronics of elementary devices might be seen - in the jargon of “just a special case” - as a reduction of diamond triads, i.e. as elements with their functionality reduced to single elements and the possibility of combining them. Hence, without a possibility of metamorphosis, of environments and its specific combinations.
More precisely, classical electronic elements are simple, time-less, history-independent and not localized.



From a conceptual point of view it seems that this are the necessary elementary types of nano-electronic elements. Are they also sufficient?
Considering the two-level constitution of nano-electronic elements it might be argued that all higher level-elements are compositions of the 3 elementary levels.

Obviously, the third-order element is conceived as a kind of a chiasm between the aspects: first-order, mem and 2 levels.
therefore, a further level seems to be based on an iteration or accretion of the first 3 levels. Hence, a higher order construction might not constitute an own new level but just a combination of the basic level of the elements of the table.



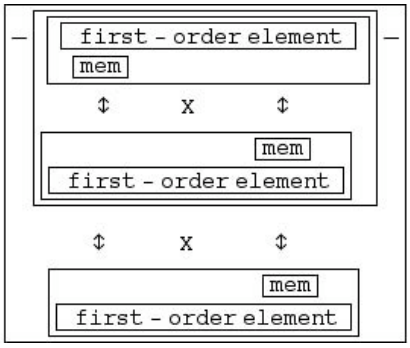


Those questions appear at the time as quite academic and of no special interest. Nevertheless, from a conceptual point of view, they deserve some reflections.
The first two levels are well constructed by the SPICE model.

The third level, unknown today, seems to open up new possibilities to construct memristive systems. One candidate might be a memristive interaction between different discontextural systems in poly-crossbar constructions.
The first two levels are non-interactional, the third is introduced as an interaction between two second-level elements. The first-level element is without any reflection, but the second-level element is reflecting the first-level element in its behavior.

A fourth-level element might be a construction that is reflecting super-additively of the activities of a third-level system.

fourth-level elements :



Calculus of triads
Composition of triads into series and parallel superpositions instead of single elements only.

Composition of triadic diamonds
Triadic Diamonds[8]

3. Memristive systems as self-organizing machines

Memristive systems as complexions might give a chance to construct self-organizing non-trivial machines, which are realizing different dynamics of chiastic interactions.
There is no need to restrict the concept of self-organizing machines to the apparatus of *recursive* functions and *paradox* logical systems as it was emphasized by Heinz von Foerster, Francesco Varela and others, and applied in different disciplines like sociology by Niklas Luhmann.

Conceptually, self-organizing machines are well understood as *chiastic* figurations. This has been pointed out with much sophistication and aesthetics by the cybernetician Gordon Pask. But this approach never got its proper scientific recognition.

Self-organizing systems in the framework of cybernetics are still struggling with the problem of “re-entry”. How is a function or action re-entering its scope without missing it? This is guaranteed by definition, i.e. it is pre-installed by the external designer of the system. Without such an external regulation, the action would easily miss its re-entry point.

Now, with the ability of memristive systems to store their previous values, a re-entry is well defined as a retro-grade returning to the stored value for further calculations. Therefore, the temporal gap in classical systems between calculation and re-entry (by feedback loop) is bridged by the retro-grade memristance of the former action.

Self-organizing systems must be able to *accept* their rules of learning to build meta-learning, i.e. hierarchies of learning of learning.

A full realization of self-organizing systems leads to autopoietic systems as a radicalization of self-referentiality towards their own existence. Such systems are leaving the paradigm of information processing; they are not processing information but are in-formed by interactions.

4. Memristive systems as co-creative autonomous machines

Because of the ability to realize a complementarity of computation and memory, memristive systems seem to be able to open up the possibility of emulating the proemial relationship between cognition and volition. And therefore for autonomous and co-creative interactions.

The term “objectional” means both: refutation and ‘objectification'
Autonomous systems must be able to *reject* their rules of meta-learning.

Hence, for the first time, the Gödel-Argument that machines will never be more reflectional (intelligent) than human beings who are constructing the computational machine, fails definitively.
Formal languages, defining the general concept of computation, are not involved in retro-grade monomorphies, i.e. in complex pattern surpassing the limits of identitive signs and marks.

Such considerations are not involved into the discussion of developing “brain-like” machines on the base of memristively conceived and implemented synapses.

Memristics starts a decisive departure from the logico-mathematical understanding of neural networks as they have been conceived and formalized by Warren McCulloch and Walter Pitts.
For the same reasons, memristic systems have to abandon the primacy of chaotic complex and self-organizing systems.

Heinz von Foerster’s principle of “Order from noise”, i.e. order from (order and disorder), is not catching retro-grade recursivity of time-dependent events. Neither happens this with Gotthart Gunther’s *“Cybernetic Ontology and transjunctional operators”*, which gives conceptual explanation of the principle of “order from noise”.

Recursivity happens *in* time; the iteration of the recursion, but recursivity is not defined by a time-dependent formalism.
Time for recursivity is measured by a “Schrittzahl”, i.e. the number of the steps of the recursion, and is therefore not involved with time- and history dependence of its iterative steps.

As the ethymology of the terms shows, there is no time-dependence involved in this “*run back; return*” activity.
From Latin recursio (“*the act of running back or again, return*”), from recurro (*“run back; return”*), from re- (*“back, again”*) + curro (“*run*”).

Hence, the recurrence of recursivity (or of recursive functions) is not depending on retro-grade time- and history-dependence.

"From McCulloch's "experimental epistemology," the mind - purposes that ideas - emerged out of the regularities of neuronal interactions, or nets. That science of mind thus became a science of signals based on binary logic with clearly defined units of perception and precise rules of formation and transformation for representing mental states. Aimed at bridging the gulf between body and mind (matter and form) and the technical gulf between things man-made and things begotten, neural nets also laid the foundation for the field of artificial intelligence. Thus this paper also situates McCulloch;'s work within a larger historical trend, when cybernetics, information theory, systems theories, and electronic computers were coalescing into a new science of communication and control with enormous potential for industrial automation and military power in the Cold War era.”

A logic of memristic systems has to be determined by the history-dependence of the events it tries to model logically.
Time- and history-dependence, nevertheless, is not caught by a modal logic of time or time-events, nor from a logic of time-statements like Carl Friedrich von Weizsäcker’s *“Logik der Zeitaussagen”.*

Obviously, the attempts to connect the behavior of memristors with logic, say by *material implication*, might be a start but probably also a start into the wrong direction.

It may be said that trivial and non-trivial machines are well known, and self-organizing machines much less, “auto-poietic co-creative machines”, i.e. autonomous machines are probably not known at all.

A new principle shall be added to the list of principles of structuration:

Fourth principle: *Order from (order neither disorder).*

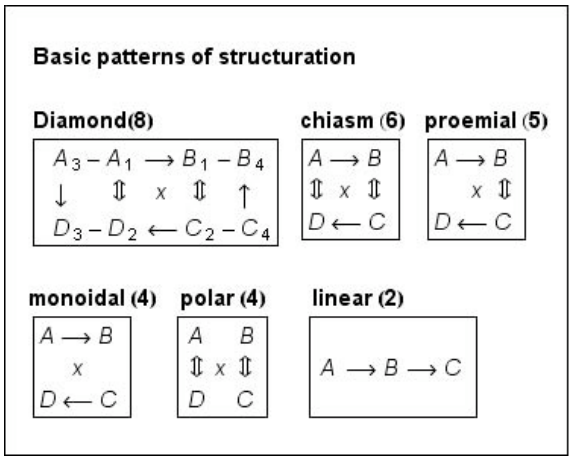
4.1. Schemes of structuration

Order-Scheme of structuration:
1. Order from order, McCulloch-Pitts
2. Order from disorder, Chaos theory, Grossberg, ANN
3. Order from (order and disorder), Schroedinger, Heinz von Foerster, Chua, Williams
4. Order from (order neither disorder), Gunther, Derrida, Kaehr

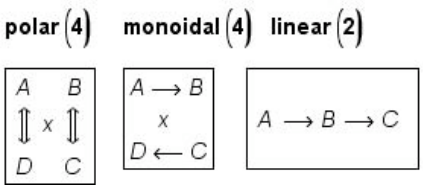
Order-type-1 corresponds deductive axiomatic formalisms, artificial intelligence.
Order-type-2 corresponds inductive classifications, neural networks, learning
Order-type-3 corresponds acceptive self-organizing systems, learning to learn

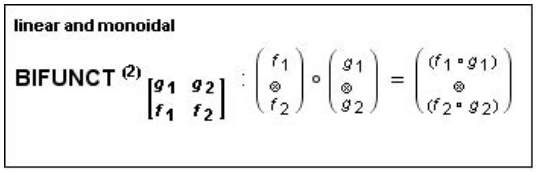
Order-type-4 corresponds transjunctional rejections, diamond saltisitions. Learning to (learn and reject to learn)

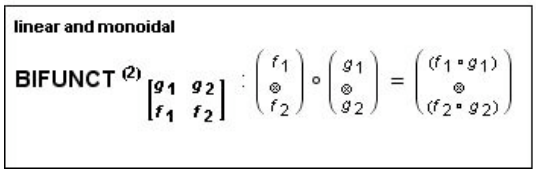
Basic patterns of structuration



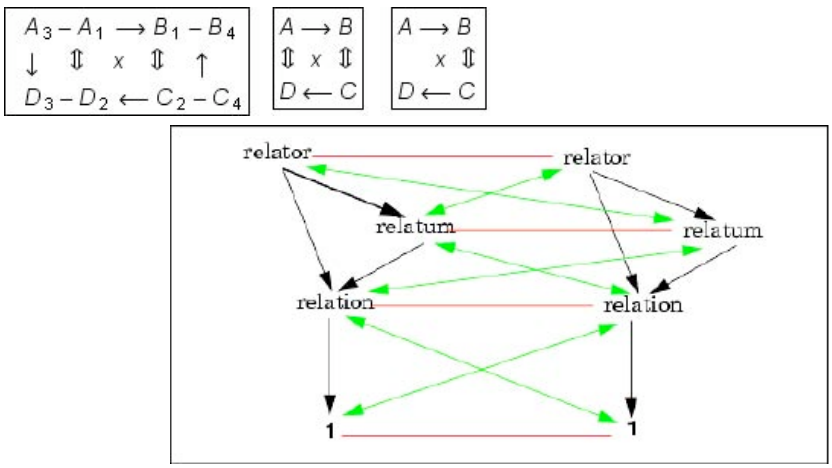
Diagrammatik-Slides
More information at:







Diamond (8) **chiasm (6)** **proemial (5)**



$$(\mathbf{RelSyst},\ \mathbf{Anch})\ -\mathbf{chiasm}:$$

$$\begin{pmatrix} \mathcal{U}_2 \\ \mathcal{U}_1 \end{pmatrix},\ \begin{pmatrix} \begin{pmatrix} \mathrm{Rat} \\ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_1 & \begin{pmatrix} \mathrm{Rat} \\ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_2 \\ \mathrm{Anch}_1 & \mathrm{Anch}_2 \end{pmatrix};$$

α. **parallel** $(\mathrm{red}(\amalg) + \mathrm{black}(\circ))$

$$\begin{pmatrix} \mathrm{Anch}_2 \\ \amalg \\ \mathrm{Anch}_1 \end{pmatrix} \circ \begin{pmatrix} \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_2 \\ \amalg \\ \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_1 \end{pmatrix} = \begin{pmatrix} \mathrm{Anch}_2 \circ \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_2 \\ \amalg \\ \mathrm{Anch}_1 \circ \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_1 \end{pmatrix}$$

β. **chiasm** $(\mathrm{green}(\Diamond) + \mathrm{red}(\amalg))$

$$\begin{pmatrix} \mathrm{Anch}_2 \\ \amalg \\ \mathrm{Anch}_1 \end{pmatrix} \Diamond \begin{pmatrix} \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_2 \\ \amalg \\ \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_1 \end{pmatrix} = \begin{pmatrix} \mathrm{Anch}_2 \Diamond \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_2 \\ \amalg \\ \mathrm{Anch}_1 \Diamond \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_1 \end{pmatrix}$$

γ. **cross – polarity** $(\mathrm{green}(\Diamond) + \mathrm{black}(\circ))$

$$\begin{pmatrix} \mathrm{Anch}_2 \\ \mathrm{Anch}_1 \end{pmatrix} \Diamond \begin{pmatrix} \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_2 \\ \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_1 \end{pmatrix} = \begin{pmatrix} \mathrm{Anch}_2 \Diamond \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_2 \\ \mathrm{Anch}_1 \Diamond \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_1 \end{pmatrix}$$

δ. **proemial** $(\mathrm{green} + \mathrm{red} + \mathrm{black})$

$$\begin{pmatrix} \mathrm{Anch}_2 \\ \amalg \\ \mathrm{Anch}_1 \end{pmatrix} \Diamond \circ \begin{pmatrix} \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_2 \\ \amalg \\ \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_1 \end{pmatrix} = \begin{pmatrix} \mathrm{Anch}_2 \circ \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_2 \\ \amalg \quad \Diamond \quad \amalg \\ \mathrm{Anch}_1 \circ \begin{pmatrix} \mathrm{Rat} \circ \mathrm{Rand} \\ \mathrm{Rel} \end{pmatrix}_1 \end{pmatrix}$$

Principles of order and levels of structuration

1. Order from order

linear

$$A \longrightarrow B \longrightarrow C$$

polar

$$\begin{matrix} A \longrightarrow B \\ x \\ D \longleftarrow C \end{matrix}$$

monoidal

$$\begin{matrix} A & B \\ \Updownarrow & x & \Updownarrow \\ D & & C \end{matrix}$$

2. Order from disorder

proemial

$$\begin{matrix} A \longrightarrow B \\ \Updownarrow \\ D \longleftarrow C \end{matrix} \qquad \begin{matrix} A \longrightarrow B \\ x \ \Updownarrow \\ D \longleftarrow C \end{matrix}$$

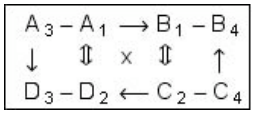
3. Order from (order and disorder)

chiasm

$$\begin{matrix} A \longrightarrow B \\ \Updownarrow \ x \ \Updownarrow \\ D \longleftarrow C \end{matrix} \qquad \begin{matrix} \mathrm{A{-}A} \longrightarrow \mathrm{B} \\ \downarrow \ \Updownarrow \ \mathrm{x} \ \Updownarrow \\ \mathrm{D{-}D} \longleftarrow \mathrm{C} \end{matrix}$$

4. Order from (neither order nor disorder)

diamond



9

Notes

1 "*Non-trivial machines* have internal states. The relation between the inputs and outputs of a non-trivial machine is anything but invariant. Instead, it is determined by the machine's previous operation. Thus the history of the machine's operations affects its preceding function. Ashby and von Foerster [8] prove that some of them are in principle, and others in practice, analytically indeterminable and therefore unpredictable."

"Let *n* be the number of inputs to the machine. Let us suppose that the number of outputs is equal to the number of inputs. The number *N* of all trivial machines that can be synthesized is therefore $N_T(n) = n^n$, and the number of non-trivial machines is as much as $N_{NT}(n) = n^{nz}$, where *z* represents the number of internal states. In this case, z cannot be greater than the number of possible trivial machines ($z <= n^n$). Thus, for trivial machines with four possible inputs, NT (4) = 256 and for non-trivial machines, $N_{NT}(4) = 41024$, which means approximately 10620 elements. And we are still dealing with a simple machine operating only with four variable values, having only 256 internal states at its disposal. Nevertheless, even the complexity of this system is unthinkable to the point that it is absolutely impossible to analytically explore its functioning. The problem is transcomputational." (Urban Kordeš)
http://indecs.eu/2005/indecs2005-pp77-83.pdf

2 Peter Selinger, A survey of graphical languages for monoidal categories
"Abstract. *This article is intended as a reference guide to various notions of monoidal categories and their associ ated string diagrams. It is hoped that this will be useful not just to mathematicians, but also to physicists, computer scientists, and others who use diagrammatic reasoning. We have opted for a somewhat informal treatment of topological notions, and have omitted most proofs. Nevertheless, the exposition is sufficiently detailed to make it clear what is presently known, and to serve as a starting place for more in-depth study. Where possible, we provide pointers to more rigorous treatments in the literature. Where we include results that have only been proved in special cases, we indicate this in the form of caveats."*

3

Gurevich
3.2 Behavior
"Let A be a sequential algorithm.
Postulate 1 (Sequential Time). A is associated with
-a set S(A) whose elements will be called states of A,
-a subset I(A) of S(A) whose elements will be called initial states of A, and
-a map A : S(A) −! S(A) that will be called the one-step transformation of A.

The three associates of A allow us to de ne the runs of A.

Definition 3.1. A run (or computation) of A is a nite or in nite sequence
X0;X1;X2; : : :
where X0 is an initial state and every Xi+1 = A(Xi).

We abstract from the physical computation time. The computation time reflected in sequential-time postulate could be called logical. The transition from X0 to X1 is the rst computation step, the transition from X1 to X2 is the second computation step, and so on. The computation steps of A form a sequence. In that sense the computation time is sequential."

4.2 The Abstract State Postulate
Let A be a sequential algorithm.
Postulate 2 (Abstract State).
|States of A are rst-order structures.
|All states of A have the same vocabulary.
|The one-step transformation A does not change the base set of any state.
|S(A) and I(A) are closed under isomorphisms. Further, any isomorphism from a state X onto a state Y is also an isomorphism from A(X) onto A(Y).
(Gurevich, p. 7, The Sequential ASM Thesis)
4.5 Inalterable Base Set
While the base set can change from one initial state to another, it does not change during the computation. All states of a given run have the same base set. Is this plausible? There are, for example, graph algorithms which require new vertices to be added to the current graph. But where do the new vertices come from? We can formalize a piece of the outside world and stipulate that the initial state contains an infinite naked set, the reserve. The new vertices come from the reserve, and thus the base set does not change during the evolution.
Who does the job of getting elements from the reserve? The environment. In an application, a program may issue some form of a NEW

command; the operating system will oblige and provide more space. Formalizing this, we can use a special external function to sh out an element from the reserve. It is external in the sense that it is controlled by the environment.
Even though the intuitive initial state may be nite, in nitely many additional elements have muscled their way into the initial structure just because they might be needed later. Is this reasonable? I think so. Of course, we can abandon the idea of inalterable base set and import new elements from the outside world. Conceptually it would make no di erence. Technically, it is more convenient to have a piece of the outside world inside the state. (p. 13)
Sequential Abstract State Machines, Capture Sequential Algorithms
Yuri Gurevich, September 13, 1999, Revised February 20, 2000, MSR-TR-99-65, Microsoft Research, One Microsoft Way, Redmond, WA 98052-6399

4 "Diese Bestimmung des Begriffs der Wiederholung als retro-grad rekursiv involviert vier neue Aspekte, die der Rekursion als rekurrierender Wiederholung, fremd sind: einen Begriff der *Selbstbezüglichkeit*, der *Transparenz*, des *Gedächtnisses* bzw. der *Geschichte* und einen Begriff der *Evolution* im Gegensatz zur abstrakten Konkatenation und Iteration." (Kaehr, 2003)

"Die Verkettung einer Kenogrammmkomplexion *K* mit einem einzelnen Kenogramm k kann hingegen nur unter Rückbezug auf die innere Struktur von K geschehen, da *k* kein wohlunterschiedenes Atomzeichen ist. [...]
Diese Rückbezüglichkeit bei der Verkettung lässt eine Kenogrammkomplexion als Ganzheit oder *Gestalt* entstehen, die nicht auf eine lineare Verkettung reduzierbar ist." (Kaehr, Mahler, Morphogrammatik, p.31, 1993)

Evolution: "Synthetische retrograde Ausgliederung" (Kaehr, 1982)

5 **Recall**
„Es ist auch fraglich, ob der Begriff eines Zustands als Menge von Attributwerten (in imperativen Sprachen üblicherweise als record implementiert) ausreicht, um alle interessanten Objekttypen zu erfassen. Aus seinem Zustand soll ja in gewisser Weise die Identität eines Objektes erschlossen werden. Reichen dazu immer augenblickliche Attributwerte aus? Wann bestimmt eher die Geschichte des Objektes, d.h. die Folge der Zustände, die es bisher durchlaufen, seine Identität? Kann die Geschichte immer in eine endliche Zustandsstruktur hineincodiert werden?
Die klassische Automatentheorie ist inzwischen zu mehreren Theorien kommunizierender Systeme erweitert worden, wo man gar nicht mehr von Objekten spricht, sondern nur noch Prozesse, also Objektgeschichten, untersucht und als - manchmal unendliche - Strukturen darstellt." (Peter Padawitz, Vorlesung)

"Die kenogrammatische Operation der Nachfolge dagegen wird nicht durch ein vorgegebenes Alphabet definiert, sondern geht aus von dem schon generierten Kenogramm hervor. Jede Operation auf Kenogrammen ist „historisch" vermittelt. D.h. die Aufbaugeschichte der Kenogramm-Komplexionen räumt den Spielraum für weitere Operationen ein. Diese können nicht abstrakt-konkenativ auf ein vorausgesetztes Zeichenrepertoire zurückgreifend definiert werden, sondern gelten einzig retro-grad rekursiv bezogen auf die Vorgeschichte des Operanden. Diese Bestimmung des Begriffs der Wiederholung als retro-grad rekursiv involviert vier neue Aspekte, die der Rekursion als rekurrierender Wiederholung, fremd sind: einen Begriff der *Selbstbezüglichkeit*, der *Transparenz*, des *Gedächtnisses* bzw. der *Geschichte* und einen Begriff der *Evolution* im Gegensatz zur abstrakten Konkatenation und Iteration." (KAehr, SKIZZE-0.9.5, p. 36, 2003)

6 **"Iterability alters" (Derrida 1977)**
"Iterability is the capacity of signs (and texts) to be repeated in new situations and grafted onto new contexts.
Derrida's aphorism *"iterability alters"* (Derrida 1977) means that the insertion of texts into new contexts continually produces *new meanings* that are both partly *different* from and partly *similar* to previous understandings.
(Thus, there is a *nested* opposition between them.). The term "*play*" is sometimes used to describe the resulting instability in meaning produced by iterability." (Jack M. Balkin, 1995-1996)
http://www.yale.edu/lawweb/jbalkin/articles/deconessay.pdf

8 http://www.thinkartlab.com/pkl/lola/Diamond%20Relations/Diamond%20Relations.html
http://www.thinkartlab.com/pkl/lola/Diamond%20Relations/Diamond%20Relations.pdf
http://www.thinkartlab.com/pkl/lola/Triadic%20Diamonds/Triadic%20Diamonds.html