Article

Infusing Autopoietic and Cognitive Behaviors into Digital Automata to Improve their Sentience, Resilience and Intelligence

Rao Mikkilineni [1,*]

[1] Ageno School of Business, Golden Gate University, San Francisco, CA 94105, USA; rmikkilineni@ggu.edu;
* Correspondence: rmikkilineni@ggu.edu ;

Abstract: The holy grail of Artificial Intelligence (AI) has been to mimic human intelligence using computing machines. Autopoiesis which refers to a system with well-defined identity and is capable of reproducing and maintaining itself and cognition which is the ability to process information, apply knowledge, and change the circumstance are associated with resilience and intelligence. While classical computer science (CCS) with symbolic and sub-symbolic computing has given us tools to decipher the mysteries of physical, chemical and biological systems in nature and allowed us to model, analyze various observations and use information to optimize our interactions with each other and with our environment, it falls short in reproducing even the basic behaviors of living organisms. We present the foundational shortcomings of CCS and discuss the science of information processing structures (SIPS) that allows us to fill the gaps. SIPS allows us to model super-symbolic computations and infuse autopoietic and cognitive behaviors into digital machines. They use common knowledge representation from the information gained using both symbolic and sub-symbolic computations in the form of system-wide knowledge networks consisting of knowledge nodes and information sharing channels with other knowledge nodes. The knowledge nodes wired together fire together to exhibit autopoietic and cognitive behaviors.

Keywords: Artificial Intelligence; Symbolic Computing; Sub-symbolic Computing; Super-symbolic Computing; Knowledge Networks; General Theory of Information

1. Introduction

As von Neumann [1] pointed out "It is very likely that on the basis of philosophy that every error has to be caught, explained, and corrected, a system of the complexity of the living organism would not last for a millisecond. Such a system is so integrated that it can operate across errors. An error in it does not in general, indicate a degenerative tendency. The system is sufficiently flexible and well organized that as soon as an error shows up in any part of it, the system automatically senses whether this error matters or not. If it doesn't matter, the system continues to operate without paying any attention to it….This is completely different philosophy from the philosophy which proclaims that the end of the world is at hand as soon as the first error has occurred."

The foundational characteristics of a living system are self-reproduction and self-maintenance (autopoiesis) defined by the boundary of the system ("self"); its interaction with the environment is seen in terms of the internal structure; consequently, the internal structure brings about a process of discrimination and selection of its interactions. This mutual co-emergence of "environment" and "self" gives rise to the process of cognition where the system develops an internal knowledge representation using its internal structures. The knowledge representation consists of not only the structures but also the dynamics. Therefore, if machines were to mimic living organisms, they must be infused with autopoietic and cognitive behaviors. This implies the need for tools that assist in self-organization, self-referentiality for defining and executing autopoietic and cognitive behaviors. The concept of "self" and the inclusion of the knowledge about the constituent components, their relationships, their interactions and consequent behavioral impact on the global state of the system play a crucial role in infusing autopoietic and cognitive behaviors into current generation digital computing machines.

Current state of the art information technologies use symbolic and sub-symbolic computations. Symbolic computing has contributed to automating tasks that can be easily described by a list of formal, mathematical rules or a sequence of event driven actions such as modeling, simulation, business workflows, interaction with devices, etc. The sub-symbolic computing, consisting of neural network model, has allowed computers to understand the world in terms of a hierarchy of concepts to perform tasks that are easy to do "intuitively", but are hard to describe formally or a sequence of event driven actions such as recognizing spoken words or faces. Current implementation of these two types of computations using John von Neumann stored program implementation of the Turing machines, are dependent on devices and methods that are governed by the Church-Turing thesis [2].

In this paper we present few foundational problems with these CCS implementations that prevent infusing autopoietic and cognitive behaviors and show how SIPS derived from the general theory of information [3 - 6] evolves current symbolic and sub-symbolic computations to include autopoietic and cognitive behaviors. A super-symbolic computing architecture [7] is introduced with a common knowledge representation from the outputs of symbolic and sub-symbolic computing structures. A resulting knowledge network provides higher level processes for organizing the constituent components into a system with self-regulation and maintain stability in the face of fluctuations in its interactions within and with its environment outside.

In section 2, we present the shortcomings of current state of the art. In Section 3, we present GTI and SIPS and provide the design for autopoietic and cognitive knowledge networks. In section 4, we present a design where the new machines can improve the sentience, resilience and intelligence of current state of the art IT without disrupting their current implementations. We conclude in section 5, the consequences of this approach in the design of future information processing structures.

2. Limitations of Current Computing Models

Limitations of Symbolic Computing Structures

All algorithms that are Turing computable fall within the boundaries of Church Turing thesis which states that “a function on the natural numbers is computable by a human being following an algorithm, ignoring resource limitations, if and only if it is computable by a Turing machine.” The resources here are the fuel for computation including CPU and memory. The stored program implementation of the Turing machine provides a computing structure where the state is represented by symbols (1s and 0s) and an algorithm operates on them to change its state.

The concept of the universal Turing machine has allowed us to create general purpose computers and [8] “use them to deterministically model any physical system, of which they are not themselves a part to an arbitrary degree of accuracy. Their logical limits arise when we try to get them to model a part of the world that includes themselves.”

There are two new drivers that are testing the boundaries of Church-Turing thesis:

1. Current business services demand non-stop operation and their performance adjusted in real-time to meet rapid fluctuations in service demand or available resources. A business application software providing these services contains many components which are often, distributed across multiple geographies and requires hardware (fuel for computation in the form of CPU and memory) owned or operated by different providers managing infrastructure from multiple vendors. Current state of the art requires myriad tools and processes to deploy and maintain the stability of the application. The result is complexity in managing local non-deterministic fluctuations in the demand for or the availability of the fuel impacting the end to end service quality.

The solution today, is either single hardware infrastructure provider lock-in or the complexity of myriad third-party tools. In addition, the speed with which the quality of service has to be adjusted to meet the demand is becoming faster than the time it takes to orchestrate the myriad infrastructure components (such as virtual machine (VM) or container images, network plumbing, application configurations, middleware etc. It takes time and effort to reconfigure distributed plumbing which results in increased cost and complexity. Church-Turing thesis boundaries are challenged when rapid non-deterministic fluctuations in the demand for or the availability of finite resources drive the demand for resource readjustment in real-time without interrupting the service transactions in progress. Autopoietic application on the other hand, would have the knowledge to reproduce the system ("self") with required resources and maintain its stability. This is analogous to how the living organisms maintain homeostasis.

2. Current business processes and their automaton assume trusted relationships between the participants in various transactions. Information processing and communication structures in turn, assume "trusted relationships" between their components. Unfortunately, global connectivity and non-deterministic fluctuations caused by the participants and also in the resource availability of information processing structures, make it necessary to verify the trust before completing transactions. In order to assure trust, application security has to become self-regulating and tolerant to manage "weak" or "no" trust in the participating entities whether they are other service components, or people or devices. The solution requires decoupling of service security mechanisms (authentication, authorization and accounting) from myriad infrastructure and service provider security operations. A cognitive application would manage its own security and privacy with appropriate knowledge independent of hardware infrastructure security mechanisms using its own autopoietic behaviors. Ths is analogous to a living organism fighting the harmful viruses with its own immune systems.

As mentioned in [8] a more foundational issue is the logical limit that arises when we try to get them to "model a part of the world that includes themselves." A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that define specific behavior or functions. The plan for implementing functional requirements is detailed in the system design, (the computed). The plan for implementing non-functional requirements is detailed in the system architecture. These requirements include availability, reliability, performance, security, scalability and efficiency at run-time (the computer). The meta-knowledge of the intent of the computed, the association of specific algorithm execution to a specific device, and the temporal evolution of information processing and exception handling when the computation deviates from the intent (be it because of software behavior or the hardware behavior or their interaction with the environment) is outside the software and hardware design and is ex-pressed in non-functional requirements. Mark Burgin calls this Infware [9] which contains the description and specification of the meta-knowledge that can also be implemented using the hardware and software to enforce the intent with appropriate actions. According to GTI, the infware provides a path to address the issue of including the computer and the computed in the information processing model of the world.

In addition, Gödel's proof [10, 11] showing the limits of logic and that axiomatizing never stops, that induction-intuition must always be present, that not all things can be proved by reason alone, allowed not only the discovery of algorithms, but also their composition. As Gilder [12] points out, "Gödel's proof prompted Alan Turing's invention in 1936 of the Turing machine—the universal computing architecture with which he showed that computer programs, like other logical schemes, not only were incomplete but could not even be proved to reach any conclusion. Any particular program might cause it to churn away forever. This was the "halting problem." Computers required what Turing called "oracles" to give them instructions and judge their outputs. Turing showed that just as the uncertainties of physics stem from using electrons and photons to measure themselves, the limitations of computers stem from recursive self-reference. Just as quantum theory fell into self-referential loops of uncertainty because it measured atoms and electrons using instruments composed of atoms and electrons, computer logic could not escape self-referential loops as its own logical structures informed its own algorithms."

The long and short of this discussion is that the foundational issues become more pronounced as the information processing structures become distributed, interact with asynchronous communication and the constituent components contributing to the end to end transaction fulfillment are managed by autonomous service providers contributing the resources for computations with different profit motives. The result is a single vendor-lockin with autocratic regulation or complexity of third party tools providing a nightmare in finding errors and fixing them when non-deterministic fluctuations cause problems in mission critical applications. A possible solution, which mimics living organisms, is self-regulation to maintain stability which requires the system composition of a "self" with an identity and processes for self-regulation using autopoiesis and cognition.

1. *Limitations of Sub-Symbolic Computing*

Deep learning, has delivered a variety of practical uses in the past decade. From revolutionizing customer experience, machine translation, language recognition, autonomous vehicle management, computer vision, text generation, speech understanding, and a multitude of other AI applications.

Deep learning models do not require algorithms to specify what to do with the data. Extraordinary amount of data, we as humans, collect and consume — is fed to deep learning models. An artificial neural network takes some input data, and transforms this input data by calculating a weighted sum over the inputs and applies a non-linear function to this transformation to calculate an intermediate state. The three steps above constitute what is known as a layer, and the transformative function is often referred to as a unit. The intermediate states—often termed features—are used as the input into another layer. Through repetition of these steps, the artificial neural network learns multiple layers of non-linear features, which it then combines in a final layer to create a prediction.

The neural network learns by generating an error signal that measures the difference between the predictions of the network and the desired values and then using this error signal to change the weights (or parameters) so that predictions get more accurate.

Therein lies the limitation of deep learning. While we gain insights about hidden correlations, extract features and distinguish categories, we lack transparency of reasoning behind these conclusions. Most importantly, there is the absence of common sense. Deep learning models might be the best at perceiving patterns. Yet they cannot comprehend what the patterns mean, and lack the ability to model their behaviors and reason about them. True intelligence involves generalizations from observations, creating models, deriving new insights from the models through reasoning. In addition, human intelligence also creates history and uses past experience in making the decision. Figure 1 shows our understanding of how we learn, memorize, model, reason and act.

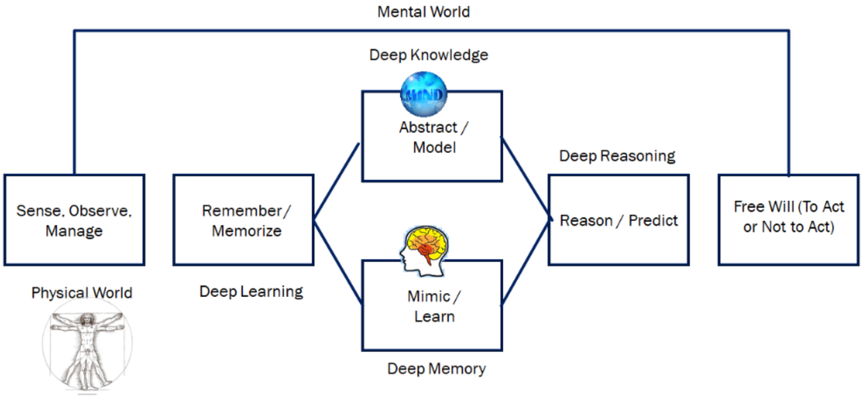

Figure 1. Model of information management exhibited by sentient, resilient and intelligent beings

entience comes from the Latin word sentient-, "feeling," and it describes physical structures that are alive, able to feel and perceive, and show awareness or responsiveness to external influences. They are aware of their inner structure (self) and their interactions with external structures both sentient and non-sentient. The degree of resilience (the capacity to recover quickly by rearranging their own structures (and also the structures they interact with) from non-deterministic difficulties without requiring a reboot or loosing self-awareness). Intelligence is the ability to acquire, model and apply knowledge and skills using the cognitive apparatuses the organism has developed.

Based on our knowledge of how natural intelligence works, we can surmise that the following key elements of human mind, which leverage the brain and the body at cellular level, are missing in current state of the art A.I.:

1. Time Dependence & History of Events: In nature, systems are continuously evolving and interacting with each other. Sentient systems (with the capacity to feel, perceive or experience) evolve using a non-Markovian process, where the conditional probability of a future state depends on not only the present state but also on its prior state history. Digital systems, to evolve to be sentient and mimic human intelligence, must include time dependence and history in their process dynamics.
2. Knowledge Composition and Transfer Learning: The main outcome of this ability is to understand and consequently predict behaviors by a succession of causal deductions supplementing correlated inductions.
3. Exploration vs. Exploitation dilemma: Creativity and expertise are the consequences of our ability to swap from the comfort zone to unchartered territories and it's a direct and key usage of our transfer learning skill. Analogies and Translations are powerful tools of creativity using knowledge in a domain and applying it in another.
4. Hierarchical structures: As was proven by Gödel, an object can only be described (and managed) by an object of a higher class. A key principle of how cells are working by exchanging proteins whose numbers, functions, and messages are supervised by DNA at cell level or group (higher) level.

Current information processing structures provide information gleamed from symbolic computing and multiple AI models providing information from voice, text and video sources. However, they are sequential in combining the knowledge gained from individual computations, and not integrated to gain concurrent insights that span across all the information available which necessitates a new approach with super-symbolic computing [7, 13]. Deep learning also lacks the ability to abstract knowledge and create models based on information from the data it processes. Learning from deep learning models provides the capability to mimic but not the ability to abstract and reason (shown in figure 1).

Super-Symbolic Computing with Integrated Knowledge Representation

As we well know, current 'state-of-the-art' information technologies (IT) utilize two forms of computation namely, symbolic computation and sub-symbolic computation. Biological systems also use both symbolic and sub-symbolic computations in the form of genes and neural networks. However, biological systems have evolved one step further by incorporating super-symbolic computation that performs computations on the combined knowledge from both symbolic and sub-symbolic computations to derive higher order autopoietic and cognitive behaviors. New computing automata called structural machines derived from GTI, provide means for modelling symbolic, sub-symbolic and super-symbolic computations performed on data and knowledge structures. In this work we argue that super-symbolic computation adds one more dimension to the general schema of computations. Synthesizing it with symbolic computation and sub-symbolic computation in one model, we come to symbiotic computation. Structural machines with flexible types of processors can accomplish symbiotic computation. Symbiotic computation combines advantages of sub-symbolic, symbolic and super-symbolic computations and advance the current state of the art IT with higher-order autopoietic and cognitive behaviors. Figure 2 shows the schema for infusing autopoietic and cognitive behaviors into current state of the art digital automata.

At the lowest level, the symbolic and sub-symbolic computations using current state of the art information technology tools and practices are depicted that provide knowledge pertaining to the domain-specific states of the systems and their evolution in the form of data structures and associated algorithms. The constituent subsystems provide the information to create a composed knowledge structure in the form of complex and multi-level network capturing the schema of various entities, their relationships and behaviors. These knowledge structures provide the means for higher level reasoning and execution of autopoietic and cognitive behaviors that define the evolution of the system as events change the state and provide new information.

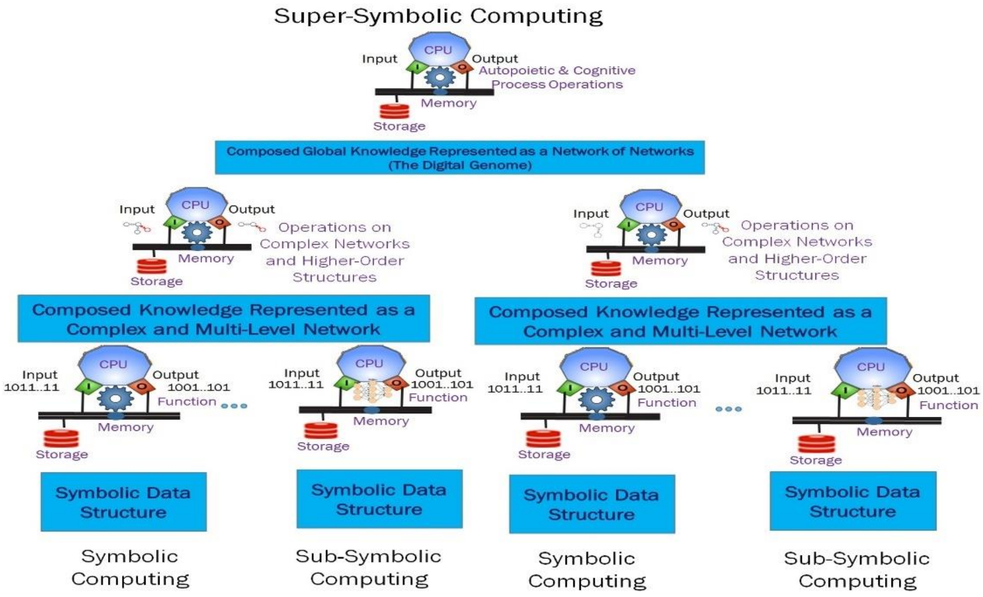
Super-Symbolic Computing
Input
CPU
Output
Autopoietic & Cognitive Process Operations
Memory
Storage
Composed Global Knowledge Represented as a Network of Networks (The Digital Genome)
Input
CPU
Output
Operations on Complex Networks and Higher-Order Structures
Memory
Storage
Input
CPU
Output
Operations on Complex Networks and Higher-Order Structures
Memory
Storage
Composed Knowledge Represented as a Complex and Multi-Level Network
Composed Knowledge Represented as a Complex and Multi-Level Network
Input 1011..11
CPU
Output 1001..101 Function
Memory
Storage
Input 1011..11
CPU
Output 1001..101 Function
Memory
Storage
Input 1011..11
CPU
Output 1001..101 Function
Memory
Storage
Input 1011..11
CPU
Output 1001..101 Function
Memory
Storage
Symbolic Data Structure
Symbolic Data Structure
Symbolic Data Structure
Symbolic Data Structure
Symbolic Computing
Sub-Symbolic Computing
Symbolic Computing
Sub-Symbolic Computing

Figure 2. The schema defining the super-symbolic computing

In the next section, we discuss the general theory of information, and various tools that provide a framework for implementing super-symbolic structures that integrate the symbolic and sub-symbolic computations with a common knowledge representation and higher level reasoning and behaviors that optimize the evolution of the system as a whole.

3. General Theory of Information and the Science of Information Processing Structures:

GTI, structural machines, and various tools derived from GTI and the theory of structural machines dealing with transformation of information and knowledge are discussed widely in several peer reviewed journals, books and other publications [3 -7, 13, 14]. In this section we summarize the role of the structural machine [14] in super-symbolic computation and its utility in infusing autopoietic and cognitive behaviors into digital automata. GTI explains and makes available constructive tools [15] "for discerning information, measures of information, information representations and carriers of information. For instance, taking a letter written on a piece of paper, we see that the paper is the carrier of information, the text on it is the representation of the information contained in this text and it is possible to measure the quantity of this information using Shannon entropy or algorithmic complexity."

The science of information processing structures deals with the theory and practice of using information that exists in the physical world in the form of structures to create abstract knowledge structures in the mental world by living organisms and improve their sentience, resilience and intelligence while dealing with fluctuations in their interactions among its constituent sub-structures and their environment.

As discussed by Mikkilineni and Burgin [14], "many philosophers, mathematicians and scientists have attempted to define knowledge and its relationship to information and its processing. Plato was perhaps the first to articulate that knowledge is always about something—some object or domain or mental concept. In order to discern a knowledge object or domain, we have to name it. A name may be a label, number, idea, text, process, and even a physical object of a relevant nature. The named objects may be composed into knowledge structures which may have inter-object and intra-object relationships and associated behaviors that may cause changes to their state, form or content. The role of information-processing structures is to discern the relationships and behaviors and evolve the state, form or content accordingly. Information in the strict sense includes a capacity to change structural elements in knowledge systems. Knowledge systems exist in the world of structures, which belongs to the well-known existential triad of the world together with the physical world and the mental world."

We summarize here the basic concepts and tools from GTI which allow us to model autopoietic and cognitive behaviors and infuse them into digital automata.

Fundamental Triad or a Named Set: Structural relationships exist between data, which are entities observed in the physical world or conceived in the mental world. These structures define the knowledge of them in terms of their properties such as attributes, relationships and dynamics of their interaction. Information processing structures organize evolution of knowledge structures by an overlay of cognitive knowledge structures, which model, monitor and manage the evolution of the information processing system. The most fundamental structure is called a fundamental triad or a named set [16]. It has the following visual representation shown in Figure 3.

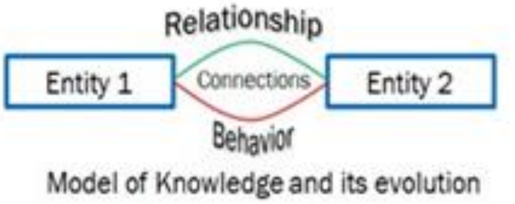

Figure 3. Visual representation of a fundamental triad (named set).

An entity or object with its own name is connected to another entity or object with its own name. The connection, which itself has a name, depicts the knowledge of the relationship and the behavioral evolution when a state change occurs in either object. The behavioral change could be either local within the object or through induced behavior using information exchange among the entities based in their relationship. Entities wired together fire together to execute the collective behavior.

It is demonstrated that fundamental triads, i.e., named sets, form the most basic structures in nature, technology and society, actually, in everything. This means that the theory of named sets (fundamental triads) is the true Theory of Everything [16].

Knowledge Structure: A knowledge structure [17, 18] is composed of related fundamental triads, and any state change causes behavioral evolution based on the connections and relationships. The theory of knowledge states that the information is transformed into knowledge in the form of fundamental tttfvriads and their composed structures. Changes in information cause changes in the knowledge structures based on their relationships and associated behaviors defining the state and its evolution. Knowledge processing is an important feature of intelligence in general and artificial intelligence in particular. To develop computing systems working with knowledge, it is necessary to elaborate the means of working with knowledge representations (as opposed to data), because knowledge is an abstract structure. It is important to emphasize the differences between data, data structures, knowledge, and knowledge structures. Data are mental or physical "observables" represented as symbols. Data structures define patterns of the relationships between data items. Knowledge is a system that includes data enhanced by relations between the data and the represented domain. Knowledge structures include data structures abstracted to various systems and inter-object and intra-object relationships and behaviors, which emerge when an event occurs, changing the objects or their relationships. The corresponding state vector defines a named set of knowledge structures, a representation of which is illustrated in Figure 4.

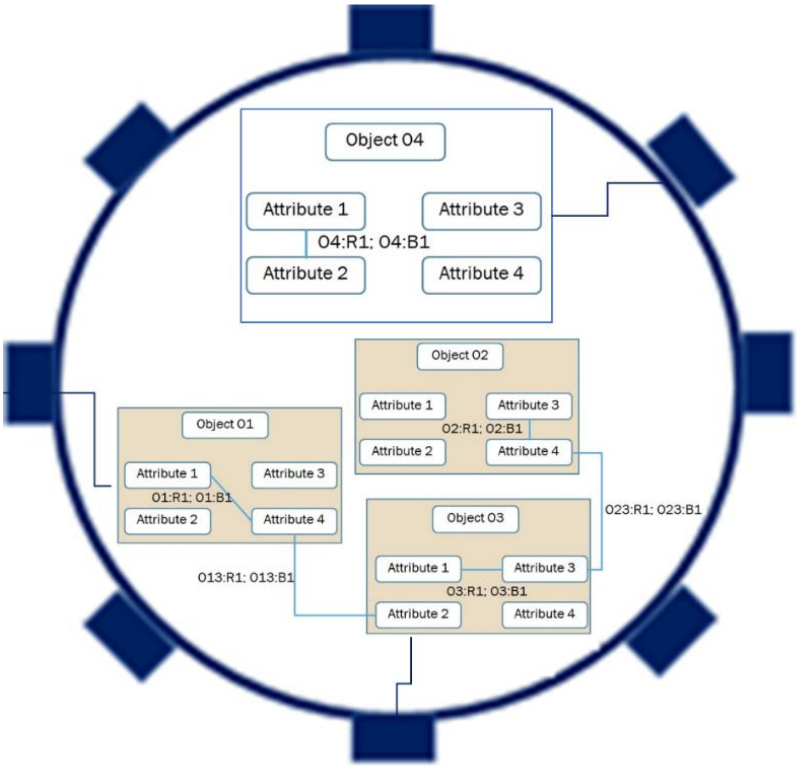

Figure 4. Knowledge structure composed of fundamental triads

This knowledge structure contains four objects, their internal parameters, their relationships and behaviors when new information changes the state. In addition, the inter-object communication, relationships and behaviors are also depicted. The composed entity provides the communication ports to other knowledge structure nodes. The knowledge structure thus represents the system state and its evolution as a complex multi-layer network where the wired nodes fire together to execute the global behavior. The knowledge structure in each node maintains its state and evolution using conventional processors.

Structural Machines: To be able to efficiently process knowledge, a computer or network system must have knowledge-oriented architecture and assembly of operations. The most powerful and at the same time, flexible, model of computing automata is a structural machine [14]. It provides architectural and operational means for operation with schemas.

Figure 5 shows the structural machine operating on the knowledge schema.

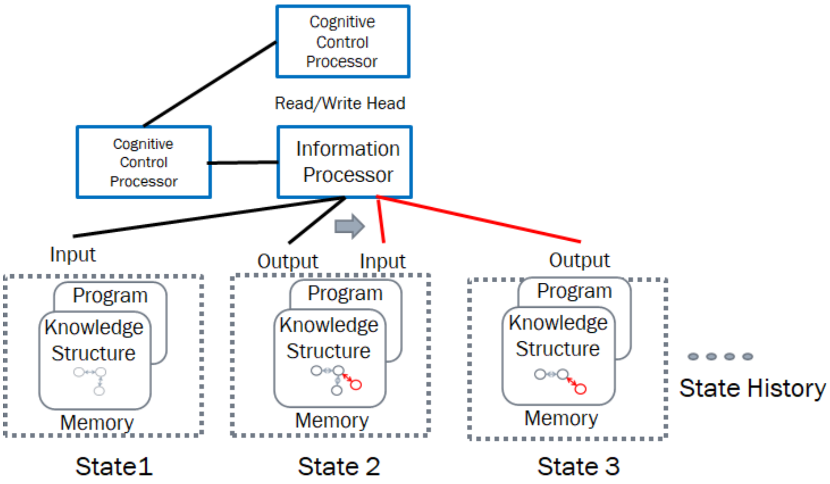

Figure 5. A structural machine operating on knowledge structure schema

A structural machine is an abstract automaton that works with structures of a given type and has three components:

1. The unified control device CM regulates the state of the machine,
2. The unified processor performs transformation of the processed structures and its actions (operations) depend on the state of the machine and the state of the processed structures
3. The functional space which, in turn, consists of three components:
   - The input space which contains the input knowledge structure.
   - The output space which contains the output knowledge structure.
   - The processing space in which the input knowledge structure is transformed into the output structure.

Oracles and Self-Managing Computing Structures: Living organisms are self-managing systems with autopoietic and cognitive behaviors. Self-management by definition implies two components in the system: the observer (or the Self) and the observed (or the environment) with which the observer interacts by monitoring and controlling various aspects that are of importance to the observer. It also implies that the observer is aware of systemic goals to measure and control its interaction with the observed. In living organisms, the self-management behavior is attributed to the sense of self and to its awareness, which contribute to defining one's multiple tasks to reach specific goals within a dynamic environment and adapting the behavior accordingly. The concepts of self and self-reflection which model the observer, the observed and their interactions are quite different from a third party observation of a set of actors or agents who monitor and control the system. The oracles in software networks [19], an extended concept of a Turing Oracle introduced in the Turing O-Machine, in the context of the theory of Oracles provide means to introduce both autopoietic and cognitive behaviors into the structural machines going beyond the capabilities of the observer.

The controller in the structural machine allows the implementation of a cognizing agent or the Oracle overlay that manages the downstream processors and associated knowledge structure evolution. According to the theory of Oracles, an agent system can work as an oracle, collecting information for the basic machines or for a more complex information processing devices, such as a network or computer cluster. For instance, in the theory of superrecursive algorithms and hypercomputation [9], computation, an important type of oracle contains information on whether a Turing machine halts when it is given a definite input or not. As we know in this case, even simple inductive Turing machines are able to compute i.e., obtain this and other information that is incomputable, , i.e., unreachable by Turing machines [9]. Consequently, it is possible to use inductive Turing machines as oracles for bringing information that is incomputable by Turing machines and other recursive algorithms information to various computing devices such as Turing machines or neural networks. In essence, the Oracles provide a means to abstract knowledge in the form of a state and its evolution based on the information provided by various other sources such as symbolic computing, sub-symbolic computing structures, common sense and other domain specific ontologies along with their behaviors.

Triadic Automata and Autopoietic Machines with Cognitive Behaviors: Current day digital information processing structures include hardware which provides the resources for computation (CPU, memory) and software that contains the algorithm in the form of a program which operates on data structures. The data structures represent the state of the system. The stability of the computing structures and the evolution of the state depends on the operation and management of both the hardware and software components. Triadic automata provide the means to include hardware and software behaviors in the knowledge representation to evolve the system as unit by including a new component called infware. The processed data structures constitute infware. The picture shows the three level structures that model both the computer and the computed to provide both autopoietic and cognitive behaviors.

Physical devices contain the hardware structures (computers, network, storage, sensors, actuators etc.). Process management and control provides the evolution of the physical structures. Software contains algorithms, programs and applications. Process management and control provides the evolution of the software structures. Infware contains data structures, information structures and the knowledge Structures. Process management and control provides the evolution of the structures based on the interactions within the structures and with their environment. Figure 6 shows the triadic automata.

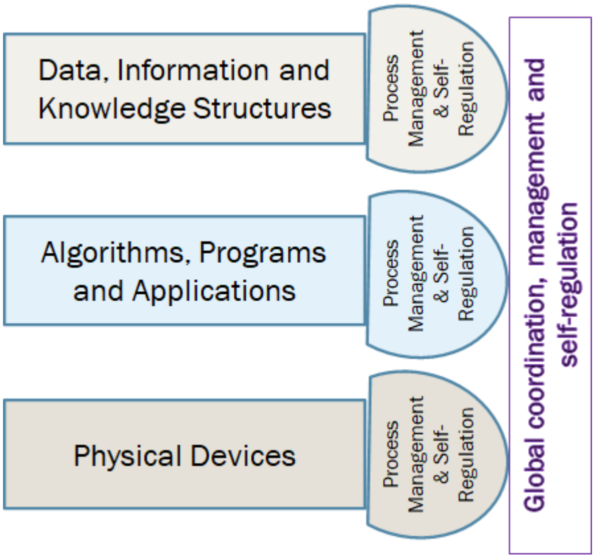
Data, Information and Knowledge Structures
Process Management & Self-Regulation
Algorithms, Programs and Applications
Process Management & Self-Regulation
Physical Devices
Process Management & Self-Regulation
Global coordination, management and self-regulation

Figure 6. Triadic automata managing hardware, software and the infware.

In the next section, we discuss how to implement the structural machine using current state of the art symbolic and sub-symbolic computations and discuss two use cases that exploit the autopoietic and cognitive behaviors.

4. Infusing Autopoietic and Cognitive behaviors into Existing Information Technology Use Cases

GTI tells us [20] that information in the physical world is transformed into knowledge using the abstract mental structures by living organisms and use this knowledge to exhibit autopoietic and cognitive behaviors building the symbolic and sub-symbolic computing processes which exploit the physical and chemical processes converting matter and energy. In addition, it helps us define [20] the digital genome as a collection of knowledge structures [4] coded in executable form to be processed with "structural machines [5], implemented using digital genes (symbolic computing) and digital neurons (sub-symbolic computing). "The digital genome enables digital process execution to discover the computing resources in the environment, use them to assemble the hardware, cognitive apparatuses in the form of digital genes and digital neurons and evolve the process of sentient, resilient, intelligent and efficient management of both the self and the environment with embedded, embodied, enacted and extended (4E) cognitive processes. The digital genome incorporates the knowledge in the form of hierarchical intelligence with a definition of the sentient digital computing structures that discover, monitor and evolve both the self and the interactions with each other and the environment based on best practices infused in them. The digital genome specifies the execution of knowledge networks using both symbolic computing and sub-symbolic computing structures. The knowledge network consists of a supersymbolic network of symbolic and sub-symbolic networks executing the functions defined in their components. The structure provides the system behavior and evolution maintaining the system's stability in the face of fluctuations in both internal and external interactions. The digital genome encapsulates both autopoietic and cognitive behaviors of digital information processing structure capable sentience, resilience and intelligence."

In this section, we propose the design of a digital genome to infuse autopoiesis and cognition into an application deployed in a public cloud using locally available infrastructure as a service (IaaS) and platform as a service (PaaS). IaaS provides the CPU and memory as the fuel for executing the software algorithm. It is usually provisioned using computing hardware and operating system which provide the required resources that are managed to meet appropriate availability, security, performance and other non-functional requirements. PaaS provides the middleware and other tools required to manage the data structures that the algorithm operates on. These include components such as data bases, file systems, web servers, application servers etc., and the associate libraries.

Figure 7 shows various components that are required to infuse autopoietic and cognitive behaviors which provide the application with self-regulation capabilities with a sense of self-awareness and the knowledge to obtain resources required from various sources and use them to deploy, operate and manage both the self and the resources to manage stability and successful execution of the application intent.

Just as the genome in living organisms contains the information required to manage the life processes, the digital genome contains all the information in the form of knowledge structures to build itself, reproduce itself and maintain its stability while using its cognitive processes to interact with the outside world whether it is the hardware infrastructure it uses to build itself or it is the business of connecting people, things and business services. The input to the genome is the information about where to get the resources, where to get the software components, and the "life" processes in the form of knowledge structures and autopoietic and cognitive behaviors to build itself, replicate and deploy various components and manage their stability.

We depict here a functional node which uses a cloud IaaS services to create the fuel for the computation, the PaaS services for providing the software components required for the application workloads to operate on the data congaing the system state. Here we show the containers and their management to provide the local management of the IaaS, PaaS and the application workloads as shown in the functional node and the digital node.

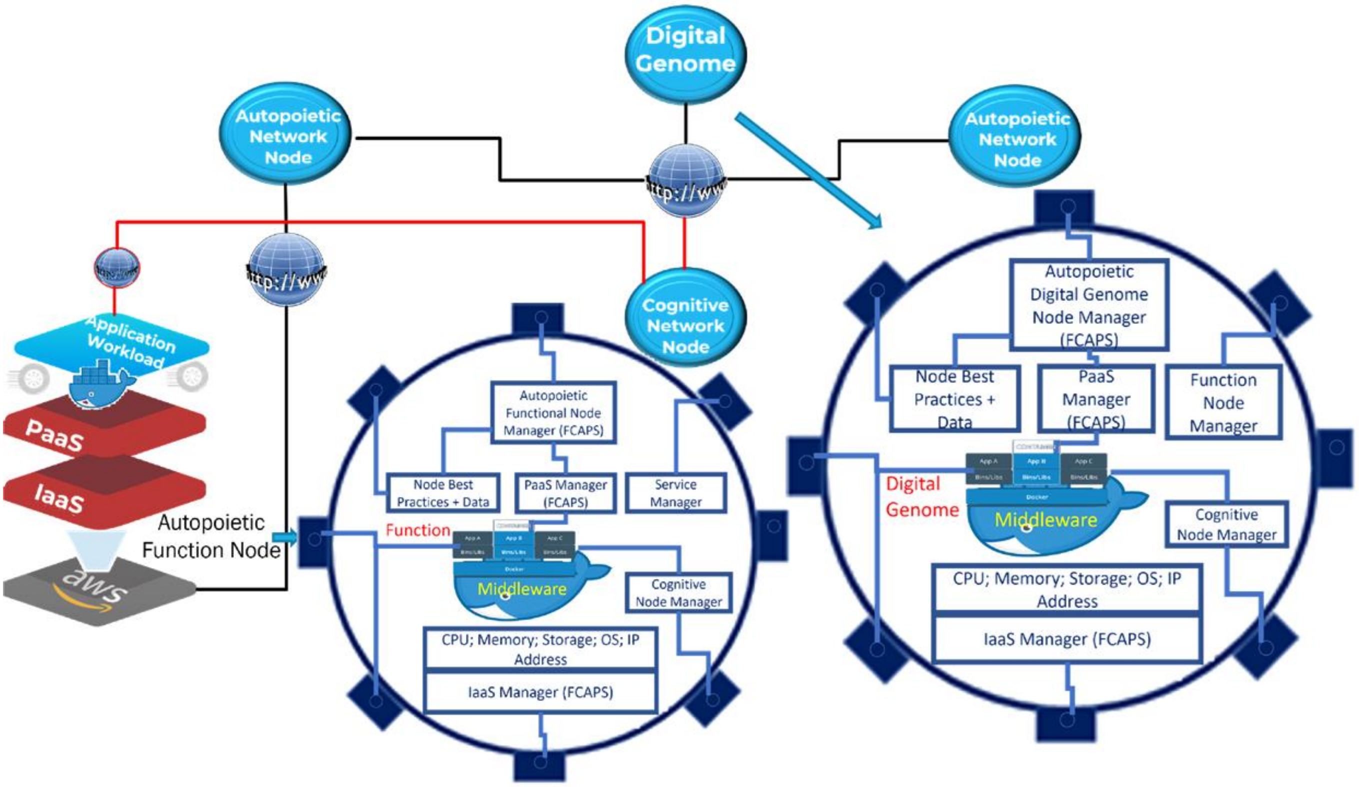

Figure 7. Infusing autopoiesis and cognition into a docker containerized application in a public cloud. (See the video for how autopoiesis and cognition improve sentience, resilience and intelligence of application deployment, operation and stability (https://youtu.be/WS6cfFN4X3A)

The triadic automata [5, 6] are implemented using the structural machine composed of five components:

1. **The digital genome node:** Contains the cognitive "life" processes, the autopoietic "life" processes to configure, monitor and manage the downstream resources (e.g., IaaS, PaaS and application component workloads which include both programs in executable form and associated data). Operations defined [5, 6] in the digital genome provide the required behaviors such as replication, transcription and translation to deploy downstream networks and their component functions just as cellular organisms do. By "life" processes we mean the structure, its initial state information, knowledge about resources, and processes to configure, monitor and manage based on known best practice knowledge encapsulated in executable form. Examples are, executable programs to configure, monitor and manage IaaS, PaaS resources and application business process executables as workloads and associated data.
2. **Autopoietic Network Node**: Contains the knowledge about the autopoietic "life" processes that deal with configuring, monitoring and managing the resources using replication, transcription of policies and best practices to execute them. The autopoietic network node contains the knowledge structures that configure, monitor and manage downstream autopoietic function nodes that are designed to configure, monitor and manage the local resources.
3. **Autopoietic Function Node**: Contains the knowledge about the autopoietic "life" processes that deal with configuring, monitoring and managing the local resources using replication, transcription of policies and best practices to execute them. For example, if the resources are available in a public cloud, the autopoietic node deploys the executable programs that use local management programs

such as Kubernetes and containers or scripts that provision, monitor and manage the resources. Figure 7 shows the two modules that are autopoietic knowledge structures which are designed to manage local IaaS and PaaS resources. The IaaS autopoietic managers configures, monitors and manages the cloud IaaS services using various scripts or tools. The PaaS autopoietic manager configures, monitors and manages PaaS services on local environment. For example Dockers and Kubernetes are configured and managed in the local environment using local scripts or tools. The knowledge structures in these nodes are designed to capture the local resources and their management. Databases, webservers and other middleware are examples of PaaS managed by the autopoietic PaaS function node.

4. **Cognitive Network Node**: Contains the knowledge about the application specific "life" processes that deal with configuring, monitoring and managing the application workloads and associated data using replication, transcription of policies and best practices to execute them. The Cognitive network node contains the knowledge structures that configure, monitor and manage downstream application components as cognitive function nodes that are designed to configure, monitor and manage the application executables and associated data. For example, the cognitive function has the knowledge about the components of the application such as web component, application logic component and database component, and where to deploy them, monitor them and manage them based on best practice policies when fluctuations point to potential instabilities. When a local component fails, the cognitive function node has the knowledge to recover with auto-failover or configure based on RPO and RTO to recover without service description or auto-scale to maintain stability using replication and managing the data to maintain service while replicating. Etc.
5. **Cognitive Function Node**: Contains the knowledge about configuring, monitoring and managing the local application function using the PaaS and IaaS resources made available with autopoietic function nodes.

In essence, autopoietic and cognitive processes encapsulate the knowledge about an application, create a sense of "self" and selects and uses the "life" processes encoded and made available in the genome to maintain stability while the application interacts with the environment (users, devices, other applications etc.)

5. Conclusion

We have presented here an application of GTI to improve sentience, resilience and intelligence of digital information processing systems. While GTI has been widely discussed in literature, its application to infusing autopoiesis and cognition into digital automata is still in its infancy. Some universities and startups are starting to build proofs of concept and prototypes and will soon make their way into publications and products with new approaches. The theory provides a foundational breakthrough and facilitates implementation without disrupting the current state of the art. Just as the mammalian brain repurposed and reused the reptilian cognitive capabilities in living organisms with a common knowledge representation including the self and the environment, super-symbolic computing provides a knowledge network integrating knowledge from symbolic and sub-symbolic computations. Knowledge nodes wired together fire together to execute both autopoietic and cognitive behaviors.

As pointed out by Burgin and Mikkilineni [21], "This is only one example demonstrating that theoretical computer science is a cornerstone for computer science as a whole and computer technology as its practical application [22]. Theoretical computer science based on mathematical structures and procedures "underlies many aspects of the construction, explanation, and understanding of computers" [23]."

As far as the related work, the author believes that GTI, while it provides a unified theory of information processing structures, has not been well understood or embraced by the classical computer scientists and current state of the art information technology professionals. GTI encompasses various aspects from philosophy, mathematics, information science, biology and the science of complex adaptive systems and provides a unified theory to bring them together. In addition, it suggests new approaches to go beyond current state of the art symbolic and sub-symbolic computing practices to address some crucial limitations discussed in this paper. Only time will tell whether this is a foundational breakthrough that provides new approach to building new class of machines that mimic the autopoietic and cognitive behaviors of living organisms or it is yet another theory that no one pays any attention.

We conclude this paper with this quote attributed to Leonardo da Vince ""He who loves practice without theory is like the sailor who boards ship without a rudder and compass and never knows where he may cast."

Funding: "This research received no external funding.

Acknowledgments: Author wishes to thank prof. Mark Burgin for various in-depth discussions on general theory of information and the theory of structural machines.

Conflicts of Interest: "The authors declare no conflict of interest."

References

[1] Von Neumann, J. (1966). In A. W. Burks (Ed.), Theory of self-reproducing automata. Urbana: University of Illinois Press. p. 71.

[2] Copeland, B. Jack, "The Church-Turing Thesis", The Stanford Encyclopedia of Philosophy (Summer 2020 Edition), Edward N. Zalta (ed.), URL = <https://plato.stanford.edu/archives/sum2020/entries/church-turing/>.

[3] Burgin, M. Theory of Information: Fundamentality, Diversity and Unification, World Scientific: Singapore, 2010.

[4] Burgin, M. Triadic Automata and Machines as Information Transformers, Information, v. 11, No. 2, 2020, 102; doi:10.3390/info11020102

[5] Burgin, M., Mikkilineni, R. and Phalke, V. Autopoietic Computing Systems and Triadic Automata: The Theory and Practice, Advances in Computer and Communications, v. 1, No. 1, 2020, pp. 16-35

[6] Burgin, M. and Mikkilineni, R. From Data Processing to Knowledge Processing: Working with Operational Schemas by Autopoietic Machines, Big Data Cogn. Comput. 2021, v. 5, 13 (https://doi.org/10.3390/bdcc5010013)

[7] Burgin, M. and Mikkilineni, R. Symbiotic information processing: A methodological analysis, "Theoretical and Foundational Problems in Information Studies" (TFP), is4si Summit 2021, Online, 12–19 September 7, 2021.

[8] P. Cockshott, L. M. MacKenzie and G. Michaelson, "Computation and Its Limits," Oxford University Press, Oxford, 2012, p. 215.

[9] Burgin, M. Superecursive Algorithms; Springer: New York, NY, USA, 2005.

[10] Gödel, K. Über formal unentscheidbare Sätze der Principia Mathematics und verwandter System I, Monatshefte für Mathematic und Physik, 38, 173-198, 1931.

[11] Turing, A. M. Systems of logic defined by ordinals. Proc. Lond. Math. Soc., Ser. 2, 45, pp. 161-228, 1939.

[12] Gilder, George. Life After Google: The Fall of Big Data and the Rise of the Blockchain Economy (pp. 25-26). Gateway Editions. Kindle Edition.

[13] Burgin, M. and Mikkilineni, R. From Symbolic Computation to Super-Symbolic Computation, EasyChair Preprint no. 6559, 2021.
https://easychair.org/publications/preprint_open/PMjC

[14] Mikkilineni, R.; Burgin, M. Structural Machines as Unconventional Knowledge Processors. *Proceedings* **2020**, *47*, 26. https://doi.org/10.3390/proceedings2020047026

[15] Burgin, M. The General Theory of Information as a Unifying Factor for Information Studies: The Noble Eight-Fold Path. *Proceedings* **2017**, *1*, 164. https://doi.org/10.3390/IS4SI-2017-04044

[16] Burgin, M. Theory of Named Sets; Nova Science Publisher Inc.: New York, NY, USA, 2011.

[17] Burgin, M. Theory of Knowledge: Structures and Processes; World Scientific Books: Singapore, 2016

[18] Burgin, M. Data, Information and Knowledge. Information 2004, 7, 47–57.

[19] Mikkilineni, R., Morana G., and Burgin, M. "Oracles in Software Networks: A New Scientific and Technological Approach to Designing Self-Managing Distributed Computing Processes", Proceedings of the 2015 European Conference on Software Architecture Workshops (ECSAW '15). ACM, 2015.

[20] Burgin, M. and Mikkilineni, R. On the Autopoietic and Cognitive Behavior, EasyChair Preprint no. 6261, version 2., 2021.
https://easychair.org/publications/preprint/tkjk

[21] Burgin, M. and Mikkilineni, R. (2018) 'Cloud computing based on agent technology, super-recursive algorithms and DNA', Int. J. Grid and Utility Computing, Vol. 9, No. 2, pp.193–204.

[22] Savage, J.E., Selman, A.L. and Smith, C. (2001) 'History and contributions of theoretical computer science', Advances in Computers, Vol. 55, pp.171–183.

[23] Funding a Revolution (1999) Government Support for Computing Research, Committee on Innovations in Computing and Communications: Lessons from, History, National Research Council, p.184.