Designing a New Class of Digital Autopoietic Machines

Rao Mikkilineni[1] and Mark Burgin[2]

[1] Ageno School of Business, Golden Gate University, San Francisco, CA 94105, USA. rmikkilineni@ggu.edu
[2] UCLA, Los Angeles, CA 90095, USA

Introduction

Function, structure and fluctuations play key roles in how physical, chemical and biological systems behave as a result of the interactions among their constituent components and their environment. Digital computing structures composed of distributed and communicating software and hardware components also, fall into the category of a complex system where fluctuations in the demand for, or the availability of, resources required to execute the computations disturb their stability and performance. The fluctuations impact the resiliency and efficiency of the structure as the scale of components increase. In addition, when the system consists of a set of communicating processes, with well-defined functional behaviors, that are concurrent and asynchronous, the emergence of global behavior depends upon the nature and the strength of the fluctuations. In this paper, we describe the theory and practice of applying the self-organizing and self-managing patterns to manage fluctuations, found in biological systems, to distributed digital computing structures and making them autopoietic machines. The term autopoiesis refers to a system with a well-defined identity and is capable of reproducing and maintaining itself.

The Theory and Practice of Information Processing Structures

Information unit is described by the existence or non-existence (1 or 0) of an entity or an object that is physically observed or mentally conceived. The difference between an entity and an object is that the entity is an abstract concept with attributes such as a computer with memory and CPU. An object is an instance of an entity with an identity, defined by two components which are the object-state and object-behavior. An attribute is a key value pair with an identity (name) and a value associated with it. The attribute state is defined by its value.

Information is related to knowledge and data as energy is related to matter. Knowledge is defined [1] by the relationships between various entities and their interactions (behaviors) when the values of the attributes change. A named set is a fundamental triad that defines the knowledge about two different entities.

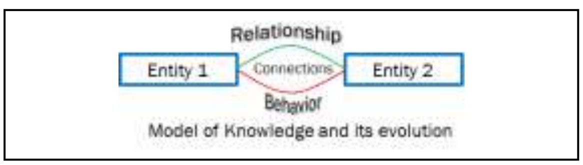

Figure 1: The fundamental triad as a knowledge structure derived from information

A knowledge structure defines various triadic relationships between all the entities that are contained in a system. The knowledge structure provides the schema and various operations to evolve the schema from one state to another. Various instances of the knowledge structure schema are used to model the domain knowledge and process information changes as they evolve with changes in their entities and their attributes and behaviors.

The structural machine is an information processing structure that represents the knowledge structures as schema and performs operations on them to evolve information changes in the system from one instant to another when any of the attributes of any of the objects change.

The structural machines supersede the Turing machines by their representations of knowledge and the operations that process information [2, 3]. Triadic structural machines with multiple general and mission-oriented processors. enable autopoietic behaviors.

1. From Turing Machines to Structural Machines **[2, 3]:**

Structural machines process *knowledge structures which incorporate domain knowledge in the form of entities, their relationships and process evolution behaviors as a network of networks* with each node defining functional behaviors and links defining the information exchange (or communication). The operations on the knowledge structure schema define the creation, deletion, connection and reconfiguration operations based on control knowledge structures. They are agnostic to what the functions of the nodes or what information is exchanged between them. This provides the composability of knowledge structures across domains in processing information. In contrast, the Turing machines process data structures which incorporate domain knowledge in the form of entities and relationships *only and their process evolution behaviors are encapsulated in algorithms (programs) which operate of the data structures*. Therefore, the Turing machine operations are domain knowledge specific and lacks composability across domains and increases complexity in processing information and its evolution.

2. Changing system's behaviors using functional communication **[3, 4]:**

The behavioral changes are embedded in the knowledge structures and therefore functional communication or information exchange induces the behavioral changes in various entities in the knowledge structures. Changes are propagated through knowledge structures when events produce changes in arbitrary attributes of the system entities. This enables self-regulation of the system. In contrast to self-regulation, the external control causes the behavioral changes by the rules embedded in the algorithms outside the data structures and to execute the behavioral changes, the programs have to understand the domain knowledge of the data structures in order to perform operations on them.

3. ***Triadic structural automata and autopoietic behavior*** **[3, 4]:**

A triadic structural machine with hierarchical control processors provides the theoretical means for the design of auto-poietic automata allowing transformation and regulation of all three dimensions of information processing and system behavior – the physical, mental and structural dimension. The control processors operate on the downstream information processing structures, where a transaction can span across multiple distributed components by reconfiguring their nodes, links and topologies based on well-defined pre-condition and post-condition transaction rules to address fluctuations; for example, in resource availability or demand.

4. Providing global optimization using shared knowledge and predictive reasoning to deal with large fluctuations [5]:

The hierarchical control process overlay in the design of the structural machine, allows implementing 4E (embedded, embodied, enactive and extended) cognitive processes with downstream autonomous components interacting with each other and with their environment using system-wide knowledge-sharing, which allows global regulation to optimize the stability of the system as a whole based on memory and historical experience-based reasoning. Downstream components provide sensory observations and control using both neural network and symbolic computing structures.

These insights allow us to design a new class of information processing systems with higher efficiency, resiliency and scalability in dealing with fluctuations going far beyond the capabilities possible for information processing systems based on the Turing machine model. In our presentation, utilization of this theory for building self-managing federated edge cloud network deploying autopoietic federated AI applications to connect people, things, and businesses for enabling global communication, collaboration and commerce with high reliability, performance, security, and regulatory compliance.

References

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

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

[3] 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

[4] 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)

[5] Mikkilineni, R. Information Processing, Information Networking, Cognitive Apparatuses and Sentient Software Systems. *Proceedings* **2020**, *47*, 27. https://doi.org/10.3390/proceedings2020047027