The Matrix Theory of Objects. An Update

Sergio Pissanetzky, Research Scientist. Member, AAAI, IEEE.
Sergio@SciControls.com

ABSTRACT

Objects are logical structures of information that the brain naturally makes and uses to think, communicate, even dream. But the process for making objects is not understood. This theory explains the process. The first postulate of the theory is that *Every system of information has a structure, and that this structure consists of a hierarchy of objects and their inheritance relationships.*

The *model* for the theory is the *Matrix Model of Computation* (MMC),[1] which consists of a single *canonical* matrix where *services* are represented in the rows and the *variables* they use in the columns. A service is like a subroutine, it takes arguments, marked with A's in the lower triangle of the matrix, and returns a variable, marked with a C on the diagonal. The presence of an A also identifies a *constraint*, because a service that uses an argument can't execute until that argument's initializing service has completed.

The model is simple, yet extremely rich in properties. It is *universal*: any system of information can be represented in MMC form. Theorems of equivalence with the Turing machine and the Universal Quantum Computer have been published. Services are executable, the matrix is a computer program, it can execute. Services represent *behavior*, variables represent *information.*

But there is a degree of freedom: the order of the services is not fixed. Row and column permutations are possible, which preserve both the canonicity of the matrix and the behavior of the system. A very large number of legal configurations exist. Actually, nothing keeps the services attached to their positions. They are free to move, so why wouldn't they? The theory allows legal symmetric row and column permutations to occurr spontaneously. This makes the system a *dynamic* one. Services behave like molecules in a gas, they constantly move and change their positions, without breaking the constraints. But this is *chaos*! The second postulate is: *Intelligent systems of information are chaotic.* Since behavior is not affected, behavior and chaos are orthogonal processes.

But there is a *cost* to chaos: information has to travel long distances. It flows through long, intersecting paths that resemble turbulence in a fluid. The point where a variable receives information from a service is marked on the diagonal of the matrix with a C. From the C, the information travels down a column to an argument of another service, which is marked with an A in that column, and then along a row to another C where it is used to initialize another variable. If the A's are far from the diagonal, the paths are long and convoluted, and the system is poorly organized.

The theory proposes the *Scope Constriction Algorithm* (SCA) as the first step for finding the objects. SCA takes advantage of the fact that the distribution of the A's in the lower triangle depends on the configuration. SCA simulates the chaotic dynamics by randomly applying legal symmetric permutations to the matrix, and introduces *dissipation* by systematically selecting the configurations that bring the A's closer to the diagonal and minimize the cost function. This process is clearly controlled by the distribution of the A's. But the A's represent constraints. The third postulate is: *The structure of a system of information is encoded in the constraints of the system.*

SCA permutations also cause similar services and variables to cluster together and form units that *encapsulate* the behavior of services and the information in variables. These units are a property of the system under analysis and represent the *objects* in it. Furthermore, when a portion of the phase space of a dissipative chaotic system "wanders away" and converges to a stable configuration, that configuration is considered to be a property of the system and said to be an *attractor*. Hence the fourth postulate: *The attractors of a system of information contain the objects.*

The SCA process is statistical, and the result is a statistically significant sample of optimized configurations. But the objects are still encoded in the sample. To decode the objects, the theory proposes the Object Recognition Algorithm (ORA). ORA effectively finds the attractors in the sample, and identifies the structure of objects and the inheritance hierarchies. To do that, ORA uses the single theoretical property that characterizes all objects: their invariance under the statistics. ORA seeks subsets of services that are present in all configurations in the sample, and concludes the search by identifying them as objects.

Every theory needs experimental evidence. What do these "mathematical objects" have to do with the "natural" objects our brains make? They are the same. Limited but very convincing experimental results indicate so. So far, I tested several small examples and two case studies, one of them still unpublished. Fortunately, there exists a huge repository of extensive and carefully documented observations: the object-oriented programs developed in the last 3 decades, covering virtually every imaginable application of human thought and experience. These programs contain man-made objects. It would suffice to apply MMC techniques to the same problems and compare the results. This has been done in at least two cases, on a small scale, and the results are excellent.

I conjecture that the brain applies the Matrix Theory to make its objects. Perhaps some day the roots of SCA and ORA will be found in our DNA. I conjecture that *understanding* is the process where the brain makes objects from what it has learned, and then tests them by making them function. I also conjecture that the ability to make objects is required for machines to achieve human-level artificial intelligence. Our brains use objects for nearly everything. In comparison, after decades of research, intelligent machines are limited and progress is very slow.

I expect the Matrix Theory to be accepted as a major theory. The MMC model applies to systems of all kinds. I propose that all systems should be described in MMC format, for the simple reason that they can then be automatically organized, or re-organized when changes arise. And I anticipate that MMC organizers will soon be as common as word processors and compilers are today. Why is the MMC comparable to humans? Because it can learn, just like schoolchildren, and because it can understand by making objects, just like we do. Why is it better than humans? 1. Copies can be made. 2. It's modular. Many MMC's can be trained on different subjects and then put together. They integrate their knowledge by training each other. 3. Features and hardware can be added when needed to make it more powerful. 4. Non-human sensors such as radar or satellite can be integrated. 5. Of course, the MMC can go where humans can't. I strongly suggest that large-scale work should start immediately.

1 - "A new Universal Model of Computation and its Contribution to Learning, Intelligence, Parallelism, Ontologies, Refactoring, and the Sharing of Resources." Sergio Pissanetzky. Int. J. of Computational Intelligence, Vol. 5, No. 2, pp. 143-173 (August 2009).