A Domain Specific Language for Complex Natural and Artificial Systems Simulations

Jean-Louis Giavitto

CNRS – IBISC, Université d’Evry, Genopole,
523 place des Terrasses de l’Agora, 91000 Evry, France

A DSL for Systems Biology. Domain specific languages are often designed to incorporate domain-specific knowledge in order to enhance expressiveness (for the programmer) and quality, flexibility, maintainability, ..., of the produced softwares. In this talk I introduce a language initially developed to ease the modeling and the simulation of developmental processes in biology. In this application domain, one must face:

- dynamical processes that are located and move in space,
- processes that interact locally with their (spatial) neighbors,
- a spatial neighborhood that is build (computed) and adjusted as a result of the process activities,
- various style of processes (numerical simulation of ODE and PDE, stochastic processes and discrete deterministic or non-deterministic transition systems).

Ideally, the specification of the biological models should be small and expressive, theoretically well founded and close to the concepts used by the modelers.

These requirements lead to the design of a rule based programming language called MGS. MGS is based on a notion of spatial n-ary interaction: such interaction represents by a rule the local evolution of a small subsystems. The structure of the subsystems is represented through topological relationships and is subject to possible drastic changes in the course of time.

MGS: Topological Rewriting. An MGS rule can also be interpreted as a rewriting rule on topological chains. A topological chain is a mathematical structure introduced in algebraic topology and used, e.g. to compute the boundary of an object composed of several elementary pieces of space glued together. With a sufficiently abstract notion of space, MGS subsumes several kind of rewriting (multiset rewriting, word rewriting) and several bio-inspired models of computation (cellular automata, Lindenmayer systems, membrane systems). And a “dictionary” can be developed to link notions relevant to rewriting systems and those relevant to the simulation of dynamical systems:
Algorithmic Examples. MGS has been validated on several large scale examples in the simulation of cellular processes in computational biology. However, the notions of topological collection and transformation initially introduced to ease the modeling of dynamical systems with a dynamical structure have proven to be useful in algorithmic tasks and we will present during the talk several examples.

MGS rules can represent arbitrary complex computations because the pattern language is very sophisticated. However, in a lot of applications, it appears that the rules corresponds to chain homomorphisms (i.e. transformations that respect the algebraic structure of chains). The computational content of a chain homomorphism is clear: it corresponds to a simple pattern of distributed computations and hence can be viewed as a skeletons that package useful and reusable patterns of parallel and distributed computations.

Autonomic Systems, Dynamical Systems and Rule Based Programming. Recently, the programming style promoted by MGS has been advocated as well suited to express autonomic properties (the so called self-* properties): the reaction rules correspond to the local actions to be taken to react to a perturbation.

The approach can be described as “autonomic computing via trajectory stabilization”. In this point of view, an autonomic system is seen as a distributed dynamical system. In the diagram above, the states of this dynamical system are

\[
\begin{array}{|c|c|}
\hline
\text{Rewriting Systems} & \text{Dynamical Systems} \\
\hline
\text{term} & \text{state} \\
\text{set of rules} & \text{evolution function} \\
\text{derivation} & \text{trajectory} \\
\text{rule application strategy} & \text{management of time} \\
\text{normal form} & \text{steady state, fixed point} \\
\cdots & \cdots \\
\hline
\end{array}
\]
figured as the ground plane and the system’s evolutions are given by a trajectory. The surface represents some potential function, for instance a quantitative evaluation of the divergence of the system from a desired behavior.

When some transient perturbations make the system leave its steady state, the local transformations triggered by the matching of some rules eventually lead to the return of the system’s state to an admissible state.

Acknowledgments. Olivier Michel and Antoine Spicher at the University of Creteil are gratefully acknowledged for our long standing collaboration. They have made the MGS project possible and fruitful. Thanks are also due to Pascal Fradet at INRIALPES, Jean-Pierre Banâte at IRISA, Christophe Godin at INRIA-Montpellier, Annick Lesne at IHES and Przemyslaw Prusinkiewicz at University of Calgary for stimulating discussions and interactions. The works presented here have been partially funded by the University of Evry, the Genopole, the CNRS, the ANR projects NanoProg and AutoChem, and the Paris Institute of Complex System.

References