Dynamic models of bacterial gene expression in the SpiCO language

Céline Kuttler[1], Cédric Lhoussaine[2], and Joachim Niehren[3]

***Short Abstract* — We present SpiCO, a novel modeling language for systems biology. It is indeed a programming language founded on the pi-calculus, a rigorous mathematical framework. The SpiCO language offers several benefits to model engineering, compared to the initial formalism. Its elaboration was driven by two quantitative modeling studies. The first focused on the molecular level of transcription initiation at the lambda switch. A second dealt with the general machinery of bacterial transcription and translation. Dynamic model execution yields continuous time Markov chains (CTMCs), or simulation trajectories via the Gillespie's algorithm.**

***Keywords* — modeling, programming language, gene expression, gene regulation, concurrency, pi-calculus, stochastic simulation, Gillespie algorithm.**

I. Regulatory Dynamics as Concurrent Computation

Quantitative modeling of gene regulatory dynamics raises multiple challenges. Gene expression is controlled by the orchestrated action of a multitude of molecular actors. The view as a sequence of independents phases has been obsoleted. Phases in the pathway from gene to protein may initiate before the preceding has completed [1-2]. The inherent control, coupling, and interdependencies in genetic regulatory systems surpass the expressiveness of many established modeling approaches.

Biological researchers pioneered in resorting to methods from computer science to represent the dynamics between biomolecules [3]. They pointed out commonalities between cellular networks and concurrent computational systems. *Concurrency* is a key aspect of complex computational systems, from operating systems, over distributed databases to the Internet. Sophisticated control and dependencies render these challenging to design, build, and maintain.

The *pi-calculus* is a widely accepted framework that captures fundamental principles of concurrency through minimal linguistic means, with a precise mathematical meaning [4].

II. Model Engineering in the SpiCO Language

We designed the SpiCO [5] language design based on our experience from two modeling studies, that contributed thorough quantitative models of bacterial gene expression and regulation in the pi-calculus [6-7]. One strength of the pi-calculus lies in formal investigation of miniature systems, yet it poorly supports larger scale model design. SpiCO seeks to bridge this methodological gap.

A. Molecules as concurrent objects with multiple profiles

Concurrent objects are at the heart of SpiCO models, representing individual molecules and their behavior. They explicitly render alternative discrete states of molecules. Each is ascribed a distinct profile,, in which the object offers distinct interaction capabilities. Objects switch profiles as a result of interaction with others and information processing.

B. Model refinement and reuse

We apply established software engineering techniques to re-use and refine existing SpiCO model components. A simple module system permits the former, inheritance relations between object classes the latter. These features were not available in previous pi-calculus based modeling approaches.

C. Quantitative interpretation of SpiCO models

SpiCO models map to continuous time Markov chains, and yield stochastic simulation via Gillespie's algorithm [8].

D. Case studies: bacterial gene expression and regulation

Our case studies pioneered in quantitative pi-calculus modeling. The first deals with transcriptional regulation at the genetic switch of bacteriophage lambda [6], the second provides a generic model of unregulated bacterial gene expression [7]. We thoroughly validated all predictions against previous knowledge.

III. Outlook

We wish to apply SpiCO to further cases in gene expression and regulation, to advance its utility as a predictive tool, and to ultimately contribute to better quantitative understanding of regulatory dynamics. The challenges encountered may well necessitate further methodological advances.

References

[1] G. Orphanides and D. Reinberg (2002): A unified theory of gene expression. Cell 108:439-451.
[2] T. Maniatis and R. Need (2002): An extensive network of coupling among gene expression machines. Nature 416:499
[3] A. Regev and E.Shapiro (2002), Nature 419:34.
[4] R. Milner (1999). The pi-calculus. Cambridge University Press.
[5] C. Kuttler, C. Lhoussaine, J. Niehren (2007). A stochastic pi-calculus for concurrent objects. Proc Algebraic Biology 2007, Springer.
[6] C. Kuttler and J. Niehren (2006): Gene regulation in the pi-calculus: simulating cooperativity at the lambda switch. Trans Comp Systems Biology VII, Springer.
[7] C. Kuttler (2006): Simulating bacterial transcription and translation in a stochastic pi-calculus. Trans Comp Systems Biology VI, Springer
[8] D.T. Gillespie (1976), Journal of Computational Physics 22, 404-434.

[1]The Microsoft Research - University of Trento Centre for Computational and Systems Biology, Italy. E-mail: kuttler@cosbi.eu
[2]LIFL, University of Lille, France. E-mail: lhoussai@lifl.fr
[3]INRIA, Lille, France. E-mail: niehhren@lifl.fr