Technical Report CoSBi 10/2007

Computational Thinking in Biology

Corrado Priami
CoSBi
and
DISI, University of Trento
priami@cosbi.eu

This is the preliminary version of a paper that will appear in
Transactions on Computational Systems Biology
available at http://www.springerlink.com/content/p2246002t2568021/

Computational Thinking in Biology

Corrado Priami
The Microsoft Research - University of Trento
Centre for Computational and Systems Biology
priami@cosbi.eu

Abstract

The paper presents a new approach based on process calculi to systems modeling suitable for biological systems. The main characteristic of process calculi is a linguistic description level to define incrementally and compositionally executable models. The formalism is suitable to be exploited on the same system at different levels of abstractions connected through well defined formal rules. The abstraction principle that represents biological entities as interacting computational units is the basis of the computational thinking that can help biology to unravel the functions of the cell machinery. We discuss then the perspectives that process calculi can open to life sciences and the impact that this can in turn produce on computer science.

1 Introduction

Systems level understanding of phenomena has recently become an issue in biology. The complexity of molecular interactions (gene regulatory networks, signaling pathways, metabolic networks, etc.) makes impossible to handle the emergent behavior of systems simply by putting together the behavior of their components. Interaction is a key point in the study of emergence and complexity in any field and hence in biology as well where the molecular machinery inside a cell determines the behavior of complex organisms.

Besides interaction, the other key issue to develop computer-based tools for systems biology is incremental construction of models. We need to add something to a model once new knowledge is available without altering what we already did. This is an essential feature for modeling formalisms being applicable to real size problems (not only in the biological applicative domain). Many approaches have been investigated in the literature to model and simulate biological systems (e.g., ODE or stochastic differential equations, Petri nets, boolean networks, agent-based systems), but most of them suffer limitations with respect to the above issues.

In recent times, programming languages based approaches have been proposed to generate executable models at a linguistic level. We think that they

are a suitable tool to address interaction, incremental building of models and complexity of emergent behavior. As usual in computer science, the definition of a high level linguistic formalism that then must be mapped onto lower level representations to be executed may loose efficiency but gain a lot in expressive power and usability. Being the systems in hand huge, we need such formalisms to minimize the error prone activity of specifying behavior.

The main idea is that computer networks, and Internet in particular, are the artificial systems most similar to biological systems. Languages developed in the last twenty years to study and predict quantitatively the dynamic evolution of these networks could be of help in modeling and analysing biological systems. Recent results show that process calculi (very simple modeling languages including the basic feature to model interaction of components) have been successfully adopted to develop simulators [22, 24, 19] that can faithfully represent biological behavior.

The correspondence between the way in which computer scientists attacked the complexity of artificial systems and the way in which such complexity is emerging in biology when interpreting living systems as information processing unit [13] is very strict. Therefore computational thinking [26] is a tool that can extremely help enhance our understanding of living systems dynamics. Computational thinking expresses the attitude of looking at the same problem at different levels of abstraction and to model it through executable formalisms that can provide insights on temporal evolution of the problem in hand. Therefore the basic feature of computational thinking is abstraction of reality in such a way that the neglected details in the model make it executable by a machine. Of course, different executable abstractions of the same problem exist and the choice is driven by the properties to be investigated. Indeed, science history shows us that a single model for the whole reality does not exist: our modeling activity must be driven by the properties of the phenomenon under investigation that we want to look at.

Process calculi have been originally introduced [15, 12] as specification languages for distributed software systems. The specification can be refined towards an actual implementation within the same formalism. Any refinement step is validated by formal proofs of correctness. This approach is a good example of a framework that imposes the application of the computational thinking and therefore we work on it to obtain a similar framework for biological systems.

We here briefly and intuitively introduce process calculi (in particular we concentrate on the β-language) to show on an example how they can be used to model biological systems. We then investigate the potential of the approach in a perspective vision. We first discuss how life scientists can improve their performance by relying on software and conceptual tools that allow them to mimick the standard activities they perform in wet labs. There are however two main advantages to work in silico: speed and cost. Actually experiments last few minutes instead of hours or days and the cost is extremely reduced. Once the scientist think of having something concrete in silico can move towards the wet lab and test *in practice* the hypotheses. Essentially there is an iterative loop between in silico production of hypothesis and wet testing of them.

The longer term perspective of the approach is related to enhancement of computer science. The knowledge we gain from developing linguistic mechanisms to describe and execute the dynamics of complex biological systems could lead to the definition of a new generation of programming languages and new programming paradigms that can enhance the software production tools now available.

2 Conclusions

The new field of computational and systems biology can have a large impact on the future of science and society. The engine driving the new coming discipline is its inherent interdisciplinarity at the convergence of computer science and life sciences. To continue fueling the progress of the field we must ensure a peer-to-peer collaboration between the scientists of the two disciplines. In fact if one discipline is considered a service for the other the cross-fertilization will stop soon. We must create common expectations and really joint projects in which both computer science and biology can enhance their state-of-the-art.

We must ensure a critical mass of people working in the field and a common language to exchange ideas. This is a major problem in current collaborations due to the lack of curricula that form people to work in this intersection area. We must invest time and resources in creating interdisciplinary curricula (together with new ways of recruiting people considering interdisciplinarity an added value) to form the new researchers of tomorrow.

Summing up, although a lot has still to be done, we started a new way of making science that can lead in the next years to unravel the machinery of cell behavior that in turn can lead to the creation of artificial systems enjoying the properties of living systems. Computational thinking is different way of approaching a problem by producing descriptions that are inherently executable (differently, e.g., from a set of equation). Furthermore the same specification can be examined at different level of abstractions simply by building a virtual hierarchy of interpretations. This a common practice in computer science where artificial systems are usually defined and described in layers depending on the growing abstraction from the physical architecture.

References

[1] M. Calder, S. Gilmore, and J. Hillston. Modelling the influence of RKIP on the ERK signalling pathway using the stochastic process algebra PEPA. *Transactions on Computational Systems Biology*, VII(4230, *LNCS*), 2006.

[2] L. Cardelli. Brane Calculi. In *CMSB '04*, volume 3082 of *LNBI*. Springer, 2005.

[3] L. Cardelli and A. D. Gordon. Mobile ambients. In *Foundations of Software Science and Computation Structures: First International Conference, FOSSACS '98*. Springer-Verlag, Berlin Germany, 1998.

[4] I. Castellani. *Process algebras with localities*, pages 945 – 1046. 2001. In Handbook of Process Algebra, J. Bergstra, A. Ponse and S. Smolka (Eds).

[5] G. Ciobanu and G. Rozenberg, editors. *Modelling in Molecular Biology*. Springer, 2004.

[6] V. Danos and J. Krivine. Formal Molecular Biology done in CCS-R, in: Proceedings of Workshop on Concurrent Models in Molecular Biology (BioCONCUR'03). *Electronic Notes in Theoretical Computer Science*, 2003.

[7] V. Danos and C. Laneve. Formal molecular biology. *TCS*, 325(1), 2004.

[8] A. Finney, H. Sauro, M. Hucka, and H. Bolouri. An xml-based model description language for systems biology simulations. Technical report, California Institute of Technology, September 2000. Technical report.

[9] D.T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

[10] P.J.E. Goss and J. Peccoud. Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. In *Proceedings of the National Academy of Sciences USA*, number 12, pages 6750–6754, 1998.

[11] M.L. Guerriero, C. Priami, and A. Romanel. Beta-binders with static compartments. In *Algebraic Biology 2007*, 2007. To appear. Also TR-09-2006. The Microsoft Research - University of Trento Centre for Computational and Systems Biology.

[12] C. A. R. Hoare. Communicating sequential processes. *Communications of the ACM*, 21(8):666–677, 1978.

[13] L. Hood and D. Galas. The digital code of DNA. *Nature*, 421:444–448, 2003.

[14] C. Kuttler and J. Niehren. Gene regulation in the pi-calculus:simulating cooperativity at the lambda switch. *Transactions on Computational Systems Biology*, VII(4230, LNCS), 2006.

[15] R. Milner. *Communication and Concurrency*. International Series in Computer Science. Prentice hall, 1989.

[16] R. Milner. *Communicating and mobile systems: the π-calculus*. Cambridge Universtity Press, 1999.

[17] B.O. Palsson. *Systems Biology. Properties of reconstructed networks.* Cambridge Universtity Press, 2006.

[18] G. Păun. *Membrane Computing. An Introduction*. Springer-Verlag, 2002.

[19] A. Phillips and L. Cardelli. A Correct Abstract Machine for the Stochastic Pi-calculus. In *BioConcur '04, Workshop on Concurrent Models in Molecular Biology*, 2004.

[20] C. Priami. Stochastic π-calculus. *The Computer Journal*, 38(6):578–589, 1995.

[21] C. Priami and P. Quaglia. Beta Binders for Biological Interactions. In V. Danos and V. Vincent Schächter, editors, *Computational Methods in Systems Biology, International Conference CMSB 2004, Paris, France, May 26-28, 2004, Revised Selected Papers*, volume 3082 of *LNCS*, pages 20–33. Springer, 2005.

[22] C. Priami, A. Regev, W. Silverman, and E. Shapiro. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Information Processing Letters*, 80(1):25–31, 2001.

[23] A. Regev, E.M. Panina, W. Silverman, L. Cardelli, and E. Shapiro. BioAmbients: An Abstraction for Biological Compartments. *TCS*, 325(1), 2004.

[24] A. Regev and E. Shapiro. Cells as computation. *Nature*, 419(6905):343, 2002.

[25] A. Romanel, L. Dematté, and C. Priami. The Beta Workbench. Technical Report TR-3-2007, The Microsoft Research - University of Trento Centre for Computational and Systems Biology, February 2007.

[26] J. Wing. Computational thinking. *Communications of the ACM*, 49(3):33–35, 2006.