Modeling and Simulating Biological Processes with Stochastic Multiset Rewriting

Matteo Cavaliere, Sean Sedwards

Microsoft Research – University of Trento
Centre for Computational and Systems Biology,
Trento, Italy
{matteo.cavaliere, sean.sedwards}@msr-unitn.unitn.it

Keywords. Modelling Biological Processes, Systems Biology, Membrane Systems

1 Extended Abstract

Membrane systems are models of computation inspired by the structure and the function of biological cells. The model was introduced in 1998 by Gh. Păun and since then many results have been obtained, mostly concerning the computational power of the model (for an updated bibliography the reader can consult the web-page [24]). More recently, membrane systems have been applied to systems biology and several models have been proposed for simulating biological processes (e.g., see the monograph dedicated to membrane systems applications, [9]).

In the original definition, membrane systems are composed of an hierarchical nesting of membranes that enclose regions in which floating objects exist. Each region can have associated rules for evolving these objects (called evolution rules, modelling the biochemical reactions present in cell regions), and/or rules for moving objects across membranes (called symport/antiport rules, modelling some kind of transport rules present in cells). Recently, inspired by *brane calculus*, [4], a model of a membrane system, having objects attached to the membranes, has been introduced in [5]. Other models bridging brane calculus and membrane systems have been proposed in [15,18]. A more general approach, considering both free floating objects and objects attached to the membranes has been proposed and investigated in [3]. The idea of these models is that membrane operations are moderated by the objects (proteins) attached to the membranes. However, in these models objects were associated to an atomic membrane which has no concept of inner or outer surface. In reality, many biological processes are driven and controlled by the presence, on the opportune side of a membrane, of certain specific proteins. For instance, receptor-mediated endocytosis, exocytosis and budding in eukaryotic cells are processes where the presence of proteins on the internal and external surface of a membrane is crucial (see e.g., [1]).

These processes are, for instance, used by eukaryotic cells to take up macromolecules and deliver them to digestive enzymes stored in lysosomes inside the cells. In general, all the compartments of a cell are in constant communication,

Dagstuhl Seminar Proceedings 06161
Simulation and Verification of Dynamic Systems
http://drops.dagstuhl.de/opus/volltexte/2006/706

with molecules being passed from a donor compartment to a target compartment by means of numerous membrane-enclosed transport packages, or *transport vesicles*. Once transported to the correct compartment the substances are then *processed* by means of local biochemical reactions (see e.g., [1]).

Motivated by this, we have introduced ([6]) a model combining some basic features found in biological cells: (*i*) *evolution* of objects (molecules) by means of multiset rewriting rules associated with specific regions of the systems (the rules model biochemical reactions); (*ii*) *transport* of objects across the regions of the system by means of rules associated with the membranes of the system and involving proteins attached to the membranes (in one or possibly both the two sides) and (*iii*) rules that take care of the *attachment/de-attachment* of objects to/from the sides of the membranes. Moreover, since we wanted to distinguish the functioning of different regions, we also associate to each membrane a unique identifier (a label).

In this paper we present a stochastic variant of the model (i.e., where each rule has an associated rate) that underlies an implemented simulator which we have used to model interesting biological cellular processes.

We wish to comment that the model implemented follows the philosophy of the evolution-communication model introduced in [7], where the system evolves by evolution of the objects and transport of objects by means of symport/antiport rules, that are essentially synchronized exchanges of objects. However, in our case the transport of objects may depend on the presence of particular *proteins* attached to the internal and external surfaces of the membranes. Therefore this paper can be seen as an implementation of a bridge between membrane systems and *projective brane calculus*, [10], where, in the framework of process algebra, directed actions associated to membranes have been considered.

Deterministic simulations are useful to describe reactions between large numbers of chemical objects, however they may not accurately represent the dynamical behaviour of small quantities of reactants. In this latter case a discrete stochastic simulation is more appropriate and, moreover, approximates the deterministic approach when the quantities are increased [12]. Hence we have created a simulator [25] which assumes discrete molecular interactions and uses the Gillespie algorithm [12] to stochastically choose at each step which single rule to apply (in one of the regions) and to calculate its stochastic time delay. Thus the more general free parallel theoretical model is here reduced to a specific sequential one.

To demonstrate the simulator we model the G-protein mating response in yeast *saccharomyces cerevisiae*, based on experimental rates provided by [14]. The G-protein transduction pathway involves membrane proteins and the transport of substances between regions and is a mechanism by which organisms detect and respond to environmental signals. It is extensively studied and many pharmaceutical agents are aimed at components of the G-protein cycle in humans. Figure 1 shows the relationships between the various reactants and regions modelled in the simulation, Figure 2 is the simulation script and Figure 3 shows the results of the simulation.

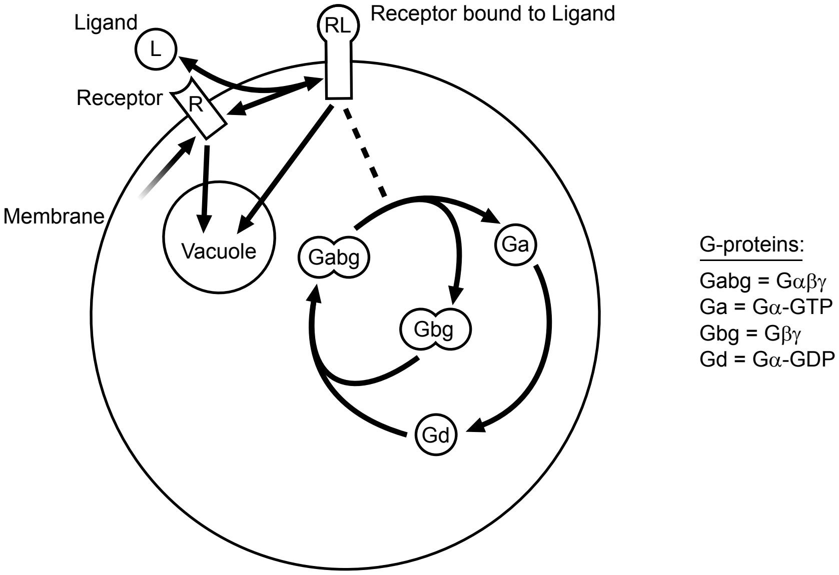

Fig. 1. Model of saccharomyces cerevisiae mating response.

A brief description of the process is that the yeast cell receives a pheromone signal (L) which binds to receptor R, integral to the cell membrane. The receptor-ligand dimer then catalyses the reaction that converts the inactive G-protein Gabg to the active Ga. A competing sequence of reactions converts Ga to Gabg via Gd in combination with Gbg. The bound and unbound receptor (RL and R, respectively) are degraded by transport into a vacuole via the cytoplasm.

2 Perspectives

We have presented a simulator that implements a stochastic variant of the model introduced in [6]. The simulator has an intuitive syntax and can be used to model biological processes where the transport of objects across membranes is coupled with the processing/decay of substances within the regions. As an example we have presented the simulation of saccharomyces cerevisiae heterotrimeric G-protein cycle.

Several different research directions may now be pursued. The model may be further developed, for example, to include evolution based on *maximal parallel* semantics, as commonly used in P systems. In that case it is most likely that many properties would not be decidable; an interesting problem is then to find (sub)classes (using restricted evolution and/or transport rules, say) where interesting properties are still decidable. Additionally, other bio-inspired operations may be introduced, such as *fission and fusion* of regions, all still dependent on

```
// Saccharomyces cerevisiae G-protein mating response
molecule L,R,RL,Gd,Gbg,Gabg,Ga
rule g_cycle {
   || 4-> |R|
   |R| + L 3.32e-18-> |RL|
   |RL| 0.011-> |R| + L
   |RL| 4.1e-3-> RL + ||
   |R| 4.1e-4-> R + ||
   Gabg + |RL| 1.0e-5-> Ga, Gbg + |RL|
   Gd + Gbg 1-> Gabg
   Ga 0.11-> Gd
}
rule vac_rule {
   || + R 4.1e-4-> R + ||
   || + RL 4.1e-3-> RL + ||
}
compartment vacuole [vac_rule]
compartment cell [vacuole, 3000 Gd, 3000 Gbg, 7000 Gabg, g_cycle :
                                                            ... |10000 R|]
system cell, 6.022e17 L
evolve 0-600000
plot cell[Gd,Gbg,Gabg,Ga:|R,RL|]
```

Fig. 2. Simulation script of G-protein cycle using data from [14].

the objects attached to the membranes, along the lines of the research found in [18].

Another direction of research is the application of the existing model. The implemented stochastic software can be used to simulate interesting biological processes where the rôle of surface proteins and transport of substances is crucial (as in drug-resistance, see e.g.,[17]).

References

1. B. Alberts, *Essential Cell Biology. An Introduction to the Molecular Biology of the Cell.* Garland Publ. Inc., New York, London, 1998.
2. N. Busi, R. Gorrieri, On the Computational Power of Brane Calculi. *Proceedings Third Workshop on Computational Methods in Systems Biology.* Edinburgh, 2005.
3. R. Brijder, M. Cavaliere, A. Riscos-Núñez, G. Rozenberg, D. Sburlan, Membrane Systems with Marked Membranes. Submitted.
4. L. Cardelli, Brane Calculi. Interactions of Biological Membranes. *Proceedings Computational Methods in System Biology 2004* (V. Danos, V. Schächter, eds.), Lecture Notes in Computer Science, 3082, Springer-Verlag, Berlin, 2005, pp. 257–278.
5. L. Cardelli, Gh. Păun, An Universality Result for a (Mem)Brane Calculus Based on Mate/Drip Operations. *Proceedings of the ESF Exploratory Workshop on Cellular Computing (Complexity Aspects)*, (M.A. Gutiérrez-Naranjo, Gh.

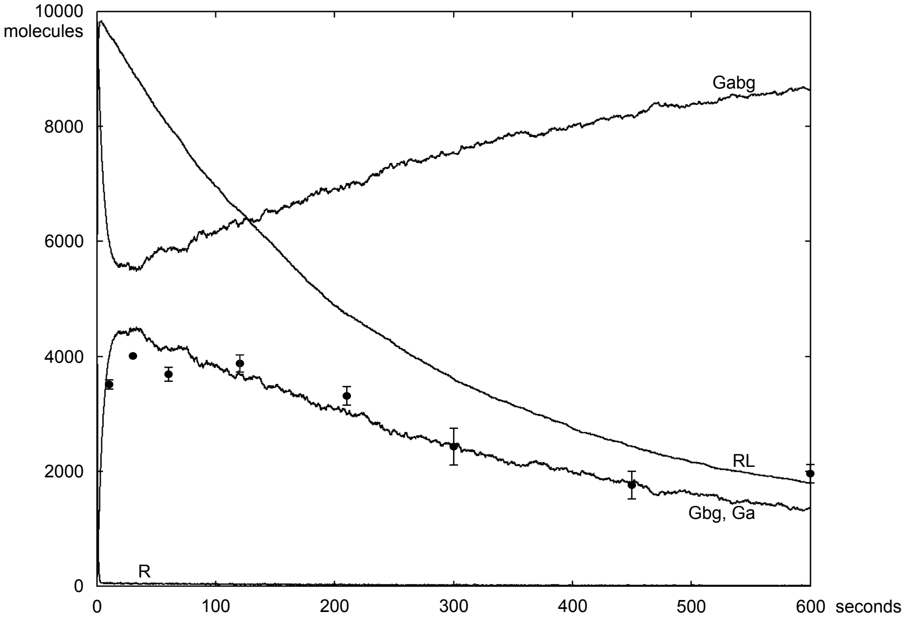

Fig. 3. Simulation results (continuous curves) and experimental data (points with error bars, [14]) corresponding to simulated Ga. Note that Gd decays rapidly and is not visible at this scale.

Păun, M.J. Pérez-Jiménez, eds.), Fénix Ed., Seville, Spain, pp. 75–94. Also at `http://www.gcn.us.es/`.

6. M. Cavaliere, S. Sedwards, Membrane Systems with Peripheral Proteins. Submitted.
7. M. Cavaliere, Evolution-Communication P Systems. *Proceedings International Workshop Membrane Computing,* (Gh. Păun, G. Rozenberg, A. Salomaa, C. Zandron eds.), Lecture Notes in Computer Science, 2597, Springer-Verlag, Berlin, 2003, pp. 134–145.
8. M. Cavaliere, D. Sburlan, Time-Independent P Systems. *Membrane Computing, 5th International Workshop, WMC2004* (G. Mauri, Gh. Păun, M.J. Pérez-Jiménez, G. Rozenberg, A. Salomaa, eds.), Lecture Notes in Computer Science, 3365, Springer-Verlag, Berlin, 2005, pp. 239–258.
9. G. Ciobanu, Gh. Păun, M.J. Pérez-Jiménez, eds., *Applications of Membrane Computing.* Springer-Verlag, Berlin, 2006.
10. V. Danos, S. Pradalier, Projective Brane Calculus. *Proceedings Computational Methods in System Biology 2004* (V. Danos, V. Schächter, eds.), Lecture Notes in Computer Science, 3082, Springer-Verlag, Berlin, 2005, pp. 134–148.
11. J. Dassow, Gh. Păun, *Regulated Rewriting in Formal Language Theory.* Springer-Verlag, Berlin, 1989.
12. D. T. Gillespie, Exact Stochastic Simulation of Coupled Chemical Reactions, *Journal of Physical Chemistry*, 81, 25, 1977.
13. J.E. Hopcroft, J.D. Ullman, *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 1979.

14. T.-M. Yi, H. Kitano, M. I. Simon, A quantitative characterization of the yeast heterotrimeric G protein cycle, *Proceedings of the National Academy of Science*, 100, 19, 2003.

15. S.N. Krishna, Universality Results for P Systems based on Brane Calculi Operations. *Theoretical Computer Science*, to appear.

16. H. Lodish, A. Berk, S.L. Zipursky, P. Matsudaira, D. Baltimore, J. Darnell, *Molecular Cell Biology*, Freeman, Fifth Edition.

17. K. H. Lundstrom, *G Protein Coupled Receptors in Drug Discovery*, Taylor & Francis, 2005.

18. A. Păun, B. Popa, P Systems with Proteins on Membranes and Membrane Division. *Proceedings Tenth International Conference in Developments in Language Theory, DLT06*, Lecture Notes in Computer Science, Springer-Verlag, to appear.

19. Gh. Păun, Computing with Membranes. *Journal of Computer and System Sciences*, 61, 1 (2000), pp. 108–143. First circulated as TUCS Research Report No 28, 1998.

20. Gh. Păun, *Membrane Computing – An Introduction*. Springer-Verlag, Berlin, 2002.

21. Gh. Păun, G. Rozenberg, A Guide to Membrane Computing. *Theoretical Computer Science*, 287-1 (2002), pp. 73–100.

22. G. Rozenberg, A. Salomaa, eds., *Handbook of Formal Languages*. Springer-Verlag, Berlin, 1997.

23. A. Salomaa, *Formal Languages*. Academic Press, New York, 1973.

24. http://psystems.disco.unimib.it

25. http://www.msr-unitn.unitn.it/downloads

A Appendix - The Simulator Syntax

An example of the basic syntax of the simulator is shown in the following script:

```
// Lotka autocatalytic reactions [Journal Am Ch Soc, 1920]

molecule X,Y1,Y2,Z

rule r1 X + Y1 0.0002-> 2Y1 + X
rule r2 Y1 + Y2 0.01-> 2Y2
rule r3 Y2 10-> Z

system 100000 X, 1000 Y1, 1000 Y2, r1,r2,r3

evolve 0-50000000

plot Y1,Y2
```

The reacting species are first listed in the type definition beginning with the keyword `molecule`.

The behaviour of the reactants is then defined using rule definitions comprising the keyword `rule` followed by a rule identifier and the rewriting rule itself. Note that rules are user-defined types which may be instantiated more than once. The value preceding the implication symbol (->) is the average reaction rate.

It is also possible to define a rule as a group of rules. E.g.,

```
rule lotka {
      r1 X + Y1 0.0002-> 2Y1 + X
      rule r2 Y1 + Y2 0.01-> 2Y2
      rule r3 Y2 10-> Z
}
```

uses the single identifier `lotka` to define the behaviour described by `r1`, `r2` and `r3` in the previous example. Such groups are convenient to describe a subsystem of behaviour.

The system is instantiated using the `system` keyword followed by a list of constituents, in the above case comprising numbers of molecules and rules.

The number of reactions to simulate is specified using the `evolve` keyword followed by the range of data points to record. The simulation will always proceed from zero to the maximum value, however data will only be recorded from the minimum given.

The species to be observed are defined using the `plot` keyword followed by a list of reactants.

Enclosed regions, subsequently referred to as *compartments*, may be defined using the keyword `compartment` followed by an identifier and a list of contents and rules, all enclosed by square brackets. For example,

```
compartment c1 [100 X, 100 Y1, r1, r2]
```

instantiates a compartment having the label `c1` containing 100 `X`, 100 `Y1` and rules `r1` and `r2`. Compartments may contain other compartments, so the following is possible given the previous definition:

```
compartment c2 [100 Y2, c1]
```

Compartments contain a notional membrane which surrounds them and to which may be attached reactants. The compartment syntax is thus extended using the symbol `||` to represent the membrane. Hence,

```
compartment c3 [100 X, c2 : 10 Y2||10 Y1]
```

has the meaning that the compartment `c3` contains 100 `X`, compartment `c2` and the membrane surrounding `c3` has 10 `Y2` attached to its inner surface and 10 `Y1` attached to its outer surface. Note that the list of floating contents and rules appears on the left of the definition and is separated from the membrane contents by a `:`. The rule syntax is correspondingly extended, so

```
rule r4 X + Y2|| 0.1-> Y2|| + X
```

means that if one `X` exists within the compartment and one `Y2` exists attached to the inner surface of the membrane, then the `X` will be transported outside the compartment and the state of the membrane will be unaffected. Hence the `+` is non-commutative: the left side represents the internal part of the compartment and the right hand side is external. Similarly, the left side of the `||` symbol represents the internal surface and the right hand side represents the external surface of the membrane. Reactants integral to the membrane (not specifically mentioned in the previous text) may be defined by listing them between the vertical bars, so

```
rule r5 X + |Y2| 0.1-> |Y2| + X
```

represents exocytosis where `Y2` must exist integral to the membrane for the reaction to proceed.

To plot the contents of a specific compartment the `plot` statement uses syntax similar to that used in the compartment definition. E.g.,

```
plot X, c3[X,Y1 : Y1|Y2|]
```

records the number of free-floating `X` in the system environment and also the contents of compartment `c3`. Specifically, it records the number of free-floating `X` and `Y1` in `c3` as well as the number of `Y1` attached to the inner surface and the number of `Y2` integral to the membrane.

The simulator will be available for free download at
`http://www.msr-unitn.unitn.it/downloads`