Journal of Computer Science 7 (11): 1667-1673, 2011
ISSN 1549-3636
© 2011 Science Publications

Membrane Computing as a Modeling Tool for Discrete Systems

[1]Ravie Chandren Muniyandi and [2]Abdullah Mohd. Zin
[1]School of Computer Science,
[2]School of Information Technology,
Faculty of Information Science and Technology,
University Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia

Abstract: Problem statement: Discrete systems have been modeled by using Ordinary Differential Equation (ODE) in which the variation of concentration of an object was modeled as continuous and deterministic manner, contrary to the real behaviors of such systems. Although, this approaches able to generate the general behavior of the system, the specific discrete processes and stochastic behaviors in the system have not been addressed. Membrane computing has been an unconventional computational approach that provides a platform for modeling discrete systems. It deals with parallel, distributed and non-deterministic computing models. **Approach:** This study was carried to compare the ODE with membrane computing approach in modeling a discrete system by taking Prey-Predator population as the case study. Membrane computing simulator based on Gillespie Algorithm and Probabilistic and Symbolic Model Checker (PRISM) were used to verify and validate the model. **Results:** Membrane computing able to not only maintain the dynamics and equilibrium of Prey-Predator population but also preserve the discrete and stochastic evolvement of the prey and predator in the population by sustaining the properties of the system. **Conclusion:** Membrane computing modeling approach preserved the characteristics of discrete systems that absent in the ODE approach.

Key words: Membrane computing, prey predator population, discrete systems, modeling approach, modeling discrete, gillespie algorithm, deterministic manner, differential equation, computing simulation

INTRODUCTION

Membrane computing (Paun, 1998) is an area of computer science that abstract computing ideas and models from the structure and the functioning of living cells. This mechanism provides a platform for modeling discrete systems in which a membrane delimits a compartment from its external environment and provides local environment that regulates specific processes.

The processes evolve in parallel and non-deterministic way in which all evolution rules are simultaneously applied to all the objects. The computation halts to produce output when no rule is applied. The discrete characteristics of membrane computing allow the dynamic systems evolve in discrete steps according to the processes.

However, some of the discrete systems have been represented in Ordinary Differential Equation (ODE) (Blanchard *et al*., 2006) which has continuous and deterministic evolution strategy. This approach has shown limitations when the variation of concentration of an object is modeled as continuous and deterministic manner, which ignores the behaviors of discrete systems itself (Jong, 2002). Membrane computing has been identified as an alternative to address these limitations.

Prey-Predator population (Jones *et al*., 2003) is a discrete system that has been modeled in ODE. The same model can be represented in membrane computing by using rewriting rules. The model is simulated with membrane computing simulation strategy based on Gillespie algorithms (Gillespie, 2001; Muniyandi and Abdullah, 2010). The properties of Prey Predator population are verified with Probabilistic Model Checker (PRISM) (Kwiatkowska *et al*., 2002; Muniyandi *et al*., 2010). The results are compared with ODE approach to certify the capability of membrane computing in modeling discrete systems.

Prey-predator population: The Prey- Predator model is a biological system that describes the dynamics of Prey-Predator population. In this model, the food supply of prey species is assumed to be abundant and no threat to its growth. Meanwhile, the only food

Corresponding Author: Ravie Chandren Muniyandi, School of Computer Science, Faculty of Information Science and Technology, University Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia

supply of predator species is the prey to determine its growth. The interaction between prey and predator is to maintain the equilibrium of Prey- Predator population over time. The prey species could grow exponentially with its unlimited food supply but the predator species act to counterbalance the prey growth rate. Therefore, two assumptions formulated for this model to maintain the equilibrium of the population. First, the size of the prey and predator population is related to the rate at which predator encountering prey. Second, the predator has to lead to natural death which is related to a rate of fixed proportion.

Based on the assumptions, the rules in the prey predator model are interpreted. First the rule over the prey in which the change in the number of prey is specified by its own growth minus the rate at which it is preyed upon. Second, the rule over predator signifies the growth of the predator population in which its growth is not necessarily equal to the rate at which it consumes the prey but there is another rule of exponential decay to represent the natural death of the predator.

The ODE model of prey-predator population: The ODE of Prey-Predator model is represented by a pair of first order, non-linear, differential equations (Jones *et al*., 2003). The interactions between prey and predator determine the number of prey and predator at certain time step in the system. Prey-Predator population is modeled in ODE as Eq. 1 and 2:

$$\frac{dX}{dt} = k_1X - k_2XY \qquad (1)$$

$$\frac{dY}{dt} = k_4XY - k_3Y \qquad (2)$$

Y represents the number of predator and X represents the number of prey and k_1, k_2, k_3 and k_4 are the kinetic constants.

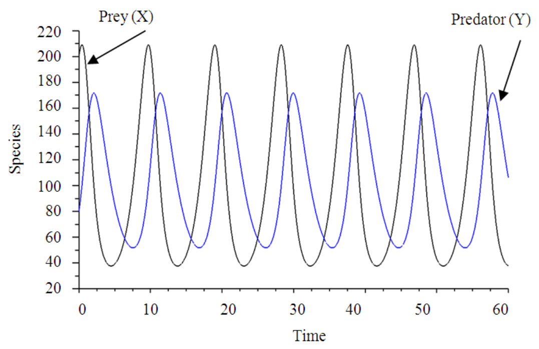

Fig. 1: Oscillation of the prey-predator model simulated by ODE

The ODE model of Prey-Predator population in Fig. 1 is simulated by ODE simulation approach (Jones *et al*., 2003). The simulation in Fig. 1 shows the oscillations of the prey-predator model simulated by ODE for 60 time steps for x = 200 and y = 80 with kinetic constants k1 = 1, k2 = 0.01, k3 = 0.05 and k4 = 0.005. The result simulated by ODE as shown in Fig. 1 has captured the oscillation behavior of prey-predator model. It has a deterministic pattern of oscillation at each cycle of the oscillation with fixed peak and dip at each cycle.

MATERIALS AND METHODS

Reactions and parameters: The case study of Prey-Predator population is taken from Jones *et al*. (2003) research paper it is modeled by using ODE approach. The number of initial species, reactions of the rules and parameters involved in the system are extracted from the ODE of the Prey-Predator model.

Kinetic constants and initial multisets: The selection of initial multisets and the kinetic constants are done to determine the computation in membrane computing of Prey-Predator model. This basically is an attempt to gain the behaviour of Prey-Predator population through membrane computing model and compared it to the results achieved through the ODE model of Prey-Predator population. Subsequently the membrane computing model is analysed whether it could preserve the stochastic characteristics of the Prey-Predator population. The value of initial multisets and kinetic constants for membrane computing simulation strategies are determined through black box testing (Beizer, 1995) in which the inputs are selected based on the expected output of the system. For instance, the initial multisets and kinetic constants extracted from the ODE model are taken as initial test cases with Gillespie Simulator. Then these test cases are adjusted accordingly to determine the appropriate amount of initial multisets and kinetic constants needed to preserve the stochastic behaviour of membrane computing model. The best kinetic constants are chosen when the oscillation of Prey-Predator population is obtained. The chosen kinetic constants for are 10, 0.02 and 15, respectively, when the initial numbers of preys and predators are fixed at 1000 and 200, respectively.

Modeling: The objects, reactions and parameters extracted from ODE model of Prey-Predator population are utilized in membrane computing modeling Prey-Predator population by using membrane computing formalism outlined by Muniyandi and Abdullah (2009).

Simulation: Gillespie Simulator is used to simulate the membrane computing model of Prey-Predator population. Firstly, the membrane computing model is converted into the notation of system biology markup language which describes the components of the biological system. Then, the simulator will specify the list of compartments and the structural hierarchy and the initial amounts of the objects from the SBML notations. This is the initial state of the system and is given to the simulator to produce an evolution of the objects over simulation steps. The membrane computing simulation results using Gillespie algorithm are compared to the results of ODE approach.

Model checking: PRISM (Kwiatkowska *et al*., 2002) is used to model check membrane computing model of Prey-Predator population by specifying the properties of the system. PRISM is a probabilistic model checker that represents a technique to formally verify quantitative properties of a stochastic system. By using the concept of rewards, PRISM is used to specify and to analyze properties of Prey-Predator. The rewards are analyzed with the R{" M "}=? [I=T], where M is the name of the rewards, I is the instantaneous rewards and T is the time steps. Given a membrane computing model of Prey-Predator population, the model is tested automatically to ascertain whether it meets the specification of the system. There are four steps in the model checking process. First, the properties for the Prey-Predator population are obtained from the behavior of this system. Second, the membrane computing model is translated into PRISM formalism. This translation technique from membrane computing into PRISM applied in this research is proposed by Romero-Campero *et al*. (2006). Third, the model in PRISM is simulated and model checked with the properties of Prey-Predator population. PRISM is used to specify and to analyze properties based on rewards. Finally, the data of the results generated by PRISM is presented into a graph and analyzed to verify whether the specified properties are preserved or not.

RESULTS

Membrane computing model of prey-predator population: Based on the characteristics of the two differential Eq. 1 and 2 in the ODE model of Prey-Predator population, the system is converted into discrete systems. There are two equations, two species and three kinetic constants in Prey-Predator population where, Y is the number of predator; X is the number of prey; dY/dt and dX/dt represents the growth of the two populations against time t; and k_1, k_2, k_3 and k_4 are parameters representing the interaction of the two species. The reactions between the species occur within a compartment.

The growth of Prey: In differential Eq. 1, with rate k_1, amount of X is increased and this process will contribute to the growth of X in the system. This process can be concluded in rewriting rule as follow Eq. 3:

$$X \xrightarrow{k_1} X + X \quad (3)$$

In this reaction, one amount of X is replaced with two amount of X with the rate of, at time step t.

The reduction of Prey: In differential Eq. 1 this process is activated by rate of. The interaction between X and Y with rate in the Eq. 1 will decrease the amount of X in the system. This process can be represented in rewriting rule as Eq. 4:

$$X + Y \xrightarrow{k_2} Y \quad (4)$$

In the reaction, the interaction between one amount of X and one amount of Y depletes one amount of X but Y unchanged, with the rate of at time step t.

The growth of Predator: In differential Eq. 2 this process is activated by rate of. The interaction between X and Y in differential Eq. 2, will increase the amount of Y with the rate of. This process can be represented in rewriting rule as Eq. 5:

$$X + Y \xrightarrow{k_4} X + Y + Y \quad (5)$$

In the reaction, interaction between one amount of X and one amount of Y with the rate of, X is remain stagnant and one amount of Y is generated at time step t.

The decay of Predator: In differential Eq. 2, the amount of Y will decrease with rate of. This process can be represented in rewriting rules as Eq. 6:

$$Y \xrightarrow{k_3} \quad (6)$$

In this reaction one amount of Y is depleted with rate at time step t.

The reactions (4) and (5) are performing similar interactions. Therefore, there are also examples in the Prey-Predator model, in which the weight of are similar to the weight of (Jones *et al*., 2003). In this investigation the weight of are taken as similar to the weight of for generalization of process in the model. In this case the reaction (4) and (5) are merged as Eq. 7:

$$X + Y \xrightarrow{k_2} Y + Y \quad (7)$$

Based on the discrete system of the Prey-Predator Population above, a membrane computing model is built. A model for Prey-Predator population (PP) is obtained by considering a membrane computing with a compartment contains rules describing the reactions between preys and predators. The model is represented as:

$$PP = (V, \mu, \omega, R)$$

The objects are prey and predator represented as X and Y respectively. They are:

$V=\{X, Y\}$

The initial multisets are:

$\omega = \{nX, mY\}$

where, n and m are integer multiplicities.

Since the system has single compartment, it only able to perform transformation of objects. Therefore, the transformation rule has the form: where are multisets in a compartment. K is a real number representing the kinetic constant, which represent the rate of reaction between objects. A rule of this form is interpreted as follows: based on the rate of k, the multiset u is transformed into multiset v inside a compartment. Based on the rewriting rules (3), (6) and (7) described above, the Prey-Predator population dynamics is described in membrane computing as follow Eq. 8-10:

$$R1:[X] \xrightarrow{k_1} [X,X] \quad (8)$$

$$R2:[X,Y] \xrightarrow{k_2} [Y,Y] \quad (9)$$

$$R3:[Y] \xrightarrow{k_3} [] \quad (10)$$

R1, R2 and R3 are prey reproduction, predator reproduction and predator death rules, respectively.

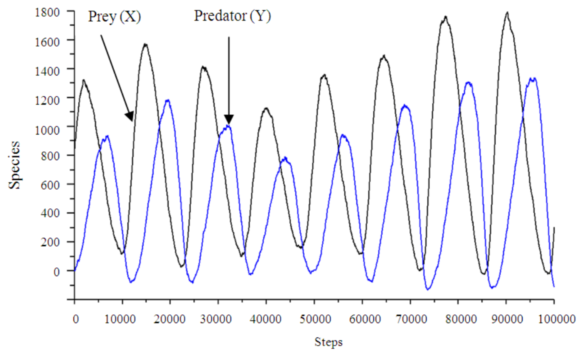

Fig. 2: Oscillation of the Prey-Predator model simulated by Gillespie algorithm with k_1=10, k_2=0.02, $k_{3=15}$

Simulation of membrane computing model: The simulation in Fig. 2 shows the Prey-Predator model of membrane computing simulated with Gillespie simulator. The simulation shows that the pattern of oscillation at each cycle in the oscillation is not fixed as in the ODE. The peak and dip of the simulation are different at each cycle of the oscillations but has maintained the general pattern of the oscillation. The stochastic behavior of the Prey-Predator Population is preserved in this non-deterministic and discrete model of membrane computing.

Model checking of membrane computing model: The properties for the Prey-Predator population are obtained from the behavior of this system elaborated by Jones *et al*. (2003). Recurrence behavior of the system and the existence of equilibrium probability distribution maintain the stability of the system in the form of oscillations. To facilitate this behavior, the Prey-Predator system should preserve the following properties: (A1) The rules are selected stochastically based on the number of prey and predator at each time steps and the value of reaction constants to maintain the equilibrium of the system; (A2) The number of prey and predator must not equal to 0 at any time steps; (A3) The number of prey and predator become equal or intersect each other twice at each cycle of the system; (A4) Percentage of increase or decrease of number of prey is higher most of the time steps than the number of predator; (A5) Percentage of change between prey and predator is higher most of the time steps for prey than predator.

Property (A1): The Fig. 3 shows the selection of rules at different periods of time steps based on the rewards. This graph shows that at each time step one of the three rules is selected stochastically. The patterns of selections differ for each period of time steps as shown by the graph.

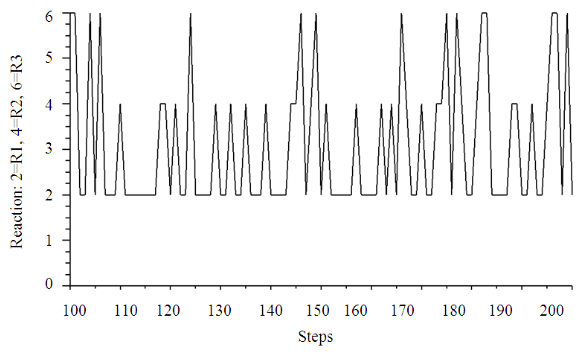

Fig. 3: Stochastic behavior of Prey-Predator model

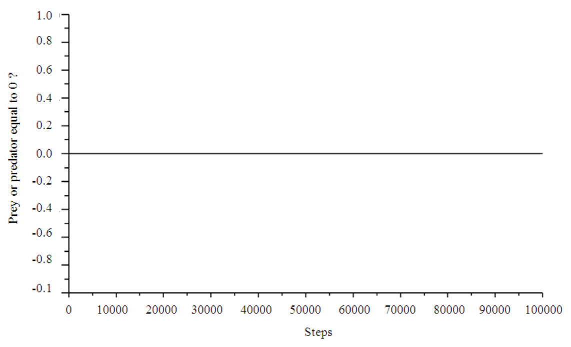

Fig. 4: The number of prey and predator must not equal to 0 at any time steps

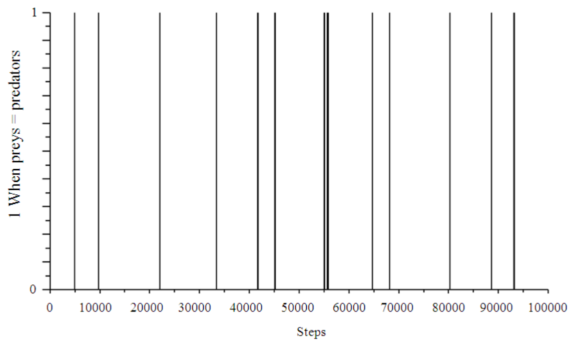

Fig. 5: The number of prey and predator become equal twice at each cycle of the system

This means that the stochastic behavior of the system is maintained to make sure the stability and consistency of the system at each cycle of the oscillation.

Property (A2): Figure 4 shows that not at once along the simulation steps of the system, the number of prey or predator have been equal to 0 based on the rewards. This result demonstrates that the behavior of the system is always consistent to make sure that the number of prey and predator must always greater than 0 to stabilize the system.

Property (A3): Figure 5 demonstrates the time steps when number of prey is equal to number of predator generated by the rewards. The intersection between prey and predator occurs twice at each cycle in the oscillations when the prey and predator either keep decreasing or increasing. However the graph shows that the period of occurrence of one intersection to another is not similar. This is mainly due to the stochastic behavior of the system. However the pattern of intersection in each of the cycle in the oscillation is preserved to maintain the stability of the system.

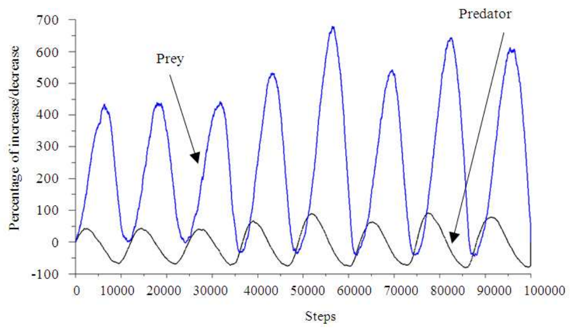

Fig. 6: Percentage of increase/decrease of Preys an Predators

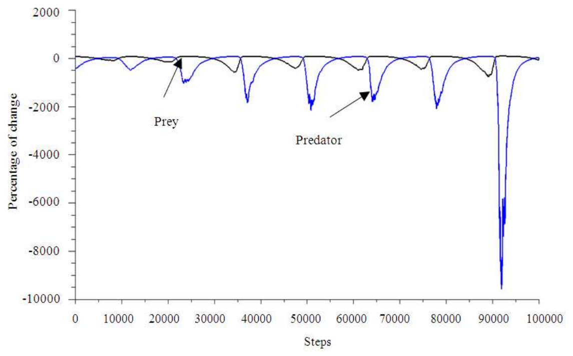

Fig. 7: Percentage of changes between prey and predator

Property (A4): Figure 6 illustrates the percentage of increase and decrease of prey and predator based on the rewards. The graph shows that the percentage of increase and decrease of predator is higher than the percentage of increase and decrease of prey. At the initial state, the number of prey is five times higher than the number of predator. The sharp increase of predator is to certain extent decrease the population of prey. Meanwhile, the sharp decrease of predator population gives some space for prey to increase its population to attain the initial level over again. The percentage of increase/decrease of prey and predator is almost similar at each of the cycle of the oscillation. This demonstrates that the equilibrium of prey-predator population has been preserved by maintaining the percentage of increase/decrease of prey and predator accordingly at each time step.

Property (A5): Figure 7 shows the percentage of changes of prey and predator compared to the opposite population as outlined in the rewards. This graph shows that at most of the time number of prey exceeds number of predator. But at certain period when number of predator is above the number of prey, prey is decreasing to control the increase of predator. When number of predator is decreasing, the number of prey is increasing. As shown in the Fig. 6, this result also shows that the stability of the system is preserved by controlling the increase and decrease of the prey and predator accordingly at each time step.

DISCUSSION

The results demonstrate that, the Prey-Predator model described in membrane computing could simulate the behavior of the system as shown in Fig. 2. This shows that membrane computing able to confine the dynamics of Prey-Predator population. ODE of Prey-Predator is used to model kinetics of the reactions of two species. It would continuously vary the concentration of species in deterministic dynamics as shown in Fig. 1. In contrast, membrane computing takes into consideration the discrete character of the quantity of species in Prey-Predator system by using rewriting rules. This demonstrates that membrane computing emulates discrete behavior of Prey-Predator system but ODE has ignored the discrete behavior by representing the system in continuous way.

Nevertheless, there are differences in performance between these approaches. The ODE simulation with the ratio of initial objects is two predators to five preys, takes around 10 units time to complete a cycle in the oscillation as shown in Fig. 1. Meanwhile, the membrane computing simulation with a ratio of initial objects is one predator to five preys takes around 15000 time steps in Fig. 2 to complete a similar cycle.

This shows that in the ODE approach, the large ratio in the mix of objects in the reaction and the deterministic feature make the reactions fast. In the membrane computing simulation, more time is needed to measure the weight of each reaction based on the ratio in the mix of objects and to subsequently choose the appropriate reaction at each time step. The inherent randomness in Prey-Predator system is captured by using stochastic simulation strategy of membrane computing.

The stochastic behavior of Pre-Predator population is maintained by making sure the rules are selected non-deterministically in membrane computing at each time step as in Fig. 3. While maintaining the stochastic behavior of the system, Fig. 4 demonstrates that membrane computing also ensure the interaction between prey and predator always stable to prevent any circumstances that could eliminate one of them from the system. This means that the equilibrium of the system represented in membrane computing is being preserved though there are situation where there could be more predator than prey or the number of prey is equal to number of predator as in Fig. 5. However, as shown by Fig. 6 and Fig. 7, membrane computing ensures that the percentage of increase of prey is higher most of the time compared to the percentage of increase of predator to maintain the equilibrium of the system. The model checking results demonstrate that the properties of Prey-Predator population have been preserved by maintaining the discrete and stochastic behavior of the system.

CONCLUSION

The discrete character of membrane computing model is not only preserving the dynamics of the prey-predator population in oscillations, but also making sure the discrete and stochastic behaviors of the system are conserved. This study underlines that the non-determinism and discrete characteristics of membrane computing capable in preserving the properties of the discrete systems better than the modeling approach of ODE. This means that membrane computing model could not only be used to analyze general behavior but also could be used to investigate specific behavior of biological system such as Prey-Predator population.

ACKNOWLEDGEMENT

This study supported by Exploratory Research Grant Scheme (ERGS), Ministry of Higher Education (Malaysia). Grant code: ERGS/1/2011/STG/UKM/03/7.

REFERENCES

Beizer, B., 1995. Black-Box Testing: Techniques for Functional Testing of Software and Systems. 1st Edn. Wiley, London, ISBN: 0471120944, pp: 294.

Blanchard, P., R.L. Devaney and G.R. Hall, 2006. Differential Equations. 3rd Edn. Cengage Learning, US, ISBN: 0495012653, pp: 823.

Gillespie, D.T., 2001. Approximate accelerated stochastic simulation of chemically reacting systems. J. Chem. Phys., 115: 1716-1733. DOI: 10.1063/1.1378322

Jones, D.S., M.J. Plank and B.D. Sleeman, 2003. Differential Equations and Mathematical Biology. 2nd Edn. Chapman and Hall/CRC, London, ISBN: 1420083570, pp: 444.

Jong, H.D., 2002. Modeling and simulation of genetic regulatory systems: A literature review. J. Comp. Biol., 9: 67-103. DOI: 10.1089/10665270252833208

Kwiatkowska, M., G. Norman and D. Parker, 2002. PRISM: Probabilistic symbolic model checker. Lectu. Notes Compu. Sci., 2324: 113-140. DOI: 10.1007/3-540-46029-2_13

Muniyandi, R.C. and M.Z. Abdullah, 2009. Modeling of biological processes by using membrane computing formalism. Am. J. Applied Sci., 6: 1961-1969. DOI: 10.3844/ajassp.2009.1960.1968

Muniyandi, R.C. and M.Z. Abdullah, 2010. Experimenting the simulation strategy of membrane computing with gillespie algorithm by using two biological case studies. J. Compu. Sci., 6: 525-535. DOI: 10.3844/jcssp.2010.525.535

Muniyandi, R.C., M.Z. Abdullah and Z. Shukor, 2010. Model checking the biological model of membrane computing with probabilistic symbolic model checker by using two biological systems. J. Compu. Sci., 6: 666-676. DOI: 10.3844/jcssp.2010.669.678

Paun, G., 1998. Computing with membranes. J. Compu. Syst. Sci., 61: 108-143. DOI: 10.1006/jcss.1999.1693

Romero-Campero, F.S., M. Gheorghe, L. Bianco, D. Pescini and M.J. Perez-Jimenez *et al*., 2006. Towards probabilistic model checking on p systems using PRISM. Lecture Notes Compu. Sci., 4361: 477-495. DOI: 10.1007/11963516_30