Computer Science Journal of Moldova, vol.**23**, no.2(68), 2015

BioMaxP : A Formal Approach for Cellular Ion Pumps

Bogdan Aman Gabriel Ciobanu

Abstract

We look at the living cells as complex systems of ion pumps working in parallel to ensure proper physiologic functionalities. To model such a system of pumps, we define a simple and elegant approach that allows working with multisets of ions, explicit interpretation of the transportation (from inside to outside, and from outside to inside) based on the number of existing ions, and a maximal parallel execution of the involved pumps.

1 Introduction

All living cells can be seen as complex systems of interacting components, having different concentrations of ions (e.g., Na^+, K^+ and Ca^{++}) across the cell membrane. Under resting conditions, Na^+ and Ca^{++} ions enter the cells and K^+ ions exit the cell because the concentration of K^+ is high inside the cell and low outside, while the opposite situation is found for Na^+ and Ca^{++}. A fundamental mechanism in most of the living cells is the Na^+/K^+-$ATPase$ that is essential for the maintenance of Na^+ and K^+ concentrations across the membrane by transporting Na^+ out of the cell and K^+ back into the cell. This pump is the first discovered ion transporter; for this discovery, the Danish chemist Jens Skou received the Nobel Prize in 1997.

In this paper we model the movement of ions and the conformational transformations of ion transporters (NaK ion pumps, Na and K ion channels) by using BioMaxP , a very simple but powerful approach. We use an operational semantics able to capture quantitative

aspects (e.g., number of ions) and abstract conditions associated with evolution (e.g., the number of ions is between certain thresholds). The modelling aims to facilitate a better understanding of the living cell viewed as a complex system of parallel ion transporters.

The novelty of our approach is that we model systems composed of more than one pump as usually done (e.g., see [5]). Since the pumps non-deterministically choose which ions to transport, the complexity of such systems increases with the number of pumps. For further notions about NaK pump, the interested reader can consult [1].

2 Syntax and Semantics of BioMaxP

The prototyping language BioMaxP provides sufficient expressiveness to model in an elegant way the interaction in complex systems of parallel ion pumps. The cell is a complex system of parallel pumps trying to keep the equilibrium of ions inside the cell. In order to model these pumps, we enforce that their functioning takes place only if the number of various types of ions is between some accepted limits given by min and max values. Therefore the syntax and semantics emphasize the process of counting them, and the way the quantities of ions vary during evolution. The semantics of BioMaxP is provided by multiset labelled transitions in which multisets of actions are executed in parallel.

Syntax of BioMaxP The syntax of BioMaxP is given in Table 1, where the following are assumed:

- a set *Chan* of ion transportation channels a, and a set Id of process identifiers (each $id \in Id$ has its arity m_{id});
- for each $id \in Id$ there is a unique process $id(u_1, \ldots, u_{m_{id}} : T_1, \ldots, T_{m_{id}}) \stackrel{def}{=} P_{id}$, where the distinct variables u_i are parameters, and the T_i are ions types;
- v is a tuple of expressions built from values, variables and allowed operations;
- T represent ions types.

Table 1. BioMaxP Syntax

Processes			
P, Q	$::=$	$a^{min}!(v : T)$ then P ı	(sending)
		$a^{max}?(f(u : T))$ then P ı	(receiving)
		$id(v)$ ı	(recursion)
		$P \mid Q$	(parallel)

A constraint ^{min} associated with a sending action $a^{min}!(z : T)$ then P makes the channel a available for sending z units/ions of type T only if the total available quantity of ions of type t is greater than min. A constraint ^{max} associated to a receiving action $a^{max}?(x : T)$ then P along a channel a is activated only if the number of ions of the type T available is less than max. The function f of the receiving action can be either id (we often omit it), meaning that the received ions are to be transported, or add, meaning that the ions are received from some other process. The only variable binding constructor is $a^{max}?(u : T)$ then P; it binds the variable u within P. The free variables of a process P are denoted by $fv(P)$; for a process definition, is assumed that $fv(P_{id}) \subseteq \{u_1, \ldots, u_{m_{id}}\}$, where u_i are the process parameters. Processes are defined up-to an alpha-conversion, and $\{v/u\}P$ denotes P in which all free occurrences of the variable u are replaced by v, eventually after alpha-converting P in order to avoid clashes. Processes are further constructed from the parallel composition $P \mid Q$. A system of parallel pumps is represented as a process with some initial values for the numbers of ions.

Remark 1 *In order to focus on the local interaction aspects of* BioMaxP *, we abstract from arithmetical operations, considering by default that the simple ones (comparing, addition, subtraction) are included in the language.*

Operational Semantics of BioMaxP The operational semantics rules of BioMaxP is presented in Table 2. The multiset labelled transi-

tions of form $P \xrightarrow{\Lambda} P'$ use a multiset Λ to indicate the actions executed in parallel in one step. When the multiset Λ contains only one action λ, in order to simplify the notation, $P \xrightarrow{\{\lambda\}} P'$ is simply written as $P \xrightarrow{\lambda} P'$. We assume that in order to interact the processes can commute, namely $P \mid Q$ is the same process as $Q \mid P$.

Table 2. BioMaxP Operational Semantics

$$\text{(Com)}\quad \frac{v : T \quad and \quad min \leq |T| \leq max}{a^{min}!\langle v\rangle \text{ then } P \mid a^{max}?(f(u:T)) \text{ then } P' \xrightarrow{\{v/u\}} P \mid \{v/u\}P' \text{ and } |T| = |T| - v \text{ if } f = id \text{ or } |T| = |T| + v \text{ if } f = add}$$

$$\text{(Call)}\quad \frac{\{v/u\}P_{id} \xrightarrow{id} P'_{id}}{id(v) \xrightarrow{id} P'_{id}} \text{ where } id(v:T) \stackrel{def}{=} P_{id}$$

$$\text{(Par1)}\quad \frac{P_1 \xrightarrow{\Lambda_1} P_1' \qquad P \not\rightarrow}{P_1 \mid P \xrightarrow{\Lambda_1} P_1' \mid P} \qquad \text{(Par2)}\quad \frac{P_1 \xrightarrow{\Lambda_1} P_1' \quad P_2 \xrightarrow{\Lambda_2} P_2'}{P_1 \mid P_2 \xrightarrow{\Lambda_1 \cup \Lambda_2} P_1' \mid P_2'}$$

In rule (Com), an output process $a^{min}!\langle v\rangle$ then P succeeds in sending a tuple of values v over channel a to process $a^{max}?(u:T)$ then P if v has the same type T as u and if the number of ions of type T is between min and max, namely $v : T$ and $min \leq |T| \leq max$. Both processes continue to execute, the first one as P and the second one as $\{v/u\}P'$. Once the ions are send away, $f = id$, the number of ions of type T becomes $T - |v|$, while if they are received, $f = add$, then the number of ions of type T becomes $T + |v|$. Rule (Call) describes the evolution of a recursion process. Rules (Par1) and (Par2) are used to compose larger processes from smaller ones by putting them in parallel, and considering the union of multisets of actions. In rule (Par2), $P \not\rightarrow$ denotes a process P that cannot evolve. It can be noticed that in rule (Par2) we use negative premises: an activity is performed based on the absence of actions. This is due to the fact that sequencing

the evolution can only be defined using negative premises, as done for sequencing processes [6, 10].

Example 1 *The use of* BioMaxP *for specifying complex systems of pumps is illustrated by describing in an explicit way the molecular interactions and conformational transformations of a large system of ion transporters, namely Na^+K^+ ATPases and Na and K ion channels, that are concerned with the movement of sodium-potassium ions in and out of a cell whenever certain thresholds are verified. The system we consider is formed from n_1 NaK pumps, n_2 Na channels and n_3 K channels. Each pump i is modelled by three processes: one that models the interaction of the pump with the environment, one modelling the interaction with the cell and another one that models the transport of ions through the membrane. The molecular components are processes modelled as the ends of a channel (one end for input, and another for output), while the molecular interaction coincides with communication on channels.*

The initial system of pumps is described in BioMaxP *by:*

$$Cell(NaEnv, KEnv, NaCell, KCell, AtP, ADP, P) =$$
$$| \; NaKPumpEnv(0) \mid NaKPumpCell(0) \mid NaKPump(0)$$
$$\cdots \mid NaKPumpEnv(n_1\text{-}1) \mid NaKPumpCell(n_1\text{-}1) \mid NaKPump(n_1\text{-}1)$$
$$\mid NaPumpEnv(n_1) \mid NaPumpCell(n_1) \mid NaPump(n_1)$$
$$\cdots \mid NaPumpEnv(n_1+n_2-1) \mid NaPumpCell(n_1+n_2-1) \mid NaPump(n_1+n_2-1)$$
$$\mid KPumpEnv(n_1+n_2) \mid KPumpCell(n_1+n_2) \mid KPump(n_1+n_2)$$
$$\cdots \mid KPumpEnv(n_1+n_2+n_3\text{-}1) \mid KPumpCell(n_1+n_2+n_3\text{-}1) \mid$$
$$KPump(n_1+n_2+n_3\text{-}1)$$
$$CreateATP \mid ConsumeADP$$

We present in detail some of the above processes. The others are written in a similar manner.

- $Cell(NaCell, KCell, AtP, ADP, NaEnv, KEnv, P)$ *is the system in which several quantities of ions are initialized.*
- *Each NaK-ATPase is described by three processes:*
  - $NaKPumpEnv(id) = site2[id]^{160}?(add(y_{na} : NaEnv))$
    then $site2[id]^{2}!\langle 2K\rangle$
    then $p[id]^{6}?(add(y_p : P))$
    then $NaKPumpEnv(id)$

The environment site of the pump contains the channel $site2[id]$ *used for receiving three ions of* Na^+ *and also for sending two ions of* K^+*, and also the channel* $p[id]$ *for receiving the produced* P *molecules. The sending and receiving operations modify also the number of ions present in the system of pumps: e.g., when sending two* K^+ *ions, an operation of the form* $KEnv = Kenv - 2$ *is performed, while receiving the* $y_{na} : NaEnv$ *ions an operation of the form* $NaEnv = NaEnv + 3$ *is performed due to the add function that is used to add the amount of received ions to the corresponding multiset.*

$*\ NaKPumpCell(id) = site1[id]^{(12,1)}!\langle(3Na, ATP)\rangle$
then $adp[id]^6?(add(x_{adp} : ADP))$
then $site1[id]^{150}?(add(x_k : KCell))$
then $NaKPumpCell(id)$

The cell site of the pump contains the channel $site1[id]$ *used for sending three ions of* Na^+ *and one* ATP *and also for receiving two ions of* K^+*, and also the channel* $adp[id]$ *for receiving the produced* ADP *molecules.*

$*\ NaKPump(id) = site1[id]^{(28,9)}?((x_{na} : NaCell, x_{atp} : ATP))$
then $adp[id]^0!\langle ADP\rangle$
then $site2[id]^{100}!\langle 2Na\rangle$
then $site2[id]^6?(y_k : KEnv)$
then $p[id]^0!\langle P\rangle$ then $site1[id]^{110}!\langle 2K\rangle$
then $NaKPump(id)$

This process describes the evolution of the pump, namely the transport of Na *and* K *ions between the environment and the cell.*

3 Timed Automata

Due to their simplicity, timed automata, extended with integer variables, structured data types, user defined functions, and channel synchronization, have been used by several tools (e.g., UPPAAL) for the simulation and verification of timed automata [2]. In what follows we consider a particular case of timed automata, namely we ignore the

time aspects as they are not relevant to our approach and will refer to timed automata as automata.

Syntax Assume a finite set of integer variables $\mathcal{C}$ ranged over by x, y, ... standing for data, and a finite alphabet Σ ranged over by a, b, ... standing for actions. A constraint is a conjunctive formula of constraints of the form $x \sim m$ for $x \in \mathcal{C}$, $\sim \in \{\leq, <, ==, >, \geq\}$, and $m \in \mathbb{N}$. The set of constraints, ranged over by g, is denoted by $\mathcal{B}(\mathcal{C})$.

Definition 1 *An* **automaton** $\mathcal{A}$ *is a tuple* $\langle N, n_0, E \rangle$*, where*

- N *is a finite set of nodes;*
- n_0 *is the initial node;*
- $E \subseteq N \times \mathcal{B}(\mathcal{C}) \times \Sigma \times \mathbb{N}^{\mathcal{C}} \times N$ *is the set of edges.*

$n \xrightarrow{g,a,r} n'$ *is a shorthand notation for* $\langle n, g, a, r, n' \rangle \in E$*. r denotes fresh assignments to variables after the transition is performed.*

Networks of Automata A network of automata is the parallel composition $\mathcal{A}_1 \mid \ldots \mid \mathcal{A}_n$ of a set of automata $\mathcal{A}_1, \ldots, \mathcal{A}_n$ combined into a single system. Synchronous communication inside the network is by handshake synchronization of input and output actions. In this case, the action alphabet Σ consists of $a?$ symbols (for input actions), $a!$ symbols (for output actions), and τ symbols (for internal actions). A detailed example is found in [9]. A network can perform action transitions (following an enabled edge). An action transition is enabled if all guards on the corresponding edges are satisfied.

Let u, v, ... denote assignments mapping $\mathcal{C}$ to naturals $\mathbb{N}$. $g \models u$ means that the values u satisfy the guard g. Let n_i stand for the ith element of a node vector n, and $n[n'_i/n_i]$ for the vector n with n_i being substituted with n'_i. A network state is a pair $\langle n, u \rangle$, where n denotes a vector of current nodes of the network (one for each automaton), and u is an assignment storing the current values of all network integer variables.

Definition 2 *The operational semantics of an automaton is a transition system where states are pairs* $\langle n, u\rangle$ *and transitions are defined by the rules:*

- $\langle n, u\rangle \xrightarrow{\tau} \langle n[n'_i/n_i], u'\rangle$ *if* $n_i \xrightarrow{g,\tau,r} n'_i$, $g \models u$ *and* $u' = r[u]$*;*
- $\langle n, u\rangle \xrightarrow{\tau} \langle n[n'_i/n_i][n'_j/n_j], u'\rangle$ *if there exist* $i \neq j$ *such that*
  1. $n_i \xrightarrow{g_i,a?,r_i} n'_i$, $n_j \xrightarrow{g_j,a!,r_j} n'_j$, $g_i \wedge g_j \models u$,
  2. $u' = r_i[r_j[u]]$.

4 Relating BioMaxP to Automata

In order to use existing tools such as Uppaal for the verification of complex systems of parallel pumps, we establish a relationship between BioMaxP and automata.

Building an automaton for each process: Given a process P without the parallel operator at the top level, we associate to it an automaton $\mathcal{A} = \langle N, n_0, E\rangle$, where $n_0 = l_0$, $N = \{l_0\}$, $E = \emptyset$. The initial values of the BioMaxP system composed of P are set as the initial values of the automaton $\mathcal{A}$. The nodes of the associated automata are labelled using a fresh label l, and an index such that the nodes are uniquely labelled in this automaton (we start with the index 0, and increment it when necessary). The components N and E are updated depending on the structure of process P:

- for $P = a^{min}!\langle v\rangle$ then P_1 we have
  - $N = N \cup \{l_{i+1}\}$ where $i = max\{j \mid l_j \in N\}$;
    - The added node l_{i+1} indicates the execution of the process P, leading to P_1.
  - $E = E \cup \{n, min \leq |T|, a!, , l_{i+1}\}$;
    - If $i > 0$ it means that the automaton already contains some edges, and the process P was launched from the then branch of a process P'. Since the translation is made depending on the structure of the processes, it means that the action leading to P is already modelled

in the automaton. If $P' = b^{min'}!\langle w\rangle$ then P or $P' = b^{max'}?(u:T')$ then P, then the action of P' is modelled by an edge with the last component l_k, and thus $n = l_k$.
  - Otherwise, $n = l_0$.

  The edge encodes the then branch leading to process P_1. Channel a is an urgent channel (communication takes place as soon as possible).
- for $P = a^{max}?(f(u:T))$ then P_1 we have
  - $N = N \cup \{l_{i+1}\}$ where $i = max\{j \mid l_j \in N\}$;
  - $E = \begin{cases} E \cup \{l_i, |T| \leq max, a!, |T| = |T| - |u|, l_{i+1}\}, \text{if } f = id; \\ E \cup \{l_i, |T| \leq max, a!, |T| = |T| + |u|, l_{i+1}\}, \text{if } f = add. \end{cases}$

  A similar reasoning as for the previous case. Depending on function f, ions are removed or added from the number representing the existing ions of type T.
- for $P = P_1 \mid \ldots \mid P_k$, $k > 1$, and P_j does not contain operator $\mid$ at top level, then
  - $N = N \cup \{l_{i+1}\}$ where $i = max\{j \mid l_j \in N\}$;
    - If P contains some indexed nodes l (namely $l_0, \ldots, l_i$), then add l_{i+1} to N.
  - $E = E \cup \{n, , a!, \{x = 0\}, l_{i+1}\}$;
    - If $i > 0$, using a similar argument as for the communication actions, it holds that $n = l_k$. We use a new channel labelled a as a broadcast channel, in order to start at the same time all the parallel processes from P.
    - Otherwise, $n = l_0$.

  The new edge leads to process P_1. For each of the other processes P_j, $j > 1$, a new automaton $\mathcal{A}_j = \langle N_j, n_{j0}, E_j, I_j\rangle$ is build, where:
    - $n_{j0} = l_0$; $N_j = \{l_0, l_1\}$; $E_j = \{l_0, , a?, \{x = 0\}, l_1\}$; $I_j(l_0) = \emptyset$.

  The automaton is constructed recursively using the definition of P_j.

Building an automaton for each process leads to the next result about the equivalence between a BioMaxP process P and its corresponding automaton $\mathcal{A}_P$ in state $\langle n_P, u_P\rangle$ (i.e., $(\mathcal{A}_P, \langle n_P, u_P\rangle)$). Their transition

systems differ not only in transitions, but also in states; thus, we adapt the notion of bisimilarity:

Definition 3 *A symmetric relation $\sim$ between* BioMaxP *processes and their corresponding automata is a bisimulation if whenever $(N, (\mathcal{A}_N, \langle n_N, u_N\rangle)) \in\sim$ if $P \xrightarrow{\lambda} P'$, then $\langle n_P, u_P\rangle \xrightarrow{\tau} \langle n_{P'}, u_{P'}\rangle$ and $(P', (\mathcal{A}_{P'}, \langle n_{P'}, u_{P'}\rangle)) \in\sim$ for some P'.*

After defining bisimulation, we can state the following result.

Theorem 1 *Given a* BioMaxP *process P, there exists an automata $\mathcal{A}_P$ with a bisimilar behaviour. Formally, $P \sim \mathcal{A}_P$.*

Proof. [Sketch] The construction of the automaton simulating a given BioMaxP process is presented above. A bisimilar behaviour is given by the fact that a communication rule is matched by a synchronization between the edges obtained by translations. □

Thus, the size of an automata $\mathcal{A}_P$ is polynomial with respect to the size of a BioMaxP process P, and the state spaces have the same number of states.

Reachability Analysis. Qualitative properties abstract away from any quantitative information like time aspects or energy costs of targeted biological systems. One of the most useful question to ask about an automaton is the reachability of a given set of final states. Such final states may be used to characterize safety properties of a system.

Definition 4 *For an automata with initial state $\langle n_0, u_0\rangle$, $\langle n, u\rangle$ is reachable if and only if $\langle n_0, u_0\rangle \xrightarrow{\tau}{}^{*} \langle n, u\rangle$. More generally, given a constraint $\phi \in \mathcal{B}(\mathcal{C})$ if $\langle n, u\rangle$ is reachable for some u satisfying ϕ, then a state $\langle n, \phi\rangle$ is reachable.*

The reachability problem is decidable [4]. The reachability problem can be also defined for BioMaxP networks.

Definition 5 *Starting from a* BioMaxP *process* P_0*, a process* P_1 *is reachable if and only if* $P_0 \overset{\lambda}{\rightarrow}{}^* P_1$.

The following result is a consequence of Theorem 1.

Corollary 1 *For a* BioMaxP *process, the reachability problem is decidable.*

5 Conclusion

Previously, we provided a formal description of the sodium-potassium ion transport across cell membranes in terms of the π-calculus [8]. In [7], the transfer mechanisms were described step by step, and a software tool called Mobility Workbench [11] was used to verify some properties of the described system formed of only one pump. Inspired by the functioning of this pump, we introduced and studied a ratio-based type system using thresholds in a bio-inspired framework [3]. The aim was to avoid errors in the definition of the formal models used to mimic the evolution of some biologic processes.

In this paper we try to unify and extend our previous attempts to model the movement of ions in the sodium-potassium-pump by using BioMaxP , a simple and elegant approach able to capture the quantitative aspects (e.g., number of ions) and abstract conditions associated with evolution (e.g., the number of ions is between certain thresholds). This approach facilitates a better understanding of the processes happening in a cell viewed as a complex system of ion pumps working in parallel. The novelty is that we are able to model systems consisting of more than just a NaK pump by adding different amounts of other types of ion pumps.

Acknowledgements. The work was supported by a grant of the Romanian National Authority for Scientific Research, project number PN-II-ID-PCE-2011-3-0919.

References

[1] B. Alberts, A. Johnson, J. Lewis, D. Morgan, M. Raff, K. Roberts, P. Walter. *Molecular Biology of the Cell*, 6th edition, Garland Science, New York (2014).

[2] R. Alur, D.L. Dill. A Theory of Timed Automata. *Theoretical Computer Science* **126**, 183–235 (1994).

[3] B. Aman, G. Ciobanu. Behavioural Types Inspired by Cellular Thresholds. *Lecture Notes in Computer Science* **8368**, 1–15 (2014).

[4] J. Bengtsson, W. Yi. Timed Automata: Semantics, Algorithms and Tools. *Lecture Notes in Computer Science* **3098**, 87–124 (2004).

[5] D. Besozzi, G. Ciobanu. A P System Description of the Sodium-Potassium Pump. *Lecture Notes in Computer Science* **3365**, 210–223 (2005).

[6] B. Bloom, S. Istrail, A.R. Meyer. Bisimulation Can't Be Traced: Preliminary Report. *In 15th ACM Symposium on Principles of Programming Languages*, 229–239, 1988.

[7] G. Ciobanu. Software Verification of the Biomolecular Systems. in *Modelling in Molecular Biology*, Natural Computing Series, Springer, 40–59 (2004).

[8] G. Ciobanu, V. Ciubotariu, B. Tanasă. A π-calculus Model of the Na Pump, *Genome Informatics* **13**, 469–472 (2002).

[9] T.A. Henzinger, X. Nicollin, J. Sifakis, S. Yovine. Symbolic Model Checking for Real-time Systems. *Information and Computation* **111**, 192–224 (1994).

[10] F. Moller. *Axioms for Concurrency*. PhD Thesis, Department of Computer Science, University of Edinburgh, 1989.

[11] B. Victor, F. Moller. The Mobility Workbench - A Tool for the π-Calculus. *Lecture Notes in Computer Science* **818**, 428–440 (1994).

Bogdan Aman, Gabriel Ciobanu Received July 21, 2015

Romanian Academy, Institute of Computer Science
Blvd. Carol I no.11, 700506 Iaşi, Romania
E–mail: baman@iit.tuiasi.ro, gabriel@info.uaic.ro