Computational Power of Chemical Kinetics in Living Cells

Gabriel Ciobanu (invited talk)
Joint work with Bogdan Aman
Romanian Academy
Institute of Computer Science
Blvd. Carol I no.8, 700505 Iaşi, Romania
baman@iit.tuiasi.ro, gabriel@info.uaic.ro

ABSTRACT

Membrane computing is a branch of natural computing working with parallel and nondeterministic computing models abstracting the living cell. We present mobile membranes in which the movement is provided by rules inspired by cells endocytosis and exocytosis. Their computational power is compared with the classical notion of Turing computability. These mobile membranes can algorithmically solve hard problems in polynomial time. Considering proteins on the surface of mobile membranes, we can increase the modelling power of mobile membranes.

Categories and Subject Descriptors

F.4 [**Computation by Abstract Devices**]: Models of computation, Modes of Computation (parallel)

General Terms

Bio-inspired models of computation, cell biology

Keywords

Membrane computing, Turing machine, endocytosis, exocytosis

1. INTRODUCTION

The theory of computational complexity involves limitations on the information processing. The structure and organization of biological systems is different from that of digital computers. The parallelism of biological systems may provide a speed-up for information processing. Therefore a biological system working as a computer could exhibit a dynamic behaviour which produces solutions of intractable problems. Moreover, the natural exponential division of biological systems is important in allowing polynomial time algorithms for these problems.

Chemical kinetics is generally concerned with the time-evolution of a system specified by a set of coupled chemical reactions. Although the reaction equations capture the key interactions between the competing species, on their own they are not enough to determine the full system dynamics.

There are various ways to represent a biological system:

- Biologists use diagrammatic schemes coupled with verbal explanations about the qualitative information of the depicted mechanisms.
- Applied mathematicians prefer to work with systems of ordinary or partial differential equations. These are more precise and fully quantitative, but also have a number of disadvantages: e.g., are too low level description, encode more than the essential features of the model.
- Biochemists view systems as networks of coupled chemical reactions. These networks are sufficiently general and detailed so that, once the kinetics have been specified, they can be used directly to construct full dynamic simulations of the system behaviour on a computer. A general chemical reaction takes the form

$$s_1X_1 + s_2X_2 + \ldots + s_nX_n \rightarrow r_1Y_1 + r_2Y_2 + \ldots + r_mY_m$$

A rather different approach is provided by membrane computing [14], a new research field which belongs to the more general area of natural computing. Membrane computing has been introduced in [13], and in 2003 has been selected by Thomson Institute for Scientific Information as a "fast emerging research front in computer science".

Membrane systems represent parallel and nondeterministic computing models inspired by cell biology. Membrane systems are complex hierarchical structures with a flow of materials and information which underlies their functioning, involving parallel application of rules, communication between membranes and membrane dissolution. The structure of the cell is represented by a set of hierarchically embedded regions, each delimited by a surrounding boundary (called membrane), and all contained inside a skin membrane. A membrane without any other membrane inside is said to be elementary, while a membrane with other membranes inside is said to be composite. Multisets of objects are distributed inside these regions, and they can be modified or communicated between adjacent compartments. Objects represent the formal counterpart of molecular species (ions, proteins, etc.) floating inside cellular compartments, and they are described by means of strings over a given alphabet. Evolution rules represent the formal counterpart of chemical reactions, and are given in the form of rewriting rules which operate on the objects, as well as on the structure by endocytosis, exocytosis and elementary division.

.

BICT 2014, December 01-03, Boston, United States

DOI 10.4108/icst.bict.2014.258046

A *computation* is performed in the following way: starting from an initial structure, the system evolves by applying the rules in a nondeterministic and maximally parallel manner. The maximally parallel way of using the rules means that in each step we apply a maximal multiset of rules, namely a multiset of rules such that no further rule can be added to the multiset. A rule is applicable when all the objects that appear on its left-hand side are available in the region where the rule is placed (the objects are not used by other rules applied in the same step). Due to the competition for available objects, some rules are applied nondeterministically. A halting configuration is reached when no rule can be applied anymore; the result is then given by the number of objects (in a specified region).

Mobile membranes represent a parallel and nondeterministic computing model inspired by cells and their movements in which mobility is given by specific endocytosis and exocytosis rules. They were investigated and studied in [2], where the computational power of mobile membranes and some links to process calculi are presented. We have introduced various membrane systems in which mobility is the key issue. The main mobility rules describing the evolution of the structure are endocytosis (moving an elementary membrane inside a neighbouring membrane), and exocytosis (moving an elementary membrane outside the membrane where it is placed); other mobility rules are given by pinocytosis and phagocytosis. We related systems of mobile membranes to some existing process calculi (mobile ambients, picalculus, brane calculi) and to Petri nets. We defined some concepts inspired from process calculi in the framework of (mobile) membrane computing. To obtain more interesting and accurate models, we added timers inspired by cell lifetime to membranes. We related these membrane systems with timers to timed mobile ambients. By describing biological systems using mobile membranes, many interesting questions regarding the biological systems can get precise answers. Using software tools and process calculi bisimulations, (potentially infinite) system behaviour can be investigated, and this fact could be interesting from a biological viewpoint.

A standard approach in membrane computing is to look for membrane systems with a minimal set of ingredients which are powerful enough to achieve the full computational power of Turing machines. We investigated the computational power and efficiency of our formalisms by using formal languages, register machines and other ingredients of computability theory. Many formal machine models (e.g., Turing machines) have an infinite number of memory locations, while membrane systems are computing devices of finite size having a finite description with a fixed amount of initial resources (membranes and objects). However, the biological operations allow an initial membrane system to evolve to a possibly infinite family of membrane systems obtained by division in order to solve a (decision) problem. We use notions from classical computational complexity theory adapted to membrane computing. The purpose of computational complexity theory is to provide bounds on the amount of resources necessary for any procedure (algorithm) solving a problem. Since mobile membranes are inspired by the biological endocytosis and exocytosis processes, the bounds of resources represent the quantitative requirements for solving a problem.

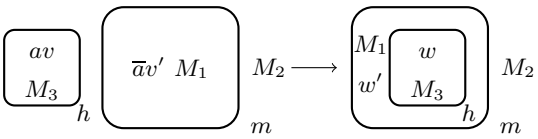
2. MOBILE MEMBRANE SYSTEMS

Mobile membranes were defined in 2005 having the following features: a spatial structure consisting of a hierarchy of membranes, multisets of objects placed inside membranes and the biologically inspired rules describing the mobility of membranes inside the structure. These rules are: endocytosis (taking a membrane that is outside a neighbouring membrane and moving it inside), exocytosis (moving a membrane outside the membrane where it is placed), and elementary division (creating two copies of an elementary membrane).

DEFINITION 1. *A system of mobile membranes with elementary division is a construct* $\Pi = (O, H, \mu, w_1, \ldots, w_n, R)$, *where:*

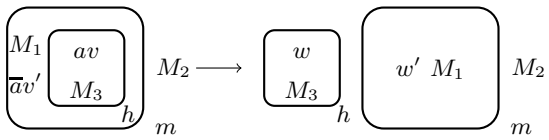
1. $n \geq 1$ *is the number of membranes in the initial configuration;* $(n \leq |H|)$
2. O *is an alphabet (its elements are called objects);*
3. H *is a finite set of labels for membranes;*
4. $\mu \subset H \times H$ *is the initial membrane structure, such that* $(i, j) \in \mu$ *denotes that the membrane labelled by* j *is contained in the membrane labelled by* i*; the hierarchical structure denoted by* μ *is described by brackets in our linear notation (e.g.,* $\mu = \{(m, h)\}$ *is* $[\,[\,]_h]_m$*).*
5. $w_1, \ldots, w_n \in O^*$ *are multisets of objects placed in the* n *regions of* μ*;*
6. R *is a finite set of developmental rules of the following forms, with* $h, m \in H$*,* $a, \overline{a} \in O$*,* $v, v', w, w' \in O^*$*,* $|v| \leq 1$*,* $|v'| \leq 1$*,* M_1 *and* M_2 *denote (possibly empty) multisets of objects, elementary and composite membranes, while* M_3 *denotes a (possibly empty) multiset of objects.*

   (a) $[av]_h[\overline{a}v']_m \rightarrow [\,[w]_h w']_m$ endocytosis (endo)



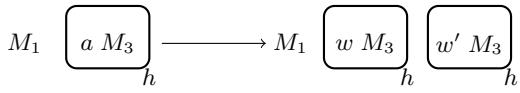
   An elementary membrane labelled h *enters the adjacent membrane labelled* m *under the control of the multisets of objects* av *and* $\overline{a}v'$*. The labels* h *and* m *remain unchanged, and the multisets of objects* av *and* $\overline{a}v'$ *are replaced by the multisets of objects* w *and* w'*, respectively.*

   (b) $[\overline{a}v'[av]_h]_m \rightarrow [w]_h[w']_m$ exocytosis (exo)



   An elementary membrane labelled h *exits a membrane labelled* m*, under the control of the multisets of objects* av *and* $\overline{a}v'$*. The labels of the two membranes remain unchanged, and the multisets of objects* av *and* $\overline{a}v'$ *are replaced by the multisets of objects* w *and* w'*, respectively.*

(c) $[a]_h \rightarrow [w]_h[w']_h$ elementary division (div)



An elementary membrane labelled h, containing an object a, is divided into two membranes labelled h. A copy of each object from membrane h is placed inside the new created membranes, except for object a which is replaced by the multisets of objects w and w'.

A configuration of a system with mobile membranes is a finite tree with nodes labelled by the elements of H, each node being assigned a finite multiset of objects. The rules are applied according to the following principles:

1. All the rules are applied in parallel by nondeterministically choosing the rules, the membranes and the objects in such a way that the parallelism is maximal; this means that in each step a set of rules is applied such that no further rule can be added to the set.
2. The membrane m from the rules of type $(a) - (c)$ is said to be passive (identified by the use of an object $\overline{a}$), while the membrane h is said to be active (identified by the use of an object a). In any step of a computation, any object and any active membrane can be involved in at most one rule, while passive membranes are not considered to be involved in the use of the rules (hence they can be used by several rules at the same time as passive membranes).
3. When a membrane is moved across another membrane by endocytosis or exocytosis, all objects contained in it are moved.
4. All the objects and membranes which do not evolve at a given step are passed unchanged to the next configuration of the system.

Transitions among the configurations of the system, transforming a configuration into a new one, are obtained by using the rules described above. A computation is a sequence of transitions starting from the initial configuration (the initial membrane structure and the initial distribution of objects within regions). A computation is successful if it halts (it reaches a configuration where no rule can be applied). The multiplicity vector of the multiset of objects from a special membrane called the output membrane is considered as a result of the computation. Thus, the result of a halting computation consists of all the vectors describing the multiplicity of objects from the output membrane; a non-halting computation provides no output.

3. COMPUTATIONAL POWER OF MOBILE MEMBRANES

Formal languages [16] and automata theory [12] are usually used to introduce abstract computing devices and investigating their computational power. Turing machines represent a classical model of computation. Actually, a Turing machine is a mathematical model that mechanically operates 0 and 1 symbols on an infinite tape according to a table of three simple operations: go left, go right, change symbol. Despite its simplicity, a Turing machine can simulate the logic of *any* computer algorithm. According to Church-Turing thesis, computable functions are exactly the functions that can be calculated using a Turing machine [18].

Minsky introduced the concept of register machines by showing that the power of Turing machines can be achieved by such abstract machines using a small number of registers for storing arbitrarily large non-negative integers. A register machine runs a program consisting of labelled instructions which encode simple operations for updating the content of the register: increment, decrement and a halt operation [12].

We were able to prove in [9] that a biologically inspired formalism based on chemical kinetics are also able to calculate *any* computable function.

THEOREM 1. *A system with three mobile membranes using endo and exo rules with an additional priority relation between rules has the same computational power as a Turing machine.*

The priority relation between rules is used when checking if some objects are present inside a membrane. The biologically motivated priority relation corresponds to the fact that certain reactions/reactants are more active than others, and can be interpreted as a competition for reactants/objects. This is calculated by looking at the cardinality of the objects in a specified *output membrane* of the mobile membrane systems at the end of a halting computation. The proof is based on the observation that each set of natural numbers generated by arbitrary grammars is the range of a partial recursive function. In [17] it was proven that a 3-register machine can compute any partial recursive function of one variable. Starting from this results we investigated in [5] new computability results using movement based on agreements. For systems of mobile membranes using endocytosis and exocytosis, we get the same computational power; the next result shows that it is possible to get similar results as for register machines. We have proven that agreement leads to a minimal number of membranes in terms of mobility: we have a skin membrane and two inner membranes able to move according to the endocytosis and exocytosis rules.

In Theorem 1 we have shown that systems with three mobile membranes using *endo* and *exo* rules have the same computational power as Turing machines, namely are able to generate all the sets of natural numbers. Another related result is presented in [1], where we prove that these systems generate the recursive enumerable sets of vectors of natural numbers.

THEOREM 2. *A system with three mobile membranes using endo and exo rules is able to generate the recursive enumerable sets of vectors of natural numbers.*

L systems [10] represent a biologically inspired branch in formal languages that provide a parallel and non-sequential grammatical formalism. L systems model biological growth and because growth happens in multiple areas of an organism, growth is parallel. Evolution in these systems are intended to capture cell divisions in multicellular organisms, where many divisions may occur at the same time. The investigations of L systems are an important and wide area in the theory of formal languages. The study of L languages has resulted in a language hierarchy, namely the L system

hierarchy [15]. We have proven in [5] that L systems can be simulated by systems of mobile membranes.

THEOREM 3. *A system with three mobile membranes using endo and exo rules is able to compute the same language as an L system.*

In [9] we studied the computational power of mobile membrane systems by using the operations of forced endocytosis $fendo$ and forced exocytosis $fexo$, and determined the border condition for achieving computational completeness: four membranes provide Turing completeness, while three membranes do not. Moreover, we proved that that the division operation (which is crucial in solving NP-complete problems) does not provide computational completeness. However, Turing completeness can be achieved with pairs of operations (exocytosis, inhibitive endocytosis *iendo*) and (inhibitive exocytosis $iexo$, endocytosis) with four membranes.

The next result defines the border for computational completeness while using the operations $endo, exo, fendo$ and $fexo$. In [9] we proved that four membranes are sufficient for computational completeness by using the fact that each recursively enumerable language can be generated by a matrix grammar in the strong binary normal form [14].

THEOREM 4. *A system with four mobile membranes using a weaker form of endo, exo rules with additional $fendo$ and $fexo$ rules has the same computational power as a Turing machine.*

We also proved in [9] that using either inhibitive endocytosis and exocytosis or endocytosis and inhibitive exocytosis, it is possible to obtain computational completeness with four membranes.

THEOREM 5.
1. *A system with four mobile membranes using iendo and exo rules has the same computational power as a Turing machine.*
2. *A system with four mobile membranes using endo and iexo rules has the same computational power as a Turing machine.*
3. *A system with four mobile membranes using restricted forms of iendo and exo rules has the same computational power as a Turing machine.*

We have also proven in [9] that going from four to three membranes leads to a smaller computational power for the same types of rules as in item 3 of the previous theorem.

THEOREM 6. *A system with three mobile membranes using restricted forms of iendo and exo rules has a strictly smaller computational power as a Turing machine.*

In [9] we proved that restrictive mobility is not powerful without division (Theorem 7 below), while division offers limited power in the absence of restrictive mobility (Theorem 8 below).

THEOREM 7. *A system with any small number of mobile membranes without using division rules computes a finite set of natural numbers.*

THEOREM 8. *A system with any small number of mobile membranes using division rules has a computational power smaller than a context-sensitive system.*

The next theorem puts together the restricted division with restricted mobility, and compare them with $ET0L$.

THEOREM 9. *A system with five mobile membranes using a restricted form of endo, exo and div rules can compute more than an L system.*

4. SOLVING NP-COMPLETE PROBLEMS POLYNOMIALLY BY USING MOBILE MEMBRANES

According to [7], the NP-complete problems are classified into *weak* (e.g., Partition, Subset Sum, Knapsack) and *strong* (e.g., SAT, Clique, Bin Packing, Common Algorithmic Problem). We showed that mobile membrane systems are efficient model of computation as they can solve both weak and strong NP-complete problems in polynomially time.

4.1 SAT Problem

The SAT problem refers to the satisfiability of a propositional logic formula in conjunctive normal form (CNF). Let $\{x_1, x_2, \ldots, x_n\}$ be a set of propositional variables. A formula in CNF is of the form $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ where each C_i, $1 \leq i \leq m$ is a disjunction of the form $C_i = y_1 \vee y_2 \vee \cdots \vee y_r$ ($r \leq n$), where each y_j is either a variable x_k or its negation $\neg x_k$. We proposed a semi-uniform linear time solution to the SAT problem using the operations of *endo*, *exo* and elementary division. The maximal parallelism and movement based on agreement are essential in order to obtain such a solution. The first one is used in order to check, after the generation stage, in only one step if all possible assignments of variables satisfy a certain clause C_i. The second one is also used in the checking stages in order to avoid overlapping of solutions, and also in the generation of the answer.

For any instance of SAT we showed in [5] how to construct a system of mobile membranes which solves it.

THEOREM 10. *Using mobile membrane systems, SAT can be solved in a polynomial number of steps.*

4.2 4QBF Problem

The Arithmetical Hierarchy is usually developed with the universal ($\forall$) and existential ($\exists$) quantifiers restricted to the integers. Levels in the Arithmetical Hierarchy are labelled as Σ_n if they can be defined by an expression beginning with a sequence of n alternating quantifiers starting with $\exists$. Similar expressions that start with $\forall$ are labelled as Π_n. Σ_0 and Π_0 are defined as having no quantifiers and are equivalent. Σ_1 and Π_1 only have the single quantifier $\exists$ and $\forall$, respectively. We only need to consider alternating pairs of the quantifiers $\forall$ and $\exists$ because two quantifiers of the same type occurring together are equivalent to a single quantifier.

A complete problem for Σ_k is satisfiability of quantified Boolean formulas with k alternations of quantifiers (abbreviated kQBF). This is the version of the boolean satisfiability problem for Σ_k. In this problem, we are given a Boolean formula φ with variables partitioned into k sets $X_1, ..., X_k$. We have to determine if it is true that $\exists X_1 \forall X_2 \exists X_3 \ldots \psi$, where ψ is in CNF. That is, is there an assignment of values to variables in X_1 such that, for all assignments of values in X_2, there exists an assignment of values to variables in $X_3 \ldots$, φ is true? The variant above is complete for Σ_k. The

variant in which the first quantifier is $\forall$, the second is $\exists$, and so on is complete for Π_k.

For any instance of 4QBF we showed in [6] how to construct a system of mobile membranes which solves it.

THEOREM 11. *Using mobile membrane systems, 4QBF can be solved in a polynomial number of steps.*

4.3 Partition Problem

The problem can be enounced as follows: given a finite set A, a weight function $g : A \to \mathbb{N}$, decide whether or not there exists a partition of A into two subsets such that they have the same weight.

For any instance of the 2-Partition problem we showed in [3] how to construct a system of mobile membranes which solves it.

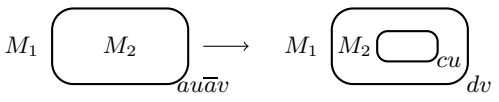
THEOREM 12. *The 2-Partition problem can be solved by a semi-uniform family of mobile membrane systems in linear time with respect to the input size.*

5. MOBILE MEMBRANES WITH PROTEINS ON SURFACE

Mobile membranes with proteins on surface represent a rule-based formalism involving parallelism and mobility we introduced in order to model more specific biological systems [4]. The biologically inspired rules taken from the immune system [8] and describing the mobility of membranes inside the structure are: pinocytosis (engulfing zero external membranes), phagocytosis (engulfing just one external membrane), and exocytosis (expelling the content of a membrane outside the membrane where it is placed). Pinocytosis and phagocytosis represent different types of endocytosis. In these rules membranes agree on their movement by using complementary objects a and $\overline{a}$. Biologically speaking, the objects a and their corresponding co-objects $\overline{a}$ fit properly.

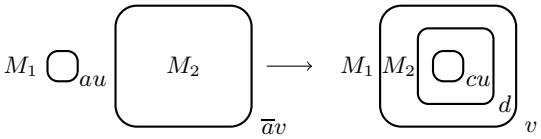
The evolution from a configuration of mobile membranes with proteins on surface to another is provided by a set R of rules defined as follows, where we use similar notations as in Definition 1:

- $[\,]_{a\,u\,\overline{a}\,v} \to [[\,]_{c\,u}]_{d\,v}$ pino



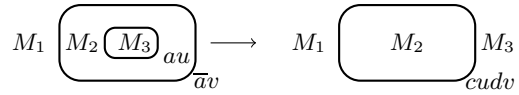
  An object a together with its complementary object $\overline{a}$ indicate the creation of an empty membrane within the membrane on which a and $\overline{a}$ objects are attached. Imagine that this initial membrane buckles towards the inside, and pinches off by breaking the connection between a and $\overline{a}$. The multiset of objects u on the new created (empty) membrane is transferred from the initial membrane. The objects a and $\overline{a}$ can be modified during this step into the multisets c and d, respectively. On the surface of the membrane appearing in the left hand side of the rule there are some objects (others than $au\overline{a}v$) which are ignored; these objects are also not specified on the right hand side of the rule, being randomly distributed between the two resulting membranes. By M_1 and M_2 are denoted (possible empty) multisets of elementary and composite membranes.

- $[\,]_{a\,u} \| [\,]_{\overline{a}\,v} \to [[[\,]_{c\,u}]_d]_v$ phago



  An object a together with its complementary object $\overline{a}$ indicate a membrane (the one with $\overline{a}$ on its surface) "eating" an elementary membrane (the one with a on its surface). The membrane having $\overline{a}$ and v on its surface wraps around the membrane having a and u on its surface. An additional membrane is created around the eaten membrane; the objects a and $\overline{a}$ are modified during this evolution into the multisets c and d (the multiset c corresponds to a and remains on the eaten membrane, while the multiset d corresponds to $\overline{a}$ and is placed on the new created membrane). On the surface of the membranes appearing in the left hand side of the rule there are some objects (others than au and $\overline{a}v$) which are ignored; these objects are also not specified on the right hand side of the rule. The objects appearing on the membrane having initially the object a on surface remain unchanged, while the objects appearing on the membrane having initially the object $\overline{a}$ on surface are randomly distributed between the two resulting membranes (the ones with d and v). By M_1 and M_2 are denoted (possible empty) multisets of elementary and composite membranes.

- $[[\,]_{a\,u}]_{\overline{a}\,v} \to [\,]_{c\,u\,d\,v}$ exo



  An object a together with its complementary object $\overline{a}$ indicate the merging of a nested membrane with its surrounding membrane. Imagine that the connection between a and $\overline{a}$ represent the point where the membranes connect each other. In this merging process (which is a smooth and continuous process), the membrane having the multiset $a\,u$ on its surface is expelled to the outside, and all objects of the two membranes are united into a multiset on the membrane which initially contained v. The objects a and $\overline{a}$ can be modified during this evolution into the multisets c and d, respectively. If the membrane having on its surface the object a is composite, then its content is released near the newly merged membrane after applying the rule. On the surface of the membranes appearing in the left hand side of the rule there are some objects (others than au and $\overline{a}v$) which are ignored; these objects are also not specified on the right hand side of the rule, being moved on the resulting membrane. By M_1, M_2 and M_3 are denoted (possible empty) multisets of elementary and composite membranes.

5.1 LDL Degradation Pathway Using Mobile Membranes

LDL is one of several complexes carrying cholesterol through the bloodstream. An LDL particle is a lipoprotein complex that contains one thousand or more cholesterol molecules in the form of cholesteryl esters. A monolayer of phospholipid surrounds the cholesterol and contains a single molecule of a large protein apolipoprotein B (known as apoB). In a

receptor-mediated endocytosis, a cell engulfs a particle of low-density lipoprotein from the outside. To do this, the cell uses receptors that specifically recognize and bind to the LDL particle. By this mechanism, cells acquire from the bloodstream the cholesterol required for the membrane synthesis that occurs during the cell growth. The degradation of LDL particles is realized in five steps (see Figure 1):

1. Cell-surface LDL receptors bind to an apoB protein of an LDL particles forming an receptor-ligand complex.
2. Clathrin-coated pits containing receptor-LDL complexes are pinched off.
3. After the vesicle coat is shed, the uncoated endocytic vesicle (early endosome) fuses with the late endosome. The acidic pH inside causes a conformational change in the LDL receptor, freeing the bound LDL particle.
4. The late endosome fuses with the lysosome, and the proteins and lipids of the free LDL particle are broken apart by enzymes in the lysosome.
5. The LDL receptor recycles to the cell surface where, at the neutral pH of the exterior medium, the receptor undergoes a conformational change such that it can bind another LDL particle.

We illustrate how the LDL degradation pathway can be described in terms of mobile membranes with proteins on surface, and simulate all these steps. An LDL particle is described as $[\,]_{apoB\ cho}$ representing the monolayer of phospholipid that contains a single *apoB* protein, and cholesterol *cho*. A cell engulfing the LDL particle is described as $[[\,]_{lyso} \parallel [\,]_{late\ aux}]_{recep\ recep}$, where:

- $[\,]_{recep\ recep}$ represents the cell containing on its surface two receptors *recep* able to recognize an *apoB* protein; clathrin and others receptors of the cell are not use since we are not interested in their evolution;
- $[\,]_{lyso}$ represents the lysosome;
- $[\,]_{late\ aux}$ represents the late endosome, and *aux* is an auxiliary object in creating new membranes by pino and phago rules.

The initial configuration of the systems is

$$M_1 = [\,]_{apoB\ cho} \parallel [[\,]_{lyso\ aux} \parallel [\,]_{late\ aux}]_{recep\ recep}$$

The steps presented in Figure 1 are described by using the following rules:

1. $[\,]_{apoB} \parallel [\,]_{recep\ recep} \to [[[\,]_{apoB}]_{recep}]_{recep}$ (phago) $(recep = \overline{apoB})$
2. $[\,]_{recep} \parallel [\,]_{late\ aux} \to [[[\,]_{recep}]_{aux}]_{late}$ (phago) $(aux = \overline{recep})$
3. $[[\,]_{recep}]_{aux} \to [\,]_{recep1\ aux}$ (exo) $(aux = \overline{recep})$
4. $[[\,]_{aux}]_{late} \to [\,]_{aux4\ late}$ (exo) $(late = \overline{aux})$
5. $[\,]_{lyso\ aux} \parallel [\,]_{late} \to [[[\,]_{late}]_{aux1}]_{lyso}$ (phago) $(aux = \overline{late})$
6. $[[\,]_{recep1}]_{aux1} \to [\,]_{recep2\ aux2}$ (exo) $(aux1 = \overline{recep1})$

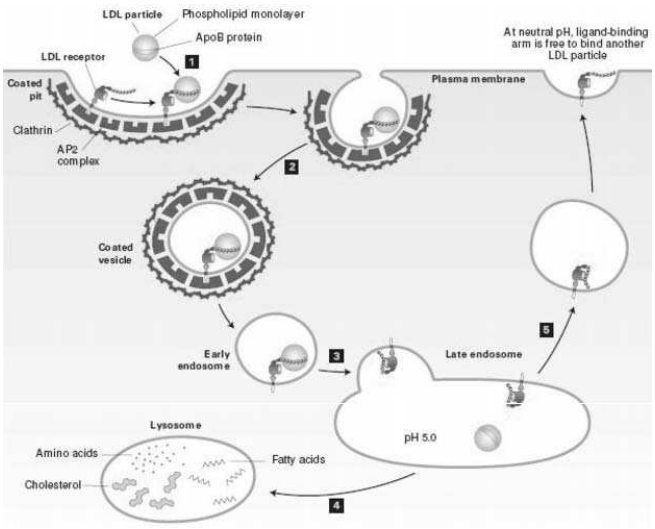


Figure 1: Endocytic Pathway for Low-Density Lipoprotein [11]

7. $[\,]_{late\ recep2\ aux2\ aux4} \to [[\,]_{late\ recep3\ aux4}]_{aux3}$ (pino) $(aux2 = \overline{recep2})$
8. $[[\,]_{aux3}]_{lyso} \to [\,]_{lyso\ aux}$ (exo) $(lyso = \overline{aux3})$
9. $[[\,]_{apoB}]_{lyso} \to [\,]_{lyso\ apoB}$ (exo) $(lyso = \overline{apoB})$
10. $[\,]_{late\ recep3\ aux4} \to [[\,]_{recep4\ aux4}]_{late}$ (pino) $(late = \overline{recep3})$
11. $[\,]_{aux4\ recep4} \to [[\,]_{recep5}]_{aux5}$ (pino) $(aux4 = \overline{recep4})$
12. $[[\,]_{aux5}]_{late} \to [\,]_{late\ aux}$ (exo) $(late = \overline{aux5})$
13. $[[\,]_{recep5}]_{recep} \to [\,]_{recep\ recep}$ (exo) $(recep = \overline{recep5})$

where $(recep = \overline{apoB})$ denotes an object *recep* that is complementary to an object *apoB*.

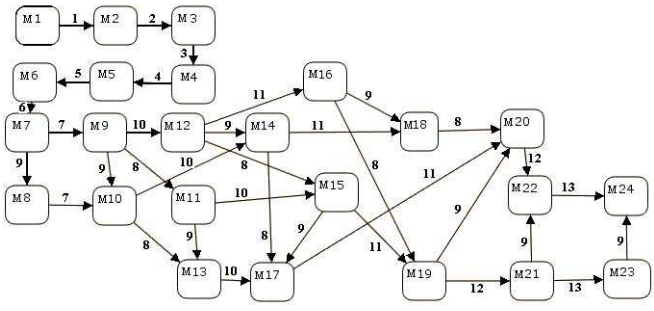


Figure 2: Evolution of the LDL degradation

The evolution of the LDL degradation could be represented graphically as in Figure 2. By $M_1, \ldots, M_{24}$ are denoted the possible configurations of the system, and on each arrow from a M_i to a M_j is placed the number of the rule

which is applied in order to evolve from M_i to M_j. To denote that an object *recep* changes its position and interacts with different objects, different notations are used (namely, $recep, recep_1, \ldots, recep_5$) in the evolution of the system. It is worth noting that the number of the applied rules to reach the configuration M_{24} starting from the configuration M_1 is always 13.

5.2 Simulating LDL Degradation by Using CPN Tools

Following [4], we present how the rules of mobile membranes used to model the LDL degradation pathway are simulated using the CPN Tools. To observe easier how the evolution takes place using CPN Tools, the system is simplified and only the transitions that eventually occur are used.

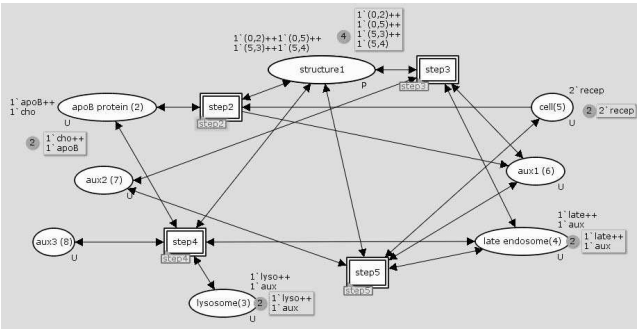


Figure 3: LDL Degradation Pathway in CPN Tools

A CPN model is always created in CPN Tools as a graphical drawing. Figure 3 describes the LDL degradation pathway model, namely the membrane configuration M_1 from Subsection 5.1. The diagram contains eight places, four substitution transitions (drawn as double-rectangular boxes), a number of directed arcs connecting places and transitions, and finally some textual inscriptions next to the places, transitions and arcs. The arc expressions are built from variables, constants, operators, and functions. When all variables in an expression are bounded to values of the correct type, the expression can be evaluated. In general, arc expressions may evaluate to a multiset of token colors. Next to each place is an inscription which determines the set of token colors (data values) that the tokens on that place are allowed to have. The set of possible token colors is specified by means of a type (as known from programming languages) which is called the color set of the place. By convention, the color set is written below the place. The place *structure*1 has the color set P, while all the others have the color set U; the color set P is used to model the structure of a membrane configuration (pairs of numbers of the form (i, j)), while the color set U is used to model the set of objects from a mobile membranes.
Color sets are defined using the CPN keyword *colset*:

$colset\ I = int;$
$colset\ P = product\ I * I;$
$colset\ U = with\ cho \mid apoB \mid lyso \mid late \mid aux \mid aux1 \mid aux2 \mid aux3 \mid aux4 \mid aux5 \mid recep \mid recep1 \mid recep2 \mid recep3 \mid recep4 \mid recep5;$

The inscription on the upper side of a place specifies the *initial marking* of that place. The inscription of the place *late endosome*(4) is $1`late + +1`aux$ specifying that the initial marking of this place consists of two tokens with the values *late* and *aux*. The symbols ‘ and ++ are operators used to construct a multiset of tokens. The infix operator ‘ takes a positive integer as its left argument, specifying the number of appearances of the element provided as the right argument. The operator ++ takes two multisets as arguments and returns their union (the sum of their multiplicities). The absence of an inscription specifying the initial marking means that the place initially contains no tokens. The marking of each place is indicated next to the place. The number of tokens on the place is shown in a small circle, and the detailed token colors are indicated in a box positioned next to the small circle.

The four transitions drawn as rectangles represent the events that can take place in the system. The names of the transitions are written inside the rectangles; these names have no formal meaning, but they are important for the readability of the model. In Figure 3 the names of the transitions are *step*2, *step*3, *step*4, and *step*5 indicating that each of these transitions simulates the corresponding steps of the LDL degradation pathway described in Figure 1.

A transition with double-line border is a substitution transition; each of them has a *substitution tag* positioned next to it. The substitution tag contains the name of a submodule which is related to the substitution transition. Intuitively, this means that the submodule presents a more detailed view of the behaviour represented by the substitution transition, and it is particularly useful when modeling large systems. To obtain a complete hierarchical model, it must be specified how the interface of each submodule is related to the interface of its substitution transition.

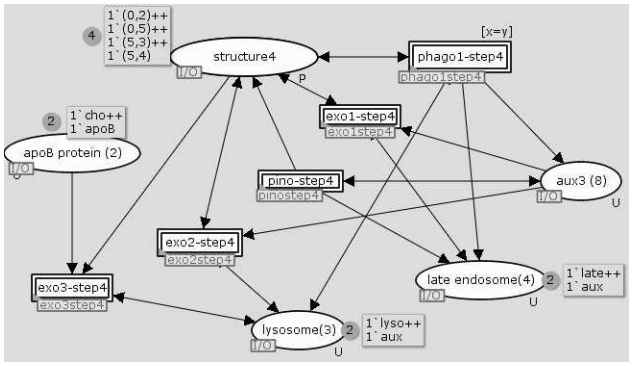


Figure 4: Step 4 Transition

For instance, behind the substitution transition *step*4 is another colored Petri net presented in Figure 4. For example, the substitution transition *phago*1-*step*4 simulates the mobile membrane rule 5 from the description of the LDL degradation pathway. It can be observed that the marking of places appearing in Figure 4 are similar with the one of the corresponding places of Figure 3.

In Figure 5, the enabled transition is *phagostep*2 that has a mark in the right corner. It removes a token *apoB* from place *apoB protein*(2), a token *recep* from place *cell*(5), and two tokens $(0, 2)$ and $(0, 5)$ from place *structure*. The arc expression of the input arc from the place *structure* are $(x, 2)$ and $(y, 5)$, and so they are tested using the test expression $[x{=}0, y{=}0]$. The test is performed in order to see that the

simulated membranes 2 and 5 have the same parent 0.

After firing the transition, a token *recep* is added to the place $aux1(6)$, while to the place *apoB protein*(2) a token *apoB* is added, and three tokens $(0,5)$, $(6,2)$ and $(5,6)$ are added to the place *structure*1.

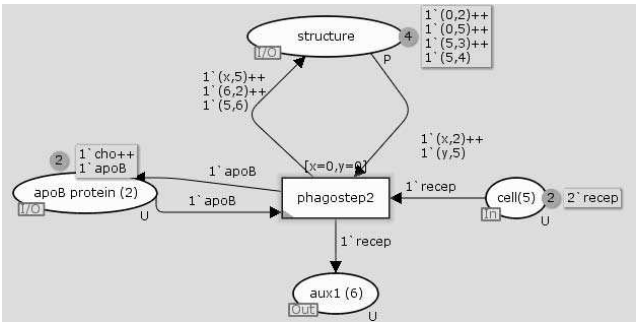


Figure 5: Phago Step 2 Transition

Since the properties of reachability, boundedness, liveness and fairness can be derived automatically by using CPN Tools, these results are of great help when studying similar properties for mobile membranes with proteins on surface. For instance, using the CPN Tools and the model for the LDL degradation pathway, it can be checked whether the configuration, in which the membrane marked by *apoB* is inside the membrane marked by *lyso*, can be reached.

Using CPN Tools for the LDL degradation pathway model, the following output file is obtained:

Home Markings: [24] Dead Markings: [24];
Dead Transition: None Live Transition: None
Fairness Properties: No infinite occurrence sequences,

meaning that always the configuration M_{24} is reached (home marking), the computation stops here (dead marking), and that there are no infinite occurrence sequences.

6. CONCLUSION

In this paper, the (abstract) operations describing the chemical kinetics in living cells are endocytosis and exocytosis. Using these operations, we prove that systems of three mobile membranes have the same computational power as a Turing machine, and so able to calculate any computable function. This is a minimal number of membranes, because we have a skin membrane and two inner membranes able to move according to endocytosis and exocytosis rules.

We also show that mobile membranes can solve computationally hard problems in polynomial time.

We have considered the biological process of low-density lipoprotein degradation pathway, and described it by using the mobile membranes with proteins on their surfaces. The translation of mobile membranes into colored Petri nets allows the use of software tools (CPN Tools) to analyze several behavioural properties: reachability, boundedness, liveness, fairness. By encoding biological systems in this way and by using complex software tools, several interesting biological behaviours can be investigated and various interesting biological questions can get a precise answer.

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

7. REFERENCES

[1] B. Aman, G.Ciobanu. Turing Completeness Using Three Mobile Membranes. *Lecture Notes in Computer Science* **5715**, 42–55 (2009).

[2] B. Aman, G. Ciobanu. *Mobility in Process Calculi and Natural Computing*. Springer (2011).

[3] B. Aman, G. Ciobanu. Solving a Weak NP-Complete Problem in Polynomial Time by Using Mutual Mobile Membrane Systems. *Acta Informatica* **48**, 409–415 (2011).

[4] B. Aman, G. Ciobanu. Mobile Membranes with Objects on Surface as Colored Petri Nets. *Lecture Notes in Computer Science* **7762**, 128-144 (2013).

[5] B. Aman, G. Ciobanu. Mobile Membranes: Computability and Complexity. *Lecture Notes in Computer Science* **8049**, 59–75 (2013).

[6] B. Aman, G. Ciobanu, S.N. Krishna. Solving the 4QBF Problem in Polynomial Time by Using the Biological-Inspired Mobility. *Lecture Notes in Computer Science* **7753**, 432–443 (2013).

[7] M. Garey, D. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. Freeman (1979).

[8] C.A. Janeway, P. Travers, M. Walport, M.J. Shlomchik. *Immunobiology - The Immune System in Health and Disease*. 5th edition, Garland Publishing (2001).

[9] S.N. Krishna, B. Aman, G. Ciobanu. On the Computability Power of Membrane Systems with Controlled Mobility. *Lecture Notes in Computer Science* **7318**, 626–635 (2012).

[10] A. Lindenmayer. Mathematical Models For Cellular Interaction in Development. *Journal of Theoretical Biology* **18**, 280–315 (1968).

[11] H. Lodish, A. Berk, P. Matsudaira, C. Kaiser, M. Krieger, M. Scott, L. Zipursky, J. Darnell. *Molecular Cell Biology*, 5th edition. W.H. Freeman (2003).

[12] M. Minsky. *Finite and Infinite Machines*. Prentice Hall (1967).

[13] Gh. Păun, *Membrane Computing. An Introduction*. Springer-Verlag, Berlin (2002).

[14] Gh. Păun, G. Rozenberg, A. Salomaa (Eds.). *The Oxford Handbook of Membrane Computing*. Oxford University Press (2010).

[15] G. Rozenberg, A. Salomaa. *The Mathematical Theory of L Systems*. Academic Press (1980).

[16] A. Salomaa. *Formal Languages*. Academic Press (1973).

[17] R. Schroeppel. *A Two Counter Machine Cannot Calculate* 2^N. Massachusetts Institute of Technology, Artificial Intelligence Memo **257** (1972).

[18] A.M. Turing. On Computable Numbers, With an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*. **42**, 230–265 (1937).