Membrane Computing as a Framework for Modeling Economic Processes

Gheorghe PĂUN
Institute of Mathematics of the Romanian Academy
PO Box 1-764, 014700 Bucureşti, Romania, and
Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla
Avda. Reina Mercedes s/n, 41012 Sevilla, Spain
gpaun@us.es

Radu A. PĂUN
University of Maryland, Department of Economics
3105 Tydings Hall
College Park, Maryland 20742-7211 USA
paun@econ.umd.edu

Abstract. This paper is a first step towards a systematic evaluation of the possibilities to use membrane computing as a modeling framework for economics.

Membrane computing is a branch of natural computing, aiming to abstract computability models from the structure and functioning of living cells and from the way cells are organized in higher order structures (tissues, organs, etc.). The obtained models, called P systems, are cell-like distributed and parallel computability devices, based on multiset processing, rather flexible and versatile, already used in several applications in biology, medicine, computer science, and linguistics.

Here we start by a simple interpretation of several basic ingredients of membrane computing in economic terms, thus proving that the (conceptual, formal, graphical) language of membrane computing can be useful for economics. We then introduce a class of P systems with *bi-rules*, which capture features of economic processes which cannot be covered by the "standard" P systems.

Some simple illustrations of the versatility of this formalism are given.

1 Introduction

Mathematical modeling in economics has a long and glorious history, with computer science based techniques also much used (as only one example, see [8]).

Most of the resulting models are of a numerical type, in many cases based on continuous mathematics (e.g., systems of differential equations), but there also are several discrete models, as well as a few "purely" symbolic models rooted in theoretical computer science. We mention here the finite automata used in [14] and the Chomsky grammars of various types used in [16] in modeling a series economic processes in a framework related to the action systems from [15]. In the last decade, agent-based computational economics has emerged as a promising and much investigated direction of research, the continuous growth in computing power leading to evolved simulations of more and more complex economic models. We only mention the book [9], and we refer to the web page from `http://www.econ.iastate.edu/tesfatsi/ace.htm` for up-to-date information in this field.

The present paper continues and complements these lines of research, proposing *membrane computing* (MC) as a framework for modeling economic processes of (almost) any type.

Membrane computing is a bio-inspired branch of natural computing, abstracting computing models from the structure and functioning of living cells and from the organization of cells in tissues or other higher order structures. A short introduction to MC will be provided in the next section. The investigations were initiated in 1998, when the report version of [17] was released, and the field has so rapidly grown that in October 2003 MC was considered by Thomson Institute for Scientific Information (ISI) as "Emergent Research Front in Computer Science" (see `http://esi-topics.com`), with the seminal paper being nominated as "fast breaking paper". Details on MC can be found in [18], with a comprehensive bibliography and recent developments available on web at `http://psystems.disco.unimib.it`.

In what follows, we will mainly consider the cell-like membrane systems (also called P systems). In short, such a system consists of a hierarchical arrangement of membranes (understood as three dimensional vesicles), in the compartments of which we have multisets (sets with multiplicities of their elements) which evolve by means of local evolution rules. The objects can also pass through membranes, in certain conditions, thus making the compartments cooperate, therefore leading to a distributed (computing) device, with compartments evolving in parallel. There are many variants of P systems, either biologically or mathematically motivated. In MC, such devices are considered as computing models, and thus their computing power (compared with Turing machines and their restrictions) and computing efficiency (the possibility of using the inherent parallelism of P systems for solving computationally hard problems) are of a central interest. However, in biological applications in general (when modeling the "life" of any cell-like system) the evolution itself of a P system is of interest, not its "output", as provided in the end of a halting computation. This will also be the case in what follows, when we propose P systems as a framework for economic modeling.

The generality and flexibility of MC techniques are rather high. The membrane structure can be represented by an Euler-Venn diagram (of a restricted type – see below), the objects can be any entity relevant for the study at hand, the concept of a multiset can cover a wide range of real situations (especially in such areas as biology and economics, where the multiplicity matters and is expressed by natural numbers), the evolution rules can be written in the membrane structure's compartments together with the multisets of objects they make evolve.

This generality ensures the possibility to use MC as a modeling framework (in biology, economics, linguistics, etc), and this can be done at various levels; the case of applications in biology (see [6]) is illustrative – and suggestive also for economics.

In many cases, what is actually used is the MC *language*, having in mind three dimensions of this aspect: (i) the long list of concepts either newly introduced, or related in a new manner in this area, (ii) the mathematical formalism of MC, and (iii) the graphical language, the way to represent membranes, cell-like structures, tissue-like structures. We will illustrate each of these three points in Section 3, but we mention here an important aspect. Many ingredients are, in a great extent, known (Euler-Venn diagrams, with labels assigned to membranes, with multisets – not sets! – of objects placed inside, with arrows describing the communication channels in the case of tissue-like and neural-like systems), but there is something essentially new: together with the membranes and the objects, also the system's evolution rules are written in compartments or near membranes, so that not only the system's state is displayed, but also its "evolution engine". Also, the localization is apparent, both for objects and rules.

However, this level of application/usefulness is only a preliminary, superficial one. The next level is to use the tools, techniques, and results of MC, either to solve problems already stated (by biologists, but the same is true for economists) in other terms and framework, or to formulate (and then answer) new questions. In general, new tools can suggest new problems, which could either not be formulated in a previous framework (in plain language, as it is the case in biology, however specialized the specific jargon is, or using other tools, such as differential equations), or have no chance to be solved in the previous framework.

In biology, problems of the first type (already examined by biologists, mainly experimentally) concern, for instance, correlations of processes, the presence/absence of certain chemicals, their multiplicity (concentration, population) in a given compartment, their interaction, while problems of the second type are topics related to trajectories of bio-systems when modeled as dynamic systems (e.g., a sequence of configurations can be finite or infinite, while in the latter case it can be periodic, ultimately periodic, almost periodic, quasi-periodic, etc., notions which are not yet present in the index of notions of biology books). Which would be the economics questions to address remains to be investigated, but, e.g., the study of economic cycles is a classic (and hard) problem, which can find a natural framework to be studied in MC.

Indeed, the answer to questions formulated about a given model can be looked for analytically, in mathematical terms, but this is not always possible (as we will point out later, because the "computing cell" is a powerful computing device, in many cases equal in power to Turing machines, no non-trivial question about it can be answered algorithmically; there is no unique algorithm able to answer a given question for all P systems of a given class). Thus, we either have to study each model separately, by ad hoc methods, or have to be satisfied with statistics about models evolution, obtained by running experiments on computer. For biology there are many available programs which can simulate various types of P systems, thus able to provide insights on the evolution of biological processes modeled by these P systems; it is a research topic whether these programs can be also used for economics, whether they can be adapted, or new programs should be written.

There are several advantages of using MC (compartmental, parallel multiset processing) as a modeling framework (e.g., in comparison with standard models based on differential equations). Among these advantages we mention: the transparency (there is no "hidden" semantics, everything is easily understandable, in terms of "chemical reactions" which handle symbol-objects), inherent modularity, easy extendability (any number of membranes and/or evolution rules can be added to a P system without essentially changing its type, its simulating program), direct programmability (P systems are algorithmic models, and there are programming languages directly dealing with rewriting-like instructions, such as CLIPS, Maude, etc.), the possibility to handle cases where certain objects appear in a reduced number of copies (in the case of large populations of objects, passing to continuous models can be both possible and adequate, but this is not the case when we deal with small populations, and when the discrete nature of the process cannot be correctly approximated/handled by continuous tools). In turn, some important features of continuous models are preserved, and notable examples are those of non-linearity and of emergent global behavior, not directly and not linearly occurring out of local behaviors.

We will further discuss and illustrate some of these aspects in the subsequent sections, after briefly presenting some basic elements of MC.

We close this section by pointing out that the possibility of using MC in modeling economic processes has already been proposed by various authors, especially from Poland; the bibliography below indicates only some papers in English, [3], [12], [13], but further titles in Polish can be found at MC website. However, our approach is much different from that of the mentioned papers: we start from the existing formalism of MC, trying to interpret it in terms of economics, then we introduce a slight modification (in order to cover the "long-distance" communication of economic systems), aiming to model complex economic systems, not only particular processes/phenomena.

2 A Glimpse to Membrane Computing

The following short introduction to MC is meant to let the economist reader have a preliminary image of what a P system is, with only a short list of features mentioned; as already mentioned, further information can be found in the monograph [18] and at MC website (in particular, applications can be found in [6]).

The starting point is the living cell, where membranes play an essential role; the membranes both delimit "protected reactors", where specific reactions take place, and also contain proteins which both control reactions taking place in their neighborhood and make possible the passage of chemicals across membranes, in a highly selective manner. Here we only keep the role of membranes as compartment separators, with filtering properties. Chemicals in a cell swim in water, hence the natural data structure to describe a solution is a *multiset*, a set with multiplicities of its elements (we are not concerned with the structure of chemicals, they are interpreted as atomic – this is why we identify them with symbols from a given alphabet). The chemicals evolve through reactions, which are of the standard form in chemistry: $aab \rightarrow cd$ means that two copies of a react with one copy of b, these objects are consumed, and as a result of the reaction we get one copy of c and one of d.

Thus, inspired from the cell structure and functioning, the basic elements of a *membrane system* (in the literature called *P system*) are (1) the *membrane structure* and the sets of (2) *evolution rules* which process (3) *multisets* of (4) *objects* placed in the compartments of the membrane structure.

A membrane structure is a hierarchically arranged set of membranes. A suggestive representation is as in Figure 1. We distinguish the external membrane (corresponding to the plasma membrane and usually called the *skin* membrane) and several internal membranes (corresponding to the membranes present in a cell, around the nucleus, in Golgi apparatus, vesicles, etc.); a membrane without any other membranes inside it is said to be *elementary.* Each membrane determines a compartment, also called *region*, the space delimited from above by it and from below by the membranes placed directly inside, if any. The membrane–region correspondence is one-to-one, and so we identify by the same label a membrane and its associated region.

Clearly, a membrane structure can be represented by a rooted (unordered) tree, with the

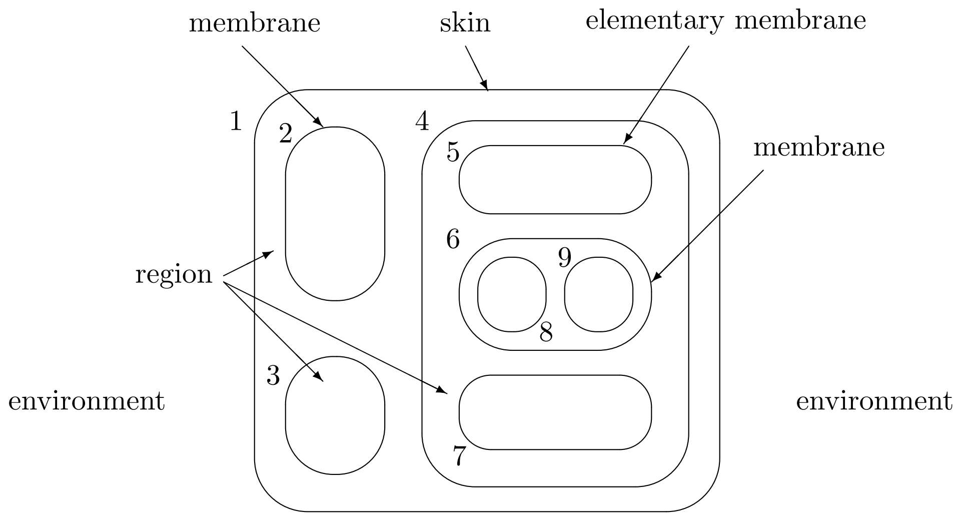

Figure 1: A membrane structure.

skin in the root. For instance, the membrane structure from Figure 1 corresponds to the tree from Figure 2.

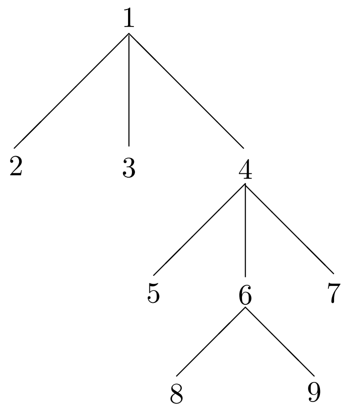

Figure 2: The tree representing the membrane structure from Figure 1.

In symbols, a suggestive representation of a membrane structure is in the form of an expression of correctly matching labeled brackets; the expression corresponding to the previous membrane structure is

$$[_1[_2\ ]_2[_3\ ]_3[_4[_5\ ]_5[_6[_8\ ]_8[_9\ ]_9]_6[_7\ ]_7]_4]_1. \tag{1}$$

Because the sibling membranes placed in the same directly upper membrane have no ordering, the tree associated with a membrane structure is not ordered, and hence the neighboring pairs of brackets from an expression as that in (1) can be interchanged (hence the same membrane structure can be represented by all bracket expressions obtained by such sibling pairs interchanged).

Each region contains a multiset[1] of objects, described by symbols from a given alphabet.

[1]Formally, a multiset over a given finite set X is a mapping $M : X \longrightarrow \mathbf{N}$, with $\mathbf{N}$ denoting the set of natural

Thus, the multisets are described by strings over the given alphabet: the number of occurrences of a symbol a in a string w is the multiplicity of object a in the multiset described by w; thus, all permutations of a string describe the same multiset. For instance, a^3bd^2 (and all its permutations) describes the multiset over $X = \{a, b, c, d\}$ where a occurs three times, b occurs once, c does not occur, and d occurs twice.

The objects evolve by means of evolution rules, which are also localized, associated with regions of the membrane structure. The typical form of such a rule is $cd \to (a, here)(b, out)(b, in)$, with the following meaning: one copy of object c and one copy of object d react and the reaction produces one copy of a and two copies of b; the newly produced copy of a remains in the same region (indication *here*), one of the copies of b exits the compartment, going to the surrounding region (indication *out*) and the other enters one of the directly inner membranes (indication *in*). We say that objects a, b, b are *communicated* as indicated by the commands associated with them in the right hand side member of the rule. When an object exits a membrane, it will go to the surrounding compartment; for the skin membrane this is the environment, hence the object is “lost”, it never comes back into the system. If no inner membrane exists (that is, the rule is associated with an elementary membrane), then the indication *in* cannot be followed, and the rule cannot be applied.

A rule as above, with several objects in its left hand side member, is said to be *cooperative*; a particular case is that of *catalytic* rules, of the form $ca \to cx$, where a is an object and c is a catalyst, which appears only in such rules, and never changes. A rule of the form $a \to x$, where a is an object, is called *non-cooperative.*

The rules associated with a compartment are applied to the objects from that compartment, in a *maximally parallel way*: a combination of rules is chosen in each compartment such that no further rule can be applied to the objects from that compartment (we assign objects to rules, until no further assignment is possible). The used objects are “consumed”, the newly produced objects are placed in the compartments of the membrane structure according to the communication commands assigned to them. The rules to be used and the objects to evolve are chosen in a non-deterministic manner. In turn, all compartments of the system evolve at the same time, synchronously (a common clock is assumed for all membranes). Thus, we have two layers of parallelism, one at compartment level, and one at the level of the whole “cell”.[2]

A membrane structure and the multisets of objects in its compartments identify a *configuration* of a P system. As suggested above, by a non-deterministic maximally parallel use of rules, we pass to another configuration, and such a step is called a *transition.* A sequence of transitions constitutes a *computation.* Results can be associated to halting computations (those reaching a configuration where no further rule can be applied), for instance, the number of objects present in a specific region of the halting configuration. This way, we get a computing device. When using a P system to model a biological – or economic – process/system, we are no longer interested in halting computations and computability results, but in the evolution itself of the system, thus adopting a dynamic system approach. Nevertheless, because of the non-deterministic way the rules apply, starting from an initial configuration we can reach several successful computations, hence several results. Thus, a P system *computes* (one also uses to say

numbers. For each $a \in X$, $M(A)$ describes the multiplicity of object a in multiset M. We consider here only finite support, finite multiplicity multisets.

[2]Note that evolution rules are stated in terms of *names of objects*, they are “multiset rewriting rules”, while their application/execution is done using *copies of objects.* For instance, the rule $aa \to bc$ is used three times in a compartment containing 7 *copies* of object a.

generates) a set of numbers.

Of course, the previous way of using the rules from a P system region reminds of the non-determinism and the (partial) parallelism within cell compartments, with the mentioning that the maximality of parallelism is mathematically oriented; when using P systems to model biological or economic systems/processes, the parallelism should be replaced with more realistic features (e.g., reaction rates, probabilities, partial parallelism).

We do not give here a formal definition of a P system, but we only mention that when presenting a P system we have to specify: the alphabet of objects (a usual finite non-empty alphabet of abstract symbols identifying the objects), the membrane structure (represented by a string of labeled matching brackets), the multisets of objects present in each region of the system (represented by strings of symbol-objects), the sets of evolution rules associated with each region, as well as the indication about the way the output is defined.

Many modifications/extensions of the very basic model sketched above are discussed in the literature. For instance, we can consider the biological process of two or more objects passing through a protein channel simultaneously, the so-called *symport* and *antiport* (see [1] for details). Symport refers to the transport where two (or more) molecules pass together through a membrane in the same direction, antiport refers to the transport where two (or more) molecules pass through a membrane simultaneously, but in opposite directions, while the case when a molecule does not need a "partner" for a passage is referred to as uniport.

In terms of P systems, we can consider object processing rules of the following forms: a symport rule (associated with a membrane i) is of the form (ab, in) or (ab, out), stating that objects a and b enter, respectively exit, together membrane i, while an antiport rule is of the form $(a, out; b, in)$, stating that, simultaneously, a exits and b enters membrane i; uniport corresponds to a particular case of symport rules, of the form (a, in) or (a, out). An obvious generalization is to consider symport rules (x, in) or (x, out) and antiport rules $(x, out; y, in)$ with x, y arbitrary multisets of objects.

Note that these rules do not change the objects, they only change their places in the system. An interesting combination of rewriting rules and symport/antiport is that of systems where the evolution of objects is based on rewriting rules (hence no target indication *here, out, in* appears in the right hand side of multiset rewriting rules) and where communication across membranes is based on symport/antiport rules (such P systems were introduced in [4] under the name of evolution-communication P systems).

There also are several ways of controlling the way rules are used, thus diminishing the non-determinism of the system evolution: priorities, promoters, inhibitors, etc. Furthermore, there are rules for changing the membrane structure (not only the objects); for instance, there are rules for dissolving, creating, or dividing membranes, for modeling other biological processes such as exocytosis and endocytosis, vesicle transport of chemicals, and so on. Some of them will be mentioned in Section 3.

We conclude this section with some remarks on P systems' computing power. Most classes of P systems were proven to be computationally universal, equal in power to Turing machines (more exactly, they characterize the family of Turing computable sets of natural numbers). This happens for rather restricted classes of systems, i.e., for restricted types of rules and a small number of membranes. For instance, P systems using only catalytic multiset rewriting rules, with only two catalysts are Turing universal (see [10]), and this is also true for symport/antiport P systems with minimal symport and antiport (always moving together two objects; see [2] and the references therein). This is a pleasant computer science result, but not so pleasant from a modeling point of view: being so powerful computing devices, P systems have many undecidable properties. For instance, we cannot find an algorithm to solve the reachability problem (given an initial configuration of the system and another specified configuration, find whether or not there is any possible evolution of the system from the initial configuration to the specified one). The interest in such a problem in (biology and) economics is obvious: starting from a "healthy" initial configuration, do we ever reach a 'bad" configuration, with specified properties (e.g., overpopulated by a specified "bad" object)? The same is true for other properties, such as halting/deadlock, existence of cycles, etc.

This does not mean that such problems cannot be answered by ad hoc means for specific systems, but only that *there is no uniform algorithmic manner to do it*, there is no unique algorithm/computer program which can answer such a question *for all systems* from a given class. In particular, especially when the analytic approach is too difficult, we can address such problems by means of computer experiments, e.g., answering questions of the following form: does a given system (which can be any one from a given class) evolve in a particular way *during first 10,000 steps*? Especially for systems where stochastic parameters are involved, such experiments are obviously useful (even necessary, when the model is complex, because of the non-linear, emergent global behavior).

3 Interpreting Some MC Ingredients in Terms of Economics

The aim of this section is to prove that many ingredients of MC can be easily interpreted in terms of economics, and thus the biological paradigm/metaphor can be of interest for modeling economic processes. Besides the notions mentioned in the previous section, we will also briefly consider further MC notions, but the discussion is rather preliminary, a more systematic and detailed "MC – economics dictionary" remains to be elaborated.

- **Object:** any unitary item which is produced, transferred, consumed, ordered, planned in an economic systems; objects can be commodities of any kind, and parts of commodities if they explicitly appear in the production process, monetary units, labor "units", and can also be the need for any one of these, or the order placed to obtain them. For instance, if a is an object (for instance, a laptop), the need of having one can be denoted by $\bar{a}$, the fact that a copy of a was ordered (to a shop by a customer, or to a company by a shop) can be denoted by $\hat{a}$, while the fact that one a was planned to be produced in a company, hence scheduled to subunits of the company, can be denoted by $\tilde{a}$. Units of energy can also be considered as objects. Thus, the objects can have a physical existence, or they can be "semantic marks", denoting non-existing stages of physical objects, but which are to be handled in an economic process.

- **Membrane:** any economic entity handling objects, starting from individuals, working places in a company, and ending with a whole economy. Each such entity should have a clear individuality, should be delimited with respect to the upper level entities. Stated otherwise, each membrane should delimit an inside from an outside, and this separation should have a meaning in the organization and functioning of the investigated economic system. Associated with each membrane we have objects and object-processing rules; further membranes may be placed in a membrane, thus defining relevant sub-entities in the entity represented by the membrane.

For instance, a customer can be considered as an elementary membrane, containing a multiset of monetary units (used to buy satisfactors–objects for personal needs), of labor units (used for getting monetary units), a general state-object, which can generate needs for various objects. Rules to handle these objects will also be associated with the customer membrane. When modeling a company, we have to consider various other membranes inside it, representing departments, sections, and sub-sections, with objects which can be physical objects (in deliverable form, or parts, to be used in the production process), orders for objects, planned/scheduled objects, monetary units, and so on.

- **Membrane structure:** most economic entities contain relevant sub-entities, and all of them are part of a higher entity, ending with the whole economy. We do not go further with the framework of hierarchical membrane structures, noting that membranes placed at a given level of the structure (of the respective tree) can form any virtual graph, e.g., in terms of communication/collaboration; that is, also horizontal, non-hierarchical structures can be handled, but not being explicit in the membrane structure. When representing the hierarchical structure of, for instance, a production unit with several sections, and each section containing several work places, in the regions corresponding to an upper membrane we will only have objects and rules specific to that level. Continuing this example, we can consider company management to be "outside" of the production unit, dealing with specific objects (such as ordered objects), scheduling parts of ordered objects to the lower level membranes, then delivering produced objects to the outside membranes which have ordered them. Therefore, orders for complex objects which are produced only at whole company level should not be handled by sections.

  It is clear in the present setup that membranes are mainly delimiters, and not so much filtering agents for objects, like in biology; this is a consequence of the fact that most membranes which we can consider in economics are virtual membranes, not physical ones.

- **Environment:** the space outside the skin membrane; it can be considered empty, with the possibility to only send objects there (e.g., waste), or it can be considered populated with objects which can be brought into the system; in the latter case, it is natural to assume that the environment is inexhaustible, an endless source of various objects – raw materials, energy, needs/orders for certain objects. Trade operations may be captured in this way.

- **Multiset rewriting rules:** they can model the object production operations, the assembly de-assembly operations, for any kind of objects.

  Examples: The fact that customer i, being in state c and having n monetary units u, has/produces the need for a laptop, $\bar{l}$, and then orders a laptop, $\hat{l}$, to a company j with an advance payment of n' monetary units u can be represented by the rules

  $$[_i cu^n \to c'u^n\bar{l}]_i, \quad [_i c'u^n\bar{l} \to c''u^{n-n'}\langle\hat{l}, c_j\rangle]_i,$$

  where we also see customer's state changes by using these rules.

  Then, assembling two processors p, one keyboard k, a screen s, four memory units m, a hard disk h, and two drivers d, by consuming 6 labor hours, w, and thus producing a laptop (well, a toy one...) l as well as 8 waste units g can be expressed as

  $$[_x p^2ksm^4d^2w^6 \to lg^8]_x,$$

where production unit, x, is indicated as well. The waste g will be expelled into environment or collected in a specialized membrane, the object l will cancel the scheduled $\tilde{l}$ and will be delivered to customer, thus also cancelling the order $\hat{l}$ and then the need $\bar{l}$, in exchange of n'' further monetary units.

The last operations imply actions performed in two separate membranes, and such operations raise some problems, which we will discuss later, so that we do not present formal rules here.

- **Symport rules:** they can represent inputs into a membrane or ejections out of a membrane (maybe into the environment) of certain objects, without anything in exchange. Waste storage or any polluting action can be represented by rules of the form (x, out), while free entrance of resource can be represented by symport rules of the form (x, in).

- **Antiport rules:** they are more interesting than symport rules, as they can represent trading operations. For instance,

$$(\langle \hat{l}, c_j \rangle u^{n''}, out; l, in),$$

  associated with the membrane representing customer i describes the fact that order $\hat{l}$ to company c_j is satisfied, in exchange of n'' monetary units. (Then, we can have $[_i l\bar{l}c'' \to lc]_i$ represent the fact that the customer has his/her need satisfied, hence returning to the initial state c.)

  Thus, roughly speaking, *production operations can be represented by multiset rewriting rules, while exchange operations can be represented by antiport rules*, in total, having an evolution-communication P system as that in [4]. Of course, many economic features are missing here; some of them will be covered by ingredients mentioned below, while others need extensions of standard features of MC.

- **Promoter/inhibitor:** certain operations can be performed only if specific objects are present/absent from the respective membrane. The fact that a rule $u \to v$ takes place only when the objects of a multiset z are present is written in the form $u \to v|_z$, while the inhibiting case (if any object from z is present, then the rule is not applied) is written as $u \to v|_{\neg z}$. For instance, the previous rule $(\langle \hat{l}, c_j \rangle u^{n''}, out; l, in)$ should be used only if the customer is still in state c'' – changing the state can mean that (s)he has changed his mind, (s)he still needs a laptop, but, at the expense of losing the advanced money, (s)he no longer purchases it. Later, if the state will return to c'', delivery of object l may be accomplished, provided that company c_j agrees to do it then. Inhibiting conditions can be easily illustrated by bank operations, where loans can be denied by the occurrence of certain objects (previous loans, bad credit history, etc.).

- **Priority relations:** even if equally applicable in what concerns the existing objects, one rule can have a higher priority than another, and is therefore applied. Customers may have preferences over the use of their money, companies may have preferences when ordering parts or raw materials. Such preferences can be represented by partial order relations over the set of rules associated with a membrane, with two possible interpretations: under the strong interpretation, if a rule is applied, then no rule of lower priority can be applied further, irrespective of whether objects remain and are not handled by the applied rule;

under the weak interpretation, if objects remain and cannot be further processed by the higher priority rule, then a rule of immediately lower priority can use them. For instance, consider the following case

$$[_0[_i\ a^8,\ r_1 : (a^3, out) > r_2 : (a, out)\ ]_i]_0.$$

Under the strong interpretation of priority, we have to use two times rule r_1, leaving inside membrane i two copies of a:

$$[_0[_i\ a^8,\ r_1 : (a^3, out) > r_2 : (a, out)\ ]_i]_0 \Longrightarrow [_0[_i\ a^2,\ r_1 : (a^3, out) > r_2 : (a, out)\ ]_i\ a^6]_0.$$

Under the weak interpretation, the competition is for objects, r_2 is used for the two copies of a not used by rule r_1, hence we get

$$[_0[_i\ a^8,\ r_1 : (a^3, out) > r_2 : (a, out)\ ]_i]_0 \Longrightarrow [_0[_i\ \ r_1 : (a^3, out) > r_2 : (a, out)\ ]_i\ a^8]_0.$$

We can see the difference between the two interpretations as the need to use an additional facility, which is not expressed by the rules. For instance, if the rules r_1, r_2 represent two different companies and object transportation across membrane i is performed by a vehicle of a third company which can be rent by only one customer, then we have the strong interpretation case, as only one of the companies r_1, r_2 can rent the vehicle, and this one is r_1.

- **Membrane creation:** in the presence of a given object, a membrane with a given label can be created; the rule is written in the form $a \to [_i v]_i$ (the new membrane, labeled i, contains the objects specified by the multiset v). The rule can also be localized, allowed to be used only in the membrane with which it is associated (this can be made explicit by writing $[_j a \to [_i v]_i]_j$). In economic terms, this corresponds to creating a new entity, with the restriction that the newly created entity is an elementary one; however, further membranes/entitities can be created inside it, or it can "engulf" other entities, e.g., by phagocytosis – see below.

- **Membrane dissolution:** this is the opposite of membrane creation, and represents the operation of cancelling an economic entity, with its contents (objects and inner membranes) being released in the surrounding membrane, and with the associated evolution rules being removed. For example, when a company section stops functioning, its activity is terminated but the employees and the inventory become a direct part of the company, which can redistribute them among the remaining sections.

- **Membrane separation:** this corresponds to the split of an economic unit into two units, each of them taking part of initial unit's content. Formally, the separation of membrane i's content into two membranes, j and k, the first one taking all objects from a set M and the latter taking the remaining objects from $O - M$, is written in the form $[_i a]_i \to [_j M]_j [_k O - M]_k$ (the object a controlling the operation is lost, but a possibility is to transform it into an object b to appear in one of the membranes j or k).

- **Membrane merging:** opposite to separation, with contents of two membranes put together in a unique membrane. The economic interpretation is obvious.

- **Phagocytosis:** this directly corresponds to the so-called integration used in economics, when a company i gains control over a (smaller) company j. The most common form is that of vertical integration, when the two companies are engaged in a customer-supplier relationship, and one acquires the other to avoid being exposed to the opportunistic behavior of its partner. This can be written in the form $[_i a]_i [_j b]_j \rightarrow [_i a' [_j b']_j]_i$ (the objects a and b control the operation and may be transformed during phagocytosis).

The previous list can be continued, and we can also consider more elaborated notions from MC. Here is just an example: **the adjacency graph** defined by a set of rewriting rules. This notion has been recently used as a proof technique in [11], but it can also be used as a way to look for cycles in a production process, with a possible deadlock appearing because of its cyclical nature. Formally, for a set of rules of the form $u \rightarrow v$ placed in a membrane i, we consider the graph of all nodes (a, i), with edges $((a, i), (b, i))$ for all a appearing in u and all b appearing in v. Graphs G_i associated with regions can then be linked by means of edges corresponding to communication rules among membranes, thus making possible the search for cycles at the whole system level. We skip the technical details as not all of them can be easily specified.

4 Two Simple Illustrations

We illustrate here the possibility of formalizing economic processes in terms of MC and operations with membranes, by considering two processes at company level, the *vertical integration* and the *spin-off.*

Vertical integration is the process by which a company gains control over (part of) one of its suppliers or customers, and its main goal is for the company not to depend too much on the trading partner, and therefore limit the likelihood of opportunistic behavior and hold-up. One obvious factor leading to vertical integration is the lack of alternative trading partners. The case of a company integrating one of its suppliers is called backward vertical integration, while forward vertical integration stands for the case of a company gaining control over one of its customers.

A schematic representation in terms of a membrane structure can be found in Figure 3, where C_1, C_2 are the two companies, each containing a production department (Pr_i), a selling department (S_i), and a management department (M_i); in the first company we have also mentioned the purchasing department (Pu_1), the one which buys through S_2 part of the products manufactured by Pr_2. By gaining control over C_2, company C_1 integrates Pr_2 as a sub-department of Pr_1, and also S_2 into S_1 in order to continue to sell (to the environment, in this case) the other products of C_2 which are not consumed by Pr_1; the management department of C_2 is dissolved. The result of this process is indicated in the bottom part of the figure.

This process can be decomposed into a sequence of biological-like operations, namely, of *phagocytosis, dissolution,* and *exocytosis.* The respective rules are the following:

$$\begin{array}{lll}
(1) & [_{C_1}\ ]_{C_1}[_{C_2}\ ]_{C_2} \rightarrow [_{C_1}[_{C_2}\ ]_{C_2}]_{C_1} & \text{(phagocytosis)} \\
(2) & [_{C_1}[_{C_2}\ ]_{C_2}]_{C_1} \rightarrow [_{C_1}\ ]_{C_1} & \text{(dissolution)} \\
(3) & [_{Pr_1}\ ]_{Pr_1}[_{Pr_2}\ ]_{Pr_2} \rightarrow [_{Pr_1}[_{Pr_2}\ ]_{Pr_2}]_{Pr_1} & \text{(phagocytosis)} \\
 & [_{S_1}\ ]_{S_1}[_{S_2}\ ]_{S_2} \rightarrow [_{S_1}[_{S_2}\ ]_{S_2}]_{S_1} & \text{(phagocytosis)} \\
(4) & [_{C_1}[_{M_2}\ ]_{M_2}]_{C_1} \rightarrow [_{C_1}\ ]_{C_1}[_{M_2}\ ]_{M_2} & \text{(exocytosis)}
\end{array}$$

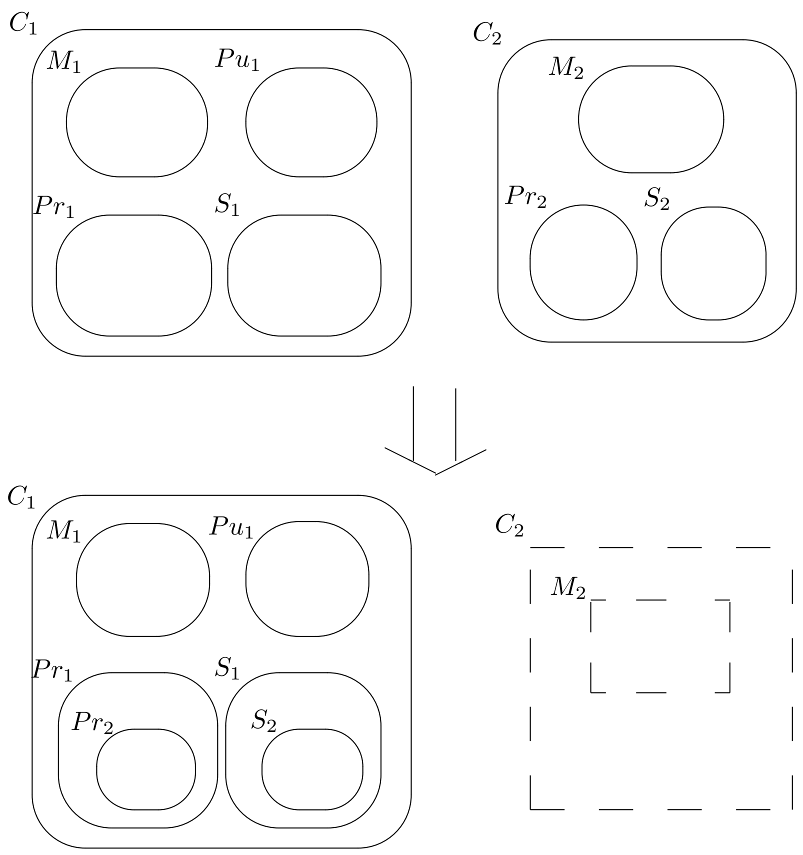

Figure 3: (Backward) Vertical Integration

The process can be described as follows:

Step	Configuration	Rules
Initial	$[_{C_1}[_{Pr_1}\ ]_{Pr_1}[_{S_1}\ ]_{S_1}[_{M_1}\ ]_{M_1}[_{Pu_1}\ ]_{Pu_1}]_{C_1}[_{C_2}[_{Pr_2}\ ]_{Pr_2}[_{S_2}\ ]_{S_2}[_{M_2}\ ]_{M_2}]_{C_2}$	
Buy	$[_{C_1}[_{Pr_1}\ ]_{Pr_1}[_{S_1}\ ]_{S_1}[_{M_1}\ ]_{M_1}[_{Pu_1}\ ]_{Pu_1}[_{C_2}[_{Pr_2}\ ]_{Pr_2}[_{S_2}\ ]_{S_2}[_{M_2}\ ]_{M_2}]_{C_2}]_{C_1}$	(1)
Dissolve	$[_{C_1}[_{Pr_1}\ ]_{Pr_1}[_{S_1}\ ]_{S_1}[_{M_1}\ ]_{M_1}[_{Pu_1}\ ]_{Pu_1}[_{Pr_2}\ ]_{Pr_2}[_{S_2}\ ]_{S_2}[_{M_2}\ ]_{M_2}]_{C_1}$	(2)
Reorganize	$[_{C_1}[_{Pr_1}[_{Pr_2}\ ]_{Pr_2}]_{Pr_1}[_{S_1}[_{S_2}\ ]_{S_2}]_{S_1}[_{M_1}\ ]_{M_1}[_{Pu_1}\ ]_{Pu_1}]_{C_1}[_{M_2}\ ]_{M_2}$	(3), (4)

One observes that in each of the first two steps only one rule is used: C_1 “phagocytates” C_2, then dissolves it; in step 3 one uses simultaneously three rules, two phagocytosis rules, for moving Pr_2 inside Pr_1 and for moving S_2 inside S_1, and one exocytosis rule, for expelling M_2 into the environment.

One can notice here the transparency of both the graphical representation and of bio-inspired operations, as well as the rewriting style of the rules, easy to transform into program instructions (e.g., written in CLIPS).

Similar representations can be produced for the process of spin-off, by which part of a company becomes an independent company, thus creating its functional departments (e.g., production, management, selling, purchasing) and separating from the mother-company. Figure 4 schematically illustrates such a case, where the production sub-department D_2 initiates the process of spin-off, leading to the creation of a new company, C_2, by "budding" new departments from the departments of C_1 (the departments M'', S'', Pu'' are parts of M, S, Pu, respectively, which leave C_1, where M', S', Pu' remain, together with D_1).

The process can be realized by means of the following rules:

$$\begin{array}{lll}
(1) & [_{Pr}[_{D_2}\ ]_{D_2}]_{Pr} \to [_{Pr}\ ]_{Pr}[_{Pr'''}\ ]_{Pr'''} & \text{(exocytosys)} \\
(2) & [_{M}\ ]_{M} \to [_{M'}\ ]_{M'}[_{M''}\ ]_{M''}|_{[_{Pr'''}]_{Pr'''}} & \text{(budding)} \\
 & [_{S}\ ]_{S} \to [_{S'}\ ]_{S'}[_{S''}\ ]_{S''}|_{[_{Pr'''}]_{Pr'''}} & \text{(budding)} \\
 & [_{Pu}\ ]_{Pu} \to [_{Pu'}\ ]_{Pu'}[_{Pu''}\ ]_{Pu''}|_{[_{Pr'''}]_{Pr'''}} & \text{(budding)} \\
(3) & [_{Pr'''}\ ]_{Pr'''} \to [_{Pr''}\ ]_{Pr''}[_{C_2'}\ ]_{C_2'} & \text{(budding)} \\
(4) & [_{C_2'}\ ]_{C_2'}[_{Pr''}\ ]_{Pr''} \to [_{C_2}[_{Pr''}\ ]_{Pr''}]_{C_2} & \text{(phagocytosis)} \\
 & [_{C_2'}\ ]_{C_2'}[_{M''}\ ]_{M''} \to [_{C_2}[_{M''}\ ]_{M''}]_{C_2} & \text{(phagocytosis)} \\
 & [_{C_2'}\ ]_{C_2'}[_{S''}\ ]_{S''} \to [_{C_2}[_{S''}\ ]_{S''}]_{C_2} & \text{(phagocytosis)} \\
 & [_{C_2'}\ ]_{C_2'}[_{Pu''}\ ]_{Pu''} \to [_{C_2}[_{Pu''}\ ]_{Pu''}]_{C_2} & \text{(phagocytosis)} \\
(5) & [_{C_0}[_{C_2}\ ]_{C_2}]_{C_0} \to [_{C_1}\ ]_{C_1}[_{C_2}\ ]_{C_2} & \text{(exocytosis)}
\end{array}$$

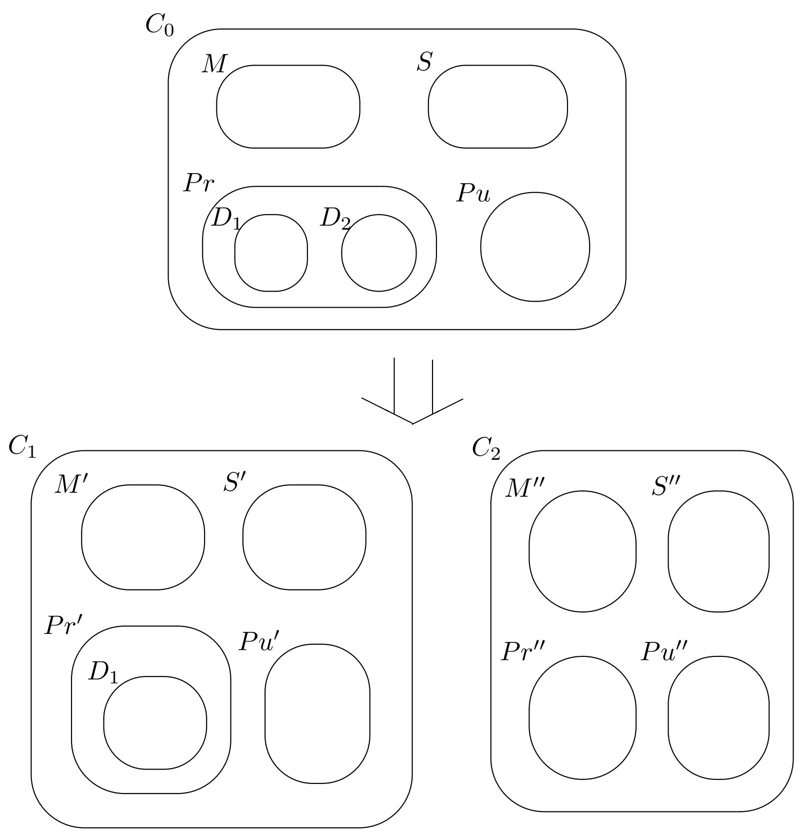

Figure 4: Spin-off

These rules have some additional features in comparison with the rules used for describing vertical integration. First, we have used here label changing; membranes participating into these operations can change their labels, and this is useful in controlling the sequencing of operations (sequencing of applying the rules). This is crucial in several steps. For instance, when separating D_2 from Pr, we create a membrane labeled Pr''', which is later transformed into a membrane labeled Pr'', after budding C_2'; this way, rule (3) can be used only once (if the label of the membrane remains unchanged, then the budding can be repeated indefinitely). Similarly, the label of the newly created membrane C_2' changes by using the rules from group (4), thus introducing the final name of the new company, C_2.

Another novelty is the fact that the rules from group (2) have "promoters", they can be used only in the presence of membrane $[_{Pr'''}\ ]_{Pr'''}$, which initiates the separation. If these rules are used without such a promoter, then they can be used in the first step of the process, at the same time with rule (1), thus making the separation a spontaneous process, not one initiated by the separation of D_2 from Pr.

Also the use of these rules needs some discussion. In the previous paragraphs it is understood that the rules from group (4) are used simultaneously, in parallel. This is a matter of convention. If we do not allow a membrane to phagocyte in the same time several other membranes, then we can use these rules sequentially, one in a separate step, but in this case we need to change the label of membrane C_1' from step to step, for instance, to C_1'', C_1''', C_1^{iv}, and only in the fourth step to C_1, thus precisely sequencing the four rules of group (4).

We leave to the reader the task of writing both these variants of rules and the steps of the process (the system configurations along the steps of using the specified rules).

5 Economic Aspects which Are Not Covered by MC

The previous list can be complemented by a similarly long list of MC ingredients which do not have good correspondents in economics (they are too biologically or too computer science oriented: membrane division, with the replication of the contents of the initial membrane; rules of the form $a \rightarrow aa$, making possible the production of exponentially many copies of a in a linear time – not to speak about the fact that the conservation law is simply ignored; reaction rates associated to rules, computed according to the populations of reactants; halting computations), as well as by a list of features of economic systems which do not have counterparts in existing MC literature. Especially these latter features are of interest, in order to adapt MC language and techniques to economics, that is why we consider some of them here.

- **Communication across several membranes:** in real-life economics it is possible for objects to pass across several membranes in a single step, which is not the case in MC (as it is not the case in biology either). For instance, in order to produce on object, a section of a company can order a part directly to another section of the same company, or even to another company, thus making necessary crossing of at least two membranes at once. Such long distance correlated events look rather common and important, so that we will pay a particular attention to them and we will incorporate them in what we call below P systems with *bi-rules*.

- **Performance criteria:** rules of a P system are chosen non-deterministically, with probabilities and reaction rates in the case of biological models, maybe controlled by priorities, promoters/inhibitors, etc., but not aiming at maximizing/minimizing numerical criteria. As a matter of fact, such criteria are almost absent in MC, the only exception being that of P systems with energy associated with rules, and where one can impose that the rules chosen to apply should have a minimal/maximal quantity of energy consumed. However, choosing multisets of rules in such a way to maximize/minimize an objective function leads to complexity complications, hence we do not further discuss this matter here.

- **Dynamical rules:** rules of P systems remain the same during the whole computation (with only the possibility to remove rules by membrane dissolution), while in economics rules continuously change, for instance, because of technological advance and market functioning (the structure of a laptop continuously changes, and the same happens with its price).

- **Psycho-social or political aspects:** rules of a P system are independent of non-mathematical factors, which is not the case in economics. Similarly to the other social sciences, economics is affected by individuals' (changing) behavior. Examples abound: a rumor can increase or decrease the price of a commodity, too many customers withdrawing money from a bank can create panic, political decisions can impose the use of certain rules instead of others (promoters/inhibitors can be used in this respect, but the problem is to produce them in various subsystems, maybe at different levels in the membrane structure, at the same time, and such broadcasting rules have not yet been considered in MC).

- **Environment evolution:** the P systems' environment contains no local rules, it participates to a computation, e.g., through symport/antiport rules, but only under the control of rules associated with the skin membrane of the system. This is not the case of economics environments, whatever the level of the discussion. This aspect can be easily covered by considering one more membrane around a given system, thus postponing the question of what happens outside this additional membrane, or by simply considering rules in the environment, possibly both rules for (some) objects' evolution, and for influencing a given system from outside (e.g., by orders for objects – a case which can correspond to export activities).

- **Probability of using a rule:** in biology (and related P systems) the reaction rates are in general proportional to the concentration of reactants, which is often different and in many cases somewhat reverse from what happens in economics. New objects appearing on the marked can attract more than old objects; objects which are available in small quantities can become collectible items and so more attractive for some rules; certain rules have probabilities assigned once forever, independent of the "concentrations of reactants", maybe even taking the form of priorities (personal ties between owners of two companies can enhance the collaboration among their companies, without making this visible in the production/exchange rules).

- **Cost of communication:** there is no such a thing in MC, but we have it in real-life economics, at least because all transactions (of any type of objects) are carried out through specific channels (e.g., carrier companies that link production companies among them and with shops). This can be captured by considering separate membranes for carriers, at the price of making the model more complicated, but it does not seem clear how to encode the price of using a rule in the rule itself, e.g., in a symport or an antiport rule (these rules only move objects, hence they cannot also change/remove objects).

The list can be continued, but we choose to stop here. Still, some of the previously considered features will be captured by the bi-rules we will introduce below, while other features remain to be examined.

6 P Systems with Bi-Rules

We start by proposing a way to capture correlation between actions taking place at distance in a membrane structure, and the idea is to consider pairs of multiset rewriting rules – we call them *bi-rules* – of the following form:

$$([_i u_1 \to v_1]_i, [_j u_2 \to v_2]_j),$$

where i, j are membranes and u_1, v_1, u_2, v_2 are non-empty multisets of objects from a given alphabet O of objects. No restriction is imposed on the labels i, j, so, in particular, they may be equal. Also, we allow one or both of i, j to be the environment (thus relating the inner evolution of the system to the evolution of the environment, in particular, allowing the evolution of objects placed in the environment).

The idea is that rule $u_1 \to v_1$ is applied in membrane i simultaneously with rule $u_2 \to v_2$ being applied in membrane j, and this is possible only if objects specified by u_1 and u_2 are present in the respective regions. (When $i = j$, the two rules are applied in the same region.)

Such a bi-rule has a very general meaning, and it can cover many economic operations.

For instance, a separate rule $u \to v$ to be applied in a membrane i independently of any other membrane can be paired with a rule $d \to d$, for a "dummy" object d, available in sufficiently many copies in any membrane of the system. If both u and v contain at least two objects (hence we can write $u = u'u'', v = v'v''$, with all u', u'', v', v'' non-empty) then we can even avoid using such dummy objects and we can use $([_i u' \to v']_i, [_i u'' \to v'']_i)$.

Then, an antiport rule $(u, out; v, in)$ associated with a membrane i can be simulated by the bi-rule $([_i u \to v]_i, [_j v \to u]_j)$, where j is the label of the immediately higher membrane with respect to i (its parent membrane). The equivalence of the antiport rule with the associated bi-rule is obvious.

Symport rules can also be simulated by bi-rules, provided that dummy objects are available: a rule (u, out) associated with a membrane i corresponds to $([_i u \to d]_i, [_j d \to u]_j)$, and a rule (u, in) corresponds to $([_i d \to u]_i, [_j u \to d]_j)$, with j being in both cases the parent membrane of i (when i is the skin membrane, j identifies the environment of the system).

A P system with bi-rules has the usual architecture: alphabet of objects, membrane structure, multisets present initially in the compartments of the membrane structure, as well as a finite set of bi-rules (the rules specify the membranes where they are used, hence the set of rules is given for the whole system); in addition, we need objects in environment. For generality, we consider that we have both objects with a finite multiplicity and objects with unlimited abundance.

For the sake of completeness, we formally give here the definition of P systems with bi-rules: such a device is a construct of the form

$$\Pi = (O, \mu, w_1, \dots, w_m, w_0, E, R),$$

where O is the alphabet of objects, μ is the membrane structure (of degree m, i.e., composed of m membranes), $w_1, \dots, w_m$ are strings over O specifying the multisets of objects present in the m regions of μ at the beginning of the system's evolution, w_0 is the multiset of objects present in environment at the beginning of evolution, $E \subseteq O$ is the set of objects which appear in environment in arbitrarily many copies, and R is a finite set of bi-rules with multisets over O and membrane labels from $\{0, 1, 2, \ldots, m\}$.

The rules in R are applied in the non-deterministic maximally parallel way: the objects to evolve and the rules by which they evolve are chosen non-deterministically, but in such a way that a maximal number of rules is used (no rule can be applied to the remaining objects). This way, transitions among configurations of the systems are obtained; any sequence of transitions starting from the initial configuration (that defined by $(w_1, w_2, \ldots, w_m, w_0)$) is called evolution of Π.

On the set R we can also consider a priority relation, in the form of a partial order relation, with the two possible interpretations, the strong and the weak one, as well as promoters or inhibitors, thus decreasing the system's degree of non-determinism. Furthermore, probabilities can be associated with rules, given in advance, when specifying the initial configuration of the system. Such additional features can be of interest when modeling real economic systems, but we do not further consider them here.

7 A Theoretical Computer Science Observation

We have mentioned above that the decidability properties of a class of P systems depends on the computing power of those systems, and this is why it is of interest to briefly examine the power of P systems with bi-rules. From the previous discussion on the possibility of simulating usual rules by bi-rules, it easily follows the computational universality of our systems.

Let us be more rigorous, and add to a P system with bi-rules also an output membrane i_o; then, consider only halting computations, and let $N(\Pi)$ be the set of numbers computed by Π at the end of halting computations (the number of objects present in region i_o at the end of a computation is the result of that computation). Let $NOP_m(n_1, n_2; n_3, n_4)$ be the family of all sets of numbers $N(\Pi)$ computed by systems with at most m membranes and using bi-rules $([_i u_1 \to v_1]_i, [_j u_2 \to v_2]_j)$ with $|u_1| \leq n_1, |v_1| \leq n_2, |u_2| \leq n_3, |v_2| \leq n_3$, where $|w|$ denotes the length of the string w (hence, in our case, the total multiplicity of objects from the multiset represented by w). Let NTC denote the family of Turing computable sets of natural numbers, with number 0 excluded (our systems cannot compute sets of numbers which contain both 0 and a non-zero number: either the output region is initially empty and then no object can be introduced in it, or it is non-empty and then 0 cannot be computed).

Because P systems with antiport rules $(u, out; v, in)$ with $|u|, |v| \leq 2$, using only one membrane, are universal, and because such systems can be simulated by P systems with bi-rules using only the multisets u, v of antiport rules, we get the equality

$$NOP_1(2, 2; 2, 2) = NTC.$$

This result can be improved in what concerns the size of rules, and to this aim we use the notion of a matrix grammar with appearance checking. Such a grammar is a construct $G = (N, T, S, M, F)$, where N, T are disjoint alphabets, $S \in N$, M is a finite set of sequences of the form $(A_1 \to x_1, \ldots, A_n \to x_n)$, $n \geq 1$, of context-free rules over $N \cup T$ (with $A_i \in N, x_i \in (N \cup T)^*$, in all cases)[3], and F is a set of occurrences of rules in M (N is the nonterminal alphabet,

[3]For any alphabet V, V^* denotes the set of all strings over V, the empty string, denoted by λ, being included.

T is the terminal alphabet, S is the axiom, while the elements of M are called matrices; note the similarity of these matrices with our bi-rules).

For $w, z \in (N \cup T)^*$ we write $w \Longrightarrow z$ if there is a matrix $(A_1 \to x_1, \dots, A_n \to x_n)$ in M and the strings $w_i \in (N \cup T)^*, 1 \leq i \leq n+1$, such that $w = w_1, z = w_{n+1}$, and, for all $1 \leq i \leq n$, either (1) $w_i = w'_i A_i w''_i, w_{i+1} = w'_i x_i w''_i$, for some $w'_i, w''_i \in (N \cup T)^*$, or (2) $w_i = w_{i+1}$, A_i does not appear in w_i, and the rule $A_i \to x_i$ appears in F. (The rules of a matrix are applied in order, possibly skipping the rules in F if they cannot be applied – therefore we say that these rules are applied in the *appearance checking* mode.)

The language generated by G is defined by $L(G) = \{w \in T^* \mid S \Longrightarrow^* w\}$, where $\Longrightarrow^*$ is the reflexive and transitive closure of the relation $\Longrightarrow$. It is known that matrix grammars with appearance checking generate the family RE of recursively enumerable languages (and that the length sets of languages in RE are exactly the sets of numbers which are Turing computable).

A matrix grammar $G = (N, T, S, M, F)$ is said to be in the *binary normal form* if $N = N_1 \cup N_2 \cup \{S, \#\}$, with these three sets mutually disjoint, and the matrices in M are in one of the following forms:

1. $(S \to XA)$, with $X \in N_1, A \in N_2$,
2. $(X \to Y, A \to x)$, with $X, Y \in N_1, A \in N_2, x \in (N_2 \cup T)^*, |x| \leq 2$,
3. $(X \to Y, A \to \#)$, with $X, Y \in N_1, A \in N_2$,
4. $(X \to \lambda, A \to x)$, with $X \in N_1, A \in N_2$, and $x \in T^*, |x| \leq 2$.

Moreover, there is only one matrix of type 1 and F consists exactly of all rules $A \to \#$ appearing in matrices of type 3; $\#$ is a trap-symbol, because once introduced, it is never removed. A matrix of type 4 is used only once, in the last step of a derivation.

According to Lemma 1.3.7 in [7], for each matrix grammar there is an equivalent matrix grammar in the binary normal form.

With these prerequisites, we can now prove the following result.

Theorem 7.1 $NOP_1(2, 2; 1, 1) = NOP_1(1, 1; 2, 2) = NTC$.

Proof. We only prove the inclusions $NTC \subseteq NOP_1(2, 2; 1, 1)$, $NTC \subseteq NOP_1(2, 2; 1, 1)$; the converse inclusions can be proved in a straightforward (but tedious) way by constructing a Turing machine which simulates a given P system, or we can simply invoke the Turing-Church thesis.

Let us consider a set of numbers $Q \in NTC$, which is supposed not to contain the number 0. Then also the set $Q' = \{n - 1 \mid n \in Q\}$ is in NTC, hence it is the length set of the language generated by a matrix grammar with appearance checking. Take such a grammar $G = (N, T, S, M, F)$, with $Q' = \{|w| \mid w \in L(G)\}$. Without loss of generality, we may assume that $T = \{a\}$ (only the length of strings matters, not their contents), and that G is in the binary normal form, hence having $N = N_1 \cup N_2 \cup \{S, \#\}$ and the matrices from M in the four forms specified above. Because the matrices of types 2, 3, 4 are bi-rules, they can be directly considered bi-rules of a P system. The only precautions should be taken with the trap-symbol and with the fact that the strings x from matrices of types 2 and 4 can be empty. Thus, we construct the following P system:

$$\begin{aligned}
\Pi &= (O, [_1\ ]_1, XAd, d, \emptyset, R), \text{ where} \\
O &= N_1 \cup N_2 \cup \{a, d, \#\}, \\
R &= \{([_1 A \to x']_1, [_1 X \to Y]_1) \mid (X \to Y, A \to x) \in M\} \\
&\cup \{([_1 A \to x']_1, [_1 X \to d]_1) \mid (X \to \lambda, A \to x) \in M\} \\
&\cup \{([_1 A \to \#]_1, [_1 X \to Y]_1) \mid (X \to Y, A \to \#) \in M\} \\
&\cup \{([_0 d \to d]_0, [_1 \# \to \#]_1),\ ([_1 dd \to d]_1, [_0 d \to d]_0)\},
\end{aligned}$$

where $x' = x$ if $x \neq \lambda$, and $x' = d$ if $x = \lambda$.

The use of d in bi-rules ensures that all multisets are non-empty, but in this way an arbitrarily large number of copies of d can be introduced. Note that one occurrence of d is present from the beginning in the system, and the same is true for the environment. The copy of d from the environment is used both in making the computation last forever if the trap-symbol # was introduced in the grammar G, and for eliminating copies of d from the system. Specifically, the number of copies of d present in the system can be decreased to one by means of the bi-rule $([_0 d \to d]_0, [_1 dd \to d]_1)$, but the last copy cannot be eliminated. Consequently, in the end we get a multiset $a^n d$, where $a^n \in L(G)$, hence $n \in Q'$. This means that the system computes the number $n + 1 \in Q$, that is $N(\Pi) = Q$, which means that $Q \in NOP_1(2, 2; 1, 1)$.

By interchanging the rules of each bi-rule of Π, we get an equivalent system whose rules are of size bounded by $(1, 1; 2, 2)$ and this completes the proof. $\square$

If we allow also rules of the form $u \to v$ with empty v, then the above construction can be further simplified: instead of $([_0 d \to d]_0, [_1 dd \to d]_1)$ we can use $([_0 d \to d]_0, [_1 d \to \lambda]_1)$, hence the universality is obtained for systems with bi-rules bounded in length by $(1, 2; 1, 1)$ or by $(1, 1; 1, 2)$.

The conclusion of the results discussed above is that even very simple P systems are non-decidable, which makes impossible their uniform treatment without imposing restrictions on their evolution (for instance, examining only a finite time horizon, which then amounts at exploring a finite space of possible evolutions; combinatorially difficult, this task can however be carried out by means of computer simulations).

8 Final Remarks

This paper is a preliminary step towards a systematic examination of the possibility to use membrane computing as a framework for building models of economic processes. Both the interpretation of MC ingredients in terms of economics and the investigation of the above introduced P systems with bi-rules request further research efforts. However, we believe that such efforts are worth paying, because this framework seems suitable to address a series of questions relevant to economic studies. Here is a quick list of such questions: Does the evolution of a system enters a deadlock, where no further step is possible? Are there isolated subsystems, which never interact with the other subsystems? Are there "separation points" in time, when such isolated subsystems appear and never come back to cooperating with the rest of the system? What about subsystems which become not only isolated, but simply idle, unable to further evolve? What about cycles, overloaded subsystems (containing too many orders or too many scheduled objects), inflation (too many monetary units), balanced distribution of objects (in particular, of monetary units), pollution, unemployment, and so on and so forth? All these problems can be formulated in the symbolic framework of a P system (with bi-rules) and can also be answered, either by directly examining a given system, or, for limited time intervals, by computer experiments.

A whole range of questions may arise if we also consider the possibility of controlling the evolution of a system: assume that we start from a "good" configuration and we evolve (maybe non-deterministically), with the possibility of reaching both "good" and "bad" configurations. Find branching points which change the type of configurations immediately reached. What action can be taken in order to correct the system's trajectory? For instance, can the trajectory be kept going through "good" configurations by inserting new objects into the system? If so, can the necessary objects be introduced by rules placed in the environment? Which is the complexity of the sequence of multisets which correct the evolution of a system?

We stop here, and we will come back to related investigations in a forthcoming work.

References

[1] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, P. Walter: *Molecular Biology of the Cell*, 4th ed. Garland Science, New York, 2002.

[2] A. Alhazov, R. Freund, Y. Rogozhin: Computational power of symport/antiport: history, advances and open problems. In *Pre-Proceedings of Sixth Workshop on Membrane Computing, WMC6*, Vienna, July 2005, 44–78.

[3] J. Bartosik: Paun's systems in modeling of human resource management. *Second Conference on Tools and Methods of Data Transformation*, WSU Kielce, 2004.

[4] M. Cavaliere: Evolution-communication P systems. In *Membrane Computing. International Workshop, WMC 2002, Curtea de Argeş, Romania, August 2002. Revised Papers* (Gh. Păun, G. Rozenberg, A. Salomaa, C. Zandron, eds.), *Lecture Notes in Computer Science* 2597, Springer-Verlag, Berlin, 2003, 134–145.

[5] M. Cavaliere, V. Deufemia: Specifying dynamic software architectures by using membrane systems. In *Proc. Third Brainstorming Week on Membrane Computing*, Sevilla, 2005, RGNC Report 01/2005, 87–106.

[6] G. Ciobanu, Gh. Păun, M.J. Pérez-Jiménez, eds.: *Applications of Membrane Computing.* Springer-Verlag, Berlin, 2006.

[7] J. Dassow, Gh. Păun: *Regulated Rewriting in Formal Language Theory.* Springer-Verlag, Berlin, 1989.

[8] A. de la Fuente: *Mathematical Methods and Models for Economists.* Cambridge University Press, 2000.

[9] J.M. Epstein, R. Axtell: *Growing Artificial Societies – Social Science from the Bottom Up.* Brookings Institution Press and The MIT Press, 1996.

[10] R. Freund, L. Kari, M. Oswald, P. Sosik: Computationally universal P systems without priorities: two catalysts are sufficient. *Theoretical Computer Sci.*, 330, 2 (2005), 251–266.

[11] M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, A. Riscos-Núñez, F.J. Romero-Campero: On the power of dissolution in P systems with active membranes. In *Pre-Proceedings of Sixth Workshop on Membrane Computing, WMC6*, Vienna, July 2005, 373–394.

[12] W. Korczynski: On a model of economic systems. *Second Conference on Tools and Methods of Data Transformation*, WSU Kielce, 2004.

[13] W. Korczynski: Păun's systems and accounting. In *Pre-Proceedings of Sixth Workshop on Membrane Computing, WMC6*, Vienna, July 2005, 461–464.

[14] J. Kornai: *Anti-Equilibrium.* The Scientific Publ. House, Bucharest, 1974.

[15] M. Nowakowska: *Languages of Actions. Language of Motivation.* Mouton, The Hague, 1973.

[16] Gh. Păun: *Modelling Economic Processes by Means of Generative Grammars.* Technical Publ. House, Bucharest, 1980 (in Romanian).

[17] Gh. Păun: Computing with membranes. *J. Computer and System Sciences*, 61, 1 (2000), 108–143.

[18] Gh. Păun: *Membrane Computing – An Introduction.* Springer-Verlag, Berlin, 2002.

[19] G. Rozenberg, A. Salomaa, eds.: *Handbook of Formal Languages.* Springer-Verlag, Berlin, 1987.

[20] The P Systems Web Page: `http://psystems.disco.unimib.it`.