Chapter 2
Mobility in Membrane Computing

Abstract Membrane computing is part of natural computing, being a rule-based formalism inspired by biological cells. The basic model of membrane computing is usually called transition membrane systems. When membrane systems are considered as computing devices, two main research directions are considered: their computational power in comparison with the classical notion of Turing computability, and their efficiency in algorithmically solving hard problems (e.g., NP-problems) in polynomial time. In this chapter we define mobile membrane systems which are both powerful (equivalent to Turing machines) and efficient (algorithms have been developed which provide efficient solutions to NP-complete problems).

2.1 Mobility in Cell Biology

The cell is the functional basic unit of life, and is often called the building block of life [7]. The cells of living organisms are categorised into prokaryotic and eukaryotic cells. Any organism contains either prokaryotic or eukaryotic cells. Prokaryotic are very simple cells that are smaller than the more complex eukaryotic cells. Bacteria are made up of one or more prokaryotic cells. The cell membrane in a prokaryotic cell consists of a fluid phospholipid bilayer without carbohydrates. The prokaryotic cell is incapable of endocytosis or exocytosis unlike the eukaryotic cell.

Eukaryotic cells are found inside plant and animal life and are more advanced and larger, and differ fundamentally from their prokaryotic counter-parts by their possession of membrane-bounded internal compartments. The emergence of an endomembrane system was a crucial stage in the prokaryote-to-eukaryote evolutionary transition [76]. A feature that distinguishes prokaryote cells from eukaryote cells is the presence in eukaryote cells of a cytoskeleton that maintains their structural integrity. Thus the cell can afford to have a membrane that consists of a fluid phospholipid bilayer that includes carbohydrates. The increased fluidity of the outer membrane allowed the development of two mechanisms, called endocytosis and exocytosis. Endocytosis and exocytosis are complementary operation that allow substances

B. Aman, G. Ciobanu, *Mobility in Process Calculi and Natural Computing*, Natural Computing Series, DOI 10.1007/978-3-642-24867-2_2,

to enter (endocytosis) or exit (exocytosis) the cell through membrane-bounded vesicles. In endocytosis, the vesicle is formed by the invagination of a small segment of the outer membrane that contains substances from outside the cell. In the reverse process of exocytosis, the vesicle fuses with the outer cell membrane releasing its content into the extracellular space.

The development of endocytosis and exocytosis prepared the way for all subsequent steps of eukaryotic evolution [100]. According to [76], several innovations appeared during this evolution from prokaryote to eukaryote cells (see Figure 2.1 and Table 2.1).

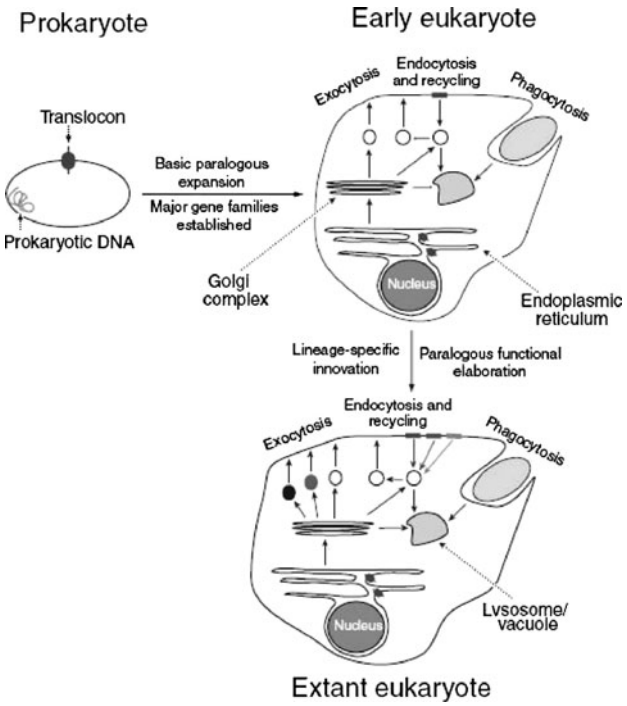


Fig. 2.1 Prokaryote to Eukaryote Evolutionary Transition

Table 2.1 Major Innovations in the Evolution of the Endomembrane System

Prokaryote	Early Eukaryote	Extant Eukaryote
Unfolded substrate exported	Folded substrate exported	Multiple modes of endocytosis
Membrane translocation	Vesicular transport	Multiple modes of exocytosis
	Limited modes of endocytosis	Tissue-specific pathways
	Phagocytosis	Lineage-specific pathways
	Lysosomal degradation	Secondary losses

Endocytosis of eukaryotic cells also enhances communication between cells [150], a feature that enables eukaryote cells to form multicellular organisms. Cells sense

the environment and communicate with each other through activation of signalling receptors on the cell surface. Endocytosis regulates cell signalling by physically reducing the number of signalling receptors available for activation. In some cases, a reduction in the number of surface receptors does not attenuate the maximal signalling response, but instead shifts the dose response relationship so that a higher concentration of ligand is required to trigger a response of the same magnitude. This is of physiological importance in settings of limited ligand concentration. In this way, endocytosis and cell signaling are intertwined in many biological processes, such as cell motility and cell fate determination [150].

We use endocytosis and exocytosis as abstract operations in membrane computing, a new biologically inspired model of computation.

2.2 Membrane Computing

Membrane computing is part of natural computing, being a rule-based formalism inspired by biological cells. The basic model of membrane computing is usually referred to as transition membrane systems. In this model, objects are represented using symbols from a given alphabet, and each symbol from this alphabet can appear inside a region in many different copies. That is, the content of a region is represented as a multiset over a given alphabet. Rules are essentially multiset rewriting rules with some extra features: targets appear in the right hand side of the rules and are used to specify where to move the produced objects.

Definition 2.1. A transition membrane system of degree $n \geq 1$ is a construct

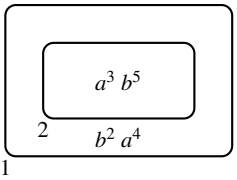
$$\Pi = (V, H, T, C, \mu, w_1, \ldots, w_n, (R_1, \rho_1), \ldots, (R_n, \rho_n), i_O)$$

where:

1. n represents the number of membranes;
2. V is an alphabet of symbols; its elements are called objects;
3. $T \subseteq V$ is the terminal (or output) alphabet;
4. $C \subseteq V$, $C \cap T = \emptyset$ is the alphabet of catalysts;
5. H is a finite set of labels for membranes;
6. $\mu \subset H \times H$ describes the membrane structure, such that $(i, j) \in \mu$ denotes that the membrane labelled by j is contained within the membrane labelled by i; we distinguish the external membrane (usually called the "skin" membrane) and several internal membranes; a membrane without any other membrane inside it is said to be elementary;
7. $w_i \in V^*$, for each $1 \leq i \leq n$ is a multiset of objects assigned initially to membrane i;
8. R_i, for all $1 \leq i \leq n$, is a finite set of evolution rules which is associated with membrane i; an evolution rule is a multiset rewriting rule of the form $u \to v$, with $u \in V^+$, $v = v'$ or $v = v'\delta$, $v' \in ((V \times \{here, out\}) \cup (V \times \{in_j \mid 1 \leq j \leq n\}))^*$, and δ a special symbol not appearing in V;

9. ρ_i, for all $1 \leq i \leq n$, is a partial order relationship defined over the rules in R_i specifying a priority relation between these rules;
10. i_O is the label of an elementary membrane of μ which identifies the output region.

$$R_1 = \{r_1 : a \rightarrow (a, in_2)\}$$
$$\cup \{r_2 : b \rightarrow (a, in_2)\}$$
$$R_2 = \{r_3 : a \rightarrow (b, out)(a, here)\}$$
$$\cup \{r_4 : b \rightarrow (b, out)\}$$



As an example, we consider a membrane system with two nested membranes (the inner membrane labelled by 2, the outer membrane labelled by 1), two sets of evolution rules R_2 and R_1 and two symbols (a and b). Initially, membrane 1 contains the multiset $b^2\ a^4$, and membrane 2 contains the multiset $a^3\ b^5$.

Fig. 2.2 A Transition Membrane System

Therefore, a transition membrane system of degree $n \geq 1$ consists of a membrane structure μ containing n membranes where each membrane i is assigned a finite multiset of objects w_i and a finite set of evolution rules R_i, with an associated priority relation ρ_i. An evolution rule is a multiset rewriting rule which consumes a multiset of objects from V and produces a multiset of pairs (a,t), with $a \in V$ and $t \in \{here, out, in\}$ a *target* specifying where to move the objects after the application of the rule. As well as this, an evolution rule can produce the special object δ to specify that, after the application of the rule, the membrane where the rule has been applied has to be dissolved. After dissolving a membrane, all objects and membranes previously contained in it become contained in the membrane containing it, while the rules of the dissolved membrane are removed.

2.3 Mobile Membranes

Mobile membranes represent a formalism describing the movement of membranes inside a spatial structure by applying rules from a given set. Mobile membranes represent a variant of membrane computing [128]. Several systems of mobile membranes are studied in [17], and their computational universality are proved by using a small number of membranes [18]. The model is characterized by two essential features:

- A spatial structure consisting of a hierarchy of membranes (which do not intersect). The membranes produce a delimitation between regions. For each membrane there is a unique associated region. To each membrane we associate a multiset of objects placed on its surface and a multiset of objects placed inside the

corresponding region. A membrane without any other membranes inside is called elementary, while a membrane containing other membranes is called composite.
- The general rules describing the evolution of the structure: endocytosis (moving an elementary membrane inside a neighbouring membrane) and exocytosis (moving an elementary membrane outside the membrane where it is placed). More specific rules are given by pinocytosis (creating an elementary membrane) and phagocytosis (engulfing just one sibling elementary membrane). A movement rule consists in fact of two steps: rewriting the objects that initiated the movement to multisets of objects and changing the membrane structure.

The computations are performed in the following way: starting from an initial structure, the system evolves by applying the rules in a nondeterministic and maximally parallel manner. A rule is applicable when all the involved objects and membranes appearing in its left hand side are available. 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 set. A halting configuration is reached when no rule is applicable. The result is represented by the number of objects associated to a specified membrane.

Let $\mathbb{N}$ be the set of positive integers, and consider a finite alphabet V of symbols. A multiset over V is a mapping $u : V \to \mathbb{N}$. The empty multiset is represented by λ. We use the string representation of multisets that is widely used in the field of membrane systems. An example of such a representation is the multiset $u = aabca$, where $u(a) = 3$, $u(b) = 1$, $u(c) = 1$. Using such a representation, the operations over multisets are defined as operations over strings. Given two multisets u, v over V, for any $a \in V$, we have $(u \uplus v)(a) = u(a) + v(a)$ as the multiset union, and $(u \backslash v)(a) = max\{0, u(a) - v(a)\}$ as the multiset difference.

2.3.1 Simple Mobile Membranes

A first definition of mobile P systems is given in [133] with rules coming from mobile ambients [42]. Inspired by the operations of endocytosis and exocytosis, namely moving a membrane inside a neighbouring membrane (endocytosis) and moving a membrane outside the membrane where it is placed (exocytosis), P systems with mobile membranes are introduced in [108] as a variant of P systems with active membranes [128]. We use the name *simple mobile membrane system* instead of P systems with mobile membranes.

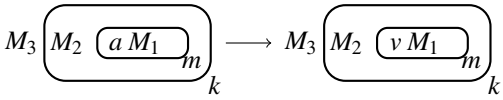
Definition 2.2. A simple mobile membrane system is a construct

$$\Pi = (V, H, \mu, w_1, \ldots, w_n, R, i_O)$$

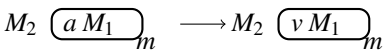
where:

1. n, V, H, μ, w_i, i_O are as in Definition 2.1;
2. R is a finite set of developmental rules, of the following forms:

(a) $[[a \to v]_m]_k$, for $k, m \in H$, $a \in V$, $v \in V^*$; local object evolution

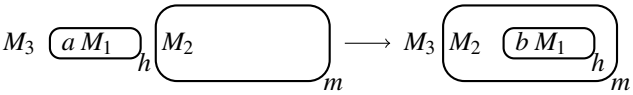


These rules are called *local* because the evolution of an object a of membrane m is possible only when membrane m is inside membrane k. By M_1, M_2 and M_3 we denote (possible empty) multisets of objects, elementary and composite membranes.



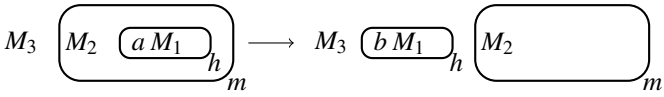
If the restriction of nested membranes is not imposed, that is, the evolution of the object a in membrane m is allowed irrespective of where membrane m is placed, then we say that we have a global evolution rule, and write it simply as $[a \to v]_m$. By M_1 and M_2 we denote (possible empty) multisets of objects, elementary and composite membranes.

(b) $[a]_h[\,]_m \to [[b]_h]_m$, for $h, m \in H$, $a, b \in V$; endocytosis



An elementary membrane labelled h enters the adjacent membrane labelled m, under the control of object a; the labels h and m remain unchanged during the process; however the object a is modified to b during the operation; m is not necessarily an elementary membrane. By M_1 we denote a (possibly empty) multiset of objects, and by M_2 and M_3 we denote (possibly empty) multisets of objects, elementary and composite membranes.

(c) $[[a]_h]_m \to [b]_h[\,]_m$, for $h, m \in H$, $a, b \in V$; exocytosis



An elementary membrane labelled h is sent out of a membrane labelled m, under the control of object a; the labels of the two membranes remain unchanged, but the object a of membrane h is modified during this operation; membrane m is not necessarily elementary. By M_1 we denote a (possibly empty) multiset of objects, and by M_2 and M_3 we denote (possibly empty) multisets of objects, elementary and composite membranes.

The rules are applied according to the following principles:

1. Rules are applied in parallel, non-deterministically choosing the rules, the membranes, and the objects in such a way that the parallelism is maximal; this means that in each step we apply a certain set of rules 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*, while the membrane h is said to be *active*. In any step of a computation, any object and any active membrane can be involved in at most one rule. However, the passive membranes can be used by several rules at the same time. In a rule $[a \rightarrow v]_m$ of type (a), object a is active, while membrane m is passive.
3. When a membrane is moved across another membrane, by endocytosis or exocytosis, its whole contents (its objects) are moved; the inner objects evolve first (if rules are applicable for them), and then any membrane is moved with the contents as obtained after its internal evolution.
4. If a membrane exits the system (by exocytosis), then its internal evolution stops, even if there are rules of type (a) which could be applied.
5. The objects and membranes which do not evolve at a given step are passed unchanged to the next configuration of the system.

By using the rules in this way, we get transitions among the configurations of the system. A sequence of transitions is a computation, and a computation is successful if, starting from the initial configuration, it halts (it reaches a configuration where no rule can be applied). The multiplicity vector of the multiset from a special membrane called the output membrane is considered to be the 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. The set of vectors of natural numbers produced in this way by a system Π is denoted by $Ps(\Pi)$. A computation can produce several vectors, all of them considered in the set $Ps(\Pi)$.

Hence a computation is structured as follows: it starts with an initial configuration of the system (the initial membrane structure and the initial distribution of objects within regions), then the computation proceeds, and when it stops the result is to be found in a specific output membrane.

2.3.2 Enhanced Mobile Membranes

Enhanced mobile membranes were introduced in [16] for describing some biological mechanisms of the immune system. The presentation of the immune system is taken from [98], a book which is revised every few years to keep pace with the new discoveries in this field. The cells of the immune system work together with different proteins to seek out and destroy anything foreign or dangerous which enters our body. It takes some time for the immune cell to be activated, but once this happens very few hostile organisms have a chance. There are several types of immune cells, each of them with its own strengths and weaknesses. Some seek out and engulf invaders, while others destroy infected or mutated body cells. One type of immune cells are the B cells which have the ability to release special proteins called antibodies which mark intruders so that they may be destroyed by macrophages. The immune system also has the ability to produce some cells able to remember ene-

mies which it fought in the past. In this way, once the immune system recognizes an invader it attacks more quickly and strongly against it.

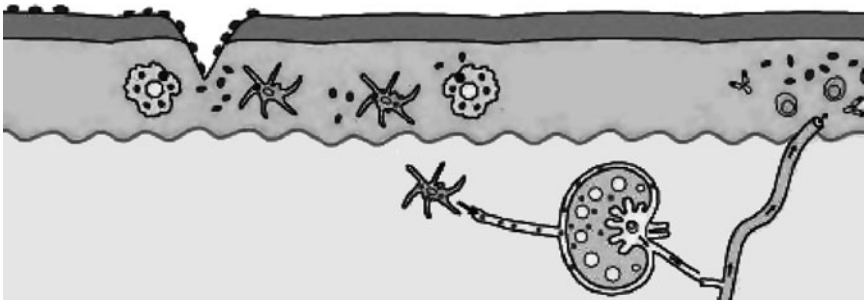


Fig. 2.3 Immune System Mechanisms

Dendritic cells can engulf bacteria, viruses, and other cells. Once a dendritic cells engulfs a bacterium, it dissolves this bacterium and places portions of bacterium proteins on its surface (see Figure 2.3). These surface markers serve as an alarm to other immune cells, namely helper T cells, which then infer the form of the invader. This mechanism makes the T cells sensitive to recognize the antigens or other foreign agents which trigger a reaction of the immune system. Antigens are often found on the surface of bacteria and viruses.

New rules are introduced following this biological example. We define a new variant of mobile membranes, namely the enhanced mobile membranes, originally introduced in [16]. The object a is the one indicating the membrane which initializes the move in the rules of type $(a)-(d)$.

Definition 2.3. *An enhanced mobile membrane system* is a construct
$$\Pi = (V, H, \mu, w_1, \ldots, w_n, R, i_O), \text{ where:}$$

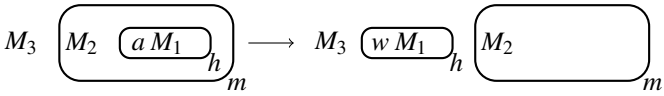
1. n, V, H, μ, w_i, i_O are as in Definition 2.1;
2. R is a finite set of *developmental rules* of the following forms:

(a) $[a]_h[\,]_m \to [[w]_h]_m$, for $h, m \in H, a \in V, w \in V^*$; endo

$$M_3\ \underset{h}{\boxed{a\ M_1}}\ \underset{m}{\boxed{M_2}} \longrightarrow M_3\ \underset{m}{\boxed{M_2\ \underset{h}{\boxed{w\ M_1}}}}$$

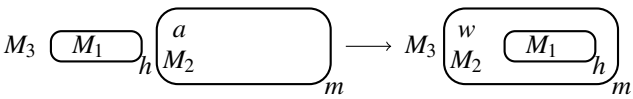
An elementary membrane labelled h enters the adjacent membrane labelled m, under the control of object a; the labels h and m remain unchanged during this process, however, the object a is modified to w during the operation; m is not necessarily an elementary membrane.

(b) $[[a]_h]_m \to [w]_h[\,]_m$, for $h, m \in H, a \in V, w \in V^*$; exo



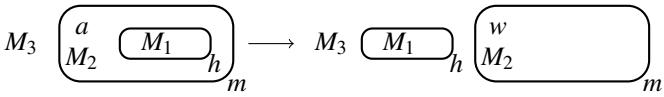
An elementary membrane labelled h exits a membrane labelled m, under the control of object a; the labels of the two membranes remain unchanged, but the object a from membrane h is modified during this operation; membrane m is not necessarily elementary. By M_1 we denote a (possibly empty) multiset of objects, and by M_2 and M_3 we denote (possibly empty) multisets of objects, elementary and composite membranes.

(c) $[\,]_h[a]_m \to [[\,]_h w]_m$, for $h,m \in H, a \in V, w \in V^*$; fendo



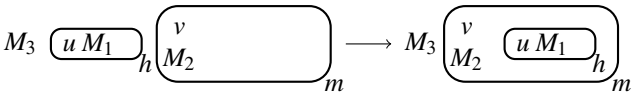
An elementary membrane labelled h is engulfed by the adjacent membrane labelled m, under the control of object a of m; the labels h and m remain unchanged during this process, however, the object a is modified to w during the operation; m is not necessarily an elementary membrane. By M_1 we denote a (possibly empty) multiset of objects, and by M_2 and M_3 we denote (possibly empty) multisets of objects, elementary and composite membranes.

(d) $[a[\,]_h]_m \to [\,]_h[w]_m$, for $h,m \in H, a \in V, w \in V^*$; fexo



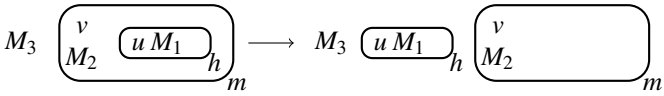
An elementary membrane labelled h is expelled by a membrane labelled m, under the control of object a of m; the labels of the two membranes remain unchanged, but the object a from membrane m is modified during this operation; membrane m is not necessarily elementary. By M_1 we denote a (possibly empty) multiset of objects, and by M_2 and M_3 we denote (possibly empty) multisets of objects, elementary and composite membranes.

(e) $[u]_h[v]_m \to [[u]_h v]_m$, for $h,m \in H, u,v \in V^*$; pendo



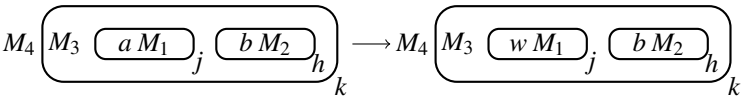
An elementary membrane labelled h containing u enters the adjacent membrane containing v; the objects do not evolve in the process. By M_1 we denote a (possibly empty) multiset of objects, and by M_2 and M_3 we denote (possibly empty) multisets of objects, elementary and composite membranes.

(f) $[v[u]_h]_m \to [u]_h[v]_m$, for $h,m \in H, u,v \in V^*$; pexo



An elementary membrane labelled h containing u comes out of the membrane labelled m containing v. The objects do not evolve in the process. By M_1 we denote a (possibly empty) multiset of objects, and by M_2 and M_3 we denote (possibly empty) multisets of objects, elementary and composite membranes.

(g) $[[a]_j[b]_h]_k \longrightarrow [[w]_j[b]_h]_k$ for $h, j, k \in H, a, b \in V, w \in V^*$; cevol



An object a in membrane m evolves into w when membranes h and m are adjacent to each other inside membrane k. By M_1, M_2, M_3 and M_4 we denote (possibly empty) multisets of objects, elementary and composite membranes.

The rules of enhanced mobile membranes are applied according to the principles of simple mobile membranes. Here *endo* and *exo* represent endocytosis and exocytosis, $fendo$ and $fexo$ represent forced endocytosis and forced exocytosis, *pendo* and *pexo* represent pure endocytosis and pure exocytosis, while *cevol* represents contextual evolution. When we restrict $|w| = 1$ in rules (a) - (d), we call the operations *rendo*, *rexo*, $rfendo$ and $rfexo$ where r stands for "restricted".

Example 2.1. In order to simulate the evolution presented in Figure 2.3, we need first to encode all the components of the immune system into a membrane system. This can be realized by associating a membrane to each component, and objects to the signals, states and parts of molecules. For the steps done by the dendritic cells presented in Figure 1 we use the following encodings:

- dendritic cell - $[eat]_{DC}$

  An immature dendritic cell is willing to eat any bacterium it encounters, so we translate it into a membrane labelled by *DC* which has inside an object *eat* used to engulf the bacterium. Once the dendritic cell matures, the object *eat* is consumed.
- bacterium cell - $[antigen]_{bacterium}$

  A bacterium cell contains antigen so we simply represent it as a membrane labelled by *bacterium* containing a single object *antigen* which contains the information of the bacterium.
- lymph node - $[\,]_{lymph\ node}$

  The lymph node is the place to which the mature dendritic cells migrate in order to start the immune response, so we translate it into a membrane labelled by *lymph node*.

Using the above membranes we can describe the membrane system as follows (here *skin* stands for the body skin):

$$[[eat]_{DC}[\,]_{lymph\ node}]_{skin}[antigen]_{bacterium}$$

with the following rules which describe its evolution:

- $[antigen]_{bacterium}[\,]_{skin} \rightarrow [[antigen]_{bacterium}]_{skin}$

  A bacterium enters through the skin by performing an endocytosis rule in order to infect the body. The bacterium contains an object *antigen* which represent its signature.
- $[eat]_{DC}[\,]_{bacterium} \rightarrow [eat[\,]_{bacterium}]_{DC}$

  Once an immature dendritic cell becomes sibling to a bacterium, it "eats" the bacterium by performing a forced endocytosis rule. Until this moment the bacterium has controlled its own movement; in this step of the evolution the movement becomes controlled by the dendritic cell which eats the bacterium.
- $[[antigen]_{bacterium}]_{DC} \rightarrow [[antigen\ \delta]_{bacterium}]_{DC}$

  Once the bacterium has entered the dendritic cell, an object δ is created in order to dissolve the membrane *bacterium*, and the content of the bacterium is released into the dendritic cell.
- $[antigen]_{DC}[\,]_{lymph\ node} \rightarrow [[antigen]_{DC}]_{lymph\ node}$

  Once the dendritic cell contains antigen, it enters the lymph node in order to activate a special class of T cells, namely the helper T cells.
- $[[eat]_{DC}]_{lymph\ node} \rightarrow [[\,]_{DC}]_{lymph\ node}$

  Once the dendritic cell enters the lymph node it matures and the capacity to engulf bacteria disappears, namely the *eat* object is consumed.

Using only these few rules we can simulate the way a bacterium is engulfed and its content is displayed by the eater cell. The proteins produced by helper T cells activate the B cells.

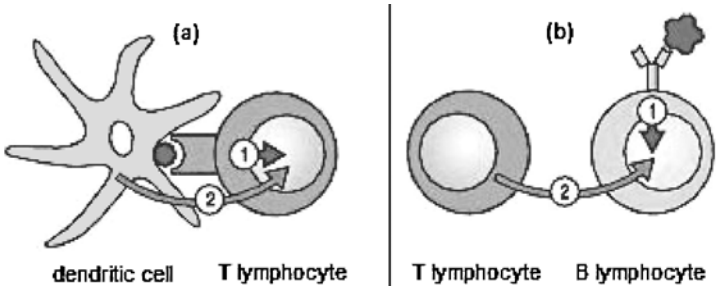


Fig. 2.4 Activation of T cells (a) and B cells (b)

For the process of activating the helper T cells and B cells we use the following encodings:

- helper T cell - $[passive]_{helper\ T\ cell}$

A helper T cell is initially passive, so we represent it as a membrane labelled *helper T cell* in which the object *passive* is placed. When the cell is activated the object *passive* is transformed into *active*.

- B cell - $[passive]_{B\ cell}$

  A B cell is initially passive, so we represent it as a membrane labelled *B cell* in which the object *passive* is placed. When the cell is activated the object *passive* is transformed into *active*.

The activation of the helper T cells and B cells is conditioned by the presence in the lymph node of the dendritic cells, and that is why we use the following contextual evolution rules:

- $[[antigen]_{DC}[passive]_{helper\ T\ cell}]_{lymph\ node} \rightarrow$
  $[[antigen]_{DC}[active]_{helper\ T\ cell}]_{lymph\ node}$

  Once the dendritic cell containing antigen enters the lymph node, it activates a special class of T cells, namely the helper T cells. This is denoted by changing the object *passive* to *active* in helper T cells.
- $[[passive]_{B\ cell}[active]_{helper\ T\ cell}]_{lymph\ node} \rightarrow$
  $[[active]_{B\ cell}[active]_{helper\ T\ cell}]_{lymph\ node}$

  Once the helper T cells are activated, the B cells that are sibling with them are the next cells which are activated.

The B cell searches for antigen matching its receptors. If it finds such antigen, then inside the B cell a triggering signal is set off. Using the proteins produced by helper T cells, the B cell starts to divide and produce clones of itself. During this process, two new cell types are created: plasma cells which produce an antibody, and memory cells which are used to "remember" specific intruders.

These examples motivate the introduction of the new class of mobile membranes; more exactly, they motivate the new rules and the way they can be used in modelling some biological systems.

2.3.3 Mutual Mobile Membranes

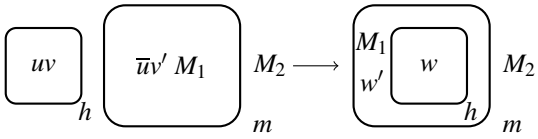
Mutual mobile membrane systems represent a variant of mobile membrane systems in which endocytosis and exocytosis work whenever the involved membranes "agree" on the movement. This agreement is described by using dual objects a and $\overline{a}$ in the involved membranes, with $\overline{\overline{a}} = a$. The duality relation is distributive over a multiset, namely $\overline{u} = \overline{a_1} \ldots \overline{a_n}$ for $u = a_1 \ldots a_n$.

Definition 2.4. A mutual mobile membrane system is a construct

$$\prod = (V, H, \mu, w_1, \ldots, w_n, R, i_O), \quad \text{where:}$$

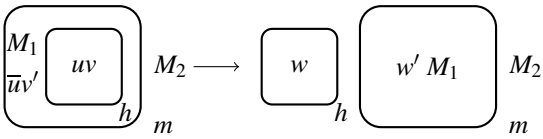
1. n, V, H, μ, w_i, i_O are as in Definition 2.1;
2. R is a finite set of developmental rules of the following forms:

(a) $[uv]_h[\overline{u}v']_m \rightarrow [\,[w]_h w']_m$ for $h,m \in H, u,\overline{u} \in V^+, v,v',w,w' \in V^*$; mendo



An elementary membrane labelled h enters the adjacent membrane labelled m under the control of the multisets of objects uv and $\overline{u}v'$. The labels h and m remain unchanged during this process; however the multisets of objects uv and $\overline{u}v'$ are replaced with the multisets of objects w and w', respectively. By M_1 and M_2 we denote (possibly empty) multisets of objects, elementary and composite membranes.

(b) $[\overline{u}v'[uv]_h]_m \rightarrow [w]_h[w']_m$ for $h,m \in H, u,\overline{u} \in V^+, v,v',w,w' \in V^*$; mexo



An elementary membrane labelled h exits a membrane labelled m, under the control of the multisets of objects uv and $\overline{u}v'$. The labels of the two membranes remain unchanged, but the multisets of objects uv and $\overline{u}v'$ are replaced with the multisets of objects w and w', respectively. By M_1 and M_2 we denote (possibly empty) multisets of objects, elementary and composite membranes.

The rules of the mutual mobile membranes are applied according to the principles of simple mobile membranes. Here *mendo* and *mexo* represent mutual endocytosis and mutual exocytosis. A multiset u indicates the membrane which initializes the move in the rules of type $(a)-(b)$, while a multiset $\overline{u}$ indicates the membrane which accepts the movement.

2.3.4 Mutual Mobile Membranes with Objects on Surface

Membrane fusion occurs when two separate membranes containing complementary proteins merge into a single membrane. The process described in Figure 2.5 is performed in several well-distinguished steps.

Initially, the two involved membranes mutually identify each other by means of complementary proteins: v-SNARES and t-SNARES. Then SNARES located on the vesicles (v-SNARES) and on the target membranes (t-SNARES) interact with one another to form a stable complex that brings the two membranes very close. Finally, the vesicle and target membranes distort and then fuse. Each vesicle must only fuse with the correct target membrane in order to avoid an unwanted mixing of proteins.

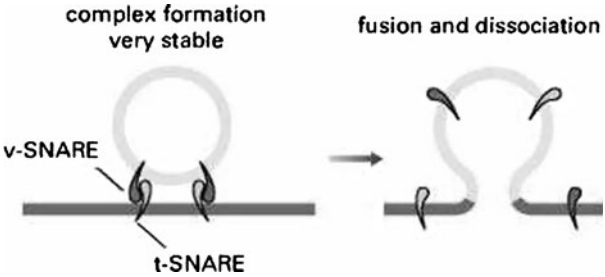


Fig. 2.5 Vesicle Fusion Mediated by Complex Formation of Complementary SNARES

These biological facts provide the motivation for using objects and co-objects for the pino, exo and phago rules as done in [20]. These rules are also related to the formal approach defined in [45]. After presenting some technical notions, we define systems of mutual mobile membranes with objects on surface.

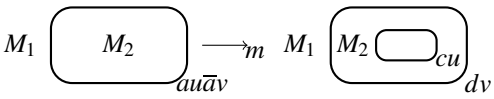
Definition 2.5. A *system of n mutual mobile membranes with objects on surface* (M^3OS_n) is a construct

$$\Pi = (V, \mu, u_1, \ldots, u_n, R, i_O)$$

where:

1. n, V, i_O are as in Definition 2.1;
2. μ is a membrane hierarchical structure with $n \geq 2$ membranes;
3. $u_1, \ldots, u_n$ are multisets of proteins (represented by strings over V) bound to the n membranes at the beginning of the computation; the membranes are bijectively mapped to $\{1, \ldots, n\}$; the skin membrane is labelled with 1 and $u_1 = \varepsilon$;
4. R is a finite set of rules of the following forms:

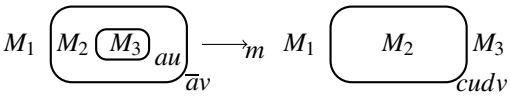
(a) $[\,]_{au\overline{a}v} \rightarrow_m [[\,]_{cu}]_{dv}$, for $a, \overline{a} \in V, c, d, u, v \in V^*$ pino



An object a together with a complementary object $\overline{a}$ model the creation of an empty membrane within the membrane on which are objects a and $\overline{a}$. We should imagine that the original membrane buckles towards the inside, and pinches off by breaking the connection between a and $\overline{a}$. The multiset of objects u on the newly created (empty) membrane is transferred from the initial membrane. The objects a and $\overline{a}$ are modified during this step to the multisets c and d, respectively. On the surface of the membrane appearing on the left hand side of the rule there are some objects (other than $au\overline{a}v$) which are ignored;

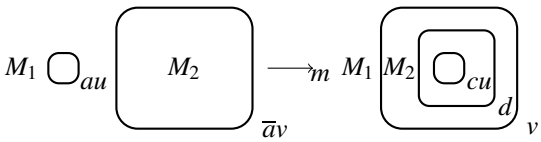
these objects are also not specified on the right hand side of the rule, being randomly distributed between the two resulting membranes. By M_1 and M_2 we denote (possibly empty) multisets of elementary and composite membranes.

(b) $[[\,]_{au}]_{\overline{a}v} \to_m [\,]_{cudv}$, for $a, \overline{a} \in V, c, d, u, v \in V^*$ exo



An object a together with a complementary object $\overline{a}$ model the merging of a nested membrane with its surrounding membrane. We should imagine that the connection between a and $\overline{a}$ represent the point where the membranes connect to each other. In this merging process (which is a smooth and continuous process), the membrane having the multiset au on its surface gets 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 $\overline{a}$ are modified during this evolution to 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 on the left hand side of the rule there are some objects (other 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 onto the resulting membrane. By M_1, M_2 and M_3 we denote (possibly empty) multisets of elementary and composite membranes.

(c) $[\,]_{au}[\,]_{\overline{a}v} \to_m [[[\,]_{cu}]_d]_v$, for $a, \overline{a} \in V, c, d, u, v \in V^*$ phago



An object a together with its complementary object $\overline{a}$ model 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 to 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 newly created membrane). On the surface of the membranes appearing on the left hand side of the rule there are some objects (other 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 initially having the object a

on surface remain unchanged, while the objects appearing on the membrane initially having 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 we denote (possibly empty) multisets of elementary and composite membranes.

The rules of mutual mobile membranes with objects on surface are applied according to the principles of simple mobile membranes.

2.4 Computability Power of Mobile Membranes

Several notions and notations from the field of formal languages that are used here can be found in [80] and [142].

2.4.1 Preliminaries

For an alphabet $V = \{a_1, \ldots, a_n\}$, we denote by V^* the set of all strings over V; λ denotes the empty string. V^* is a monoid with λ as its unit element. For a string $x \in V^*$, $|x|_a$ denotes the number of occurrences of symbol a in x. A multiset over V is represented by a string over V (together with all its permutations), and each string precisely identifies a multiset. For an alphabet V, the *Parikh vector* is $\psi_V : V^* \to \mathbf{N}^n$ with $\psi_V(x) = (|x|_{a_1}, \ldots, |x|_{a_n})$, for all $x \in V^*$. For a language L, the Parikh vector is $\psi_V(L) = \{\psi_V(x) \mid x \in L\}$, while for a family FL of languages, the Parikh vector is $PsFL = \{\psi_V(L) \mid L \in FL\}$.

Definition 2.6. A *matrix grammar with appearance checking* $G = (N, T, S, M, F)$ is a construct where N, T are disjoint alphabets of non-terminals and terminals, $S \in N$ is the axiom, M is a finite set of matrices $(A_1 \to x_1, \ldots, A_n \to x_n)$ of context-free rules, and F is a set of occurrences of rules in M. For $w, z \in (N \cup T)^*$, we write $w \Rightarrow_m z$ if there is a matrix $m = (A_1 \to x_1, \ldots, A_n \to x_n)$ 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 i, $1 \leq i \leq n$, 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 language generated by G is $L(G) = \{x \in T^* \mid S \Rightarrow^* x\}$.

Definition 2.7. A *matrix grammar in the strong binary form* $G = (N, T, S, M, F)$ is a construct where $N = N_1 \cup N_2 \cup \{S, \#\}$, with these three sets mutually disjoint, two distinguished symbols $B^{(1)}, B^{(2)} \in N_2$, and the matrices in M of 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) $(X \to Y, B^{(j)} \to \#)$, with $X, Y \in N_1$, $j = 1, 2$;

(4) $(X \to \lambda, A \to x)$, with $X \in N_1, A \in N_2, x \in T^*$, $|x| \leq 2$.

If we ignore the empty string when comparing languages, then the rules of type (4) are of the form $(X \to a, A \to x)$, with $X \in N_1$, $a \in T$, $A \in N_2$, $x \in T^*$.

We denote by *PsRE* the family of Turing computable sets of vectors generated by arbitrary grammars.

By $\mathbf{N}RCM(M, CF - \lambda, ac)$ and $\mathbf{N}MAT_{ac}$ we denote the families of sets of numbers computed by random context non-erasing matrix grammars with appearance checking, and non-erasing matrix grammars with appearance checking, respectively. These can also be looked at as the families of sets of numbers recognized by these languages. It is known that $\mathbf{N}RCM(M, CF - \lambda, ac) = \mathbf{N}MAT_{ac} \subset \mathbf{N}RE$ (see [80]).

Definition 2.8 (Left Quotient). The left quotient of a language L by a letter a is given by $\partial_a(L) = \{x \mid ax \in L\}$.

Definition 2.9 (Random Context Matrix Grammars). A random context matrix grammar is a construct $G = (N, T, M, S, F)$ where N, T, S are as in a usual matrix grammar and M is a finite set of triples $((A_1 \to x_1, A_2 \to x_2, \ldots, A_n \to x_n), Q, R)$ where $A_i \to x_i$ are context-free rules, $1 \leq i \leq n$, $Q, R \subseteq N$, $Q \cap R = \emptyset$. A matrix can be applied to a string $x = x_1 X_1 x_2 X_2 \ldots X_l x_{l+1}$ in order to rewrite effectively the symbols $X_1, \ldots, X_l$ only if $x_1, \ldots x_{l+1}$ contains all symbols of Q and no symbols of R. We denote by $RCM(M, \beta, \max(\alpha, \gamma))$ the family of languages generated by random context matrix grammars $G = (N, T, S, M, F)$ with rules of type β, with $\beta \in \{CF, CF - \lambda\}$. If $\gamma = ac$, then F is arbitrary, and if γ is empty, then $F = \emptyset$. If $\alpha = ac$, then R is arbitrary in $((r_1, \ldots, r_n), Q, R) \in M$ and if α is empty, no forbidding contexts are involved. $\max(\alpha, \gamma) = ac$ if at least one of α, γ is ac. Thus, if no appearance checking is used, and if no forbidding contexts are used, we have the family $RCM(M, \beta, \emptyset)$.

Minsky introduced the concept of register machines in [125] by showing that the power of Turing machines can be achieved by such abstract machines using a finite 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.

Definition 2.10 (Register Machine). An n-register machine is $M = (n, B, l_0, l_h, I)$, where:

- n is the number of registers; B is a set of labels; l_0 and l_h are the labels of the initial and halting instructions;
- I is a set of labelled instructions of the form $l_i : (op(r), l_j, l_k)$, where $op(r)$ is an operation on register r of M, and l_i, l_j, l_k are labels from the set B.
- the machine is capable of the following instructions:

  1. $l_i : (ADD(r), l_j, l_k)$: Add one to the content of register r and proceed, in a non-deterministic way, to instruction with label l_j or to instruction with label l_k; in the deterministic variant, $l_j = l_k$ and then the instruction is written in the form $l_i : (ADD(r), l_j)$.

2. $l_i : (SUB(r), l_j, l_k)$: If register r is not empty, then subtract one from its contents and go to instruction with label l_j, otherwise proceed to instruction with label l_k.
3. $l_h : halt$: This instruction stops the machine and can only be assigned to the final label l_h.

Theorem 2.1 ([146]). *A 3-register machine can compute any partial recursive function of one variable. It starts with the argument in a counter, and (if it halts) leaves the answer in a counter.*

Definition 2.11 (E0L System). $G = (V, T, \omega, R)$ is a construct where V is the alphabet, $T \subseteq V$ is the terminal alphabet, $\omega \in V^*$ is the axiom, and R is a finite set of rules of the form $a \to v$ with $a \in V$ and $v \in V^*$ such that for each $a \in V$ there is at least one rule $a \to v$ in R. For $w_1, w_2 \in V^*$, we say that $w_1 \Rightarrow w_2$ if $w_1 = a_1 \dots a_n$, $w_2 = v_1 \dots v_n$ for $a_i \to v_i \in R$, $1 \leq i \leq n$. The generated language is $L(G) = \{x \in T^* \mid \omega \Rightarrow^* x\}$.

Definition 2.12 (ET0L System). $G = (V, T, \omega, R_1, \dots R_n)$ is a construct such that each (V, T, ω, R_i) is an E0L system; each R_i is called a table, $1 \leq i \leq n$. The generated language is defined as $L(G) = \{x \in T^* \mid \omega \Rightarrow_{R_{j_1}} \cdots \Rightarrow_{R_{j_m}} w_m = x\}$, with $m \geq 0$, $1 \leq j_i \leq n$, $1 \leq i \leq m$. We denote by $PsET0L$ the families of languages generated by extended table 0L grammars.

2.4.2 Simple Mobile Membranes

The computational power of simple mobile membranes is treated in [108].

The family of all sets $Ps(\Pi)$ generated by systems of degree at most n using rules $\alpha \in \{levol, gevol, endo, exo\}$ is denoted by $PsMM(\alpha)$ If the number of membranes is not bounded, this is denoted by $PsMM_*(levol, endo, exo)$. Here *levol* and *gevol* represent local and global evolution, *endo* and *exo* represent endocytosis and exocytosis. The number of membranes does not increase during the computation, but it can decrease by sending membranes out of the skin.

The following result establishes a universality result using nine membranes and the operations of endocytosis and exocytosis:

Theorem 2.2 ([108]). $PsMM_9(endo, exo) = PsRE$.

A strengthening of the previous universality result is:

Corollary 2.1 ([108]). $PsMM_*(endo, exo) = PsMM_n(endo, exo) = PsMM_n(gevol,\ endo, exo) = PsMM_n(levol, endo, exo) = PsRE$, *for all* $n \geq 9$.

An improvement of Theorem 2.2 is:

Theorem 2.3 ([103]). $PsMM_4(gevol, endo, exo) = PsRE$.

We improve the previous result by decreasing the number of membranes to three.

Theorem 2.4. $PsMM_3(levol, endo, exo) = PsRE$.

Proof. Consider a matrix grammar $G = (N, T, S, M, F)$ in the improved strong binary normal form (hence with $N = N_1 \cup N_2 \cup \{S; \#\}$), having n_1 matrices of types (2) and (4) (that is, not used in the appearance checking mode), and n_2 matrices of type (3) (with appearance checking rules). Let $B^{(1)}$ and $B^{(2)}$ be the two objects in N_2 for which we have rules $B^{(j)} \to \#$ in matrices of M. The matrices of the form $(X \to Y, B^{(j)} \to \#)$ are labelled by m'_i, with $i \in lab_j$, for $j \in \{1,2\}$, such that lab_1, lab_2, and $lab_0 = \{1, \ldots, n_1\}$ are mutually disjoint sets.

We construct a mobile membrane system $\Pi = (V, H, \mu, w_1, w_2, w_3, R, 2)$ of degree three, where:

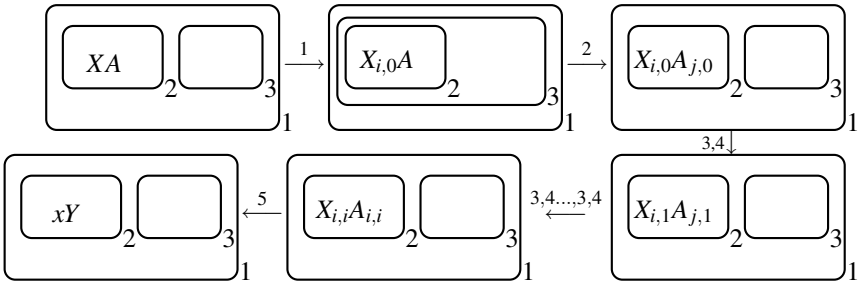
$V = N \cup \{X, X_{i,j} \mid X \in N_1, 1 \le i \le n_1, 0 \le j \le n_1\}$
$\quad \cup \{a, a' \mid a \in T\} \cup \{x \mid x \in (N_2 \cup T)^*\}$
$\quad \cup \{A, A_{i,j} \mid A \in N_2, 1 \le i \le n_1, 0 \le j \le n_1\}$
$H = \{1,2,3\}$
$\mu = \{(1,2);(1,3)\}$
$w_2 = XA$, where $(S \to XA)$ is the initial matrix of G
$w_h = \lambda$, for all $h \in \{1,3\}$

The set R of rules is constructed as follows:

(i) For each (nonterminal) matrix $m_i : (X \to Y, A \to x)$, $X, Y \in N_1$, $x \in (N_2 \cup T)^*$, $A \in N_2$, with $1 \le i \le n_1$, we consider the rules:

1. $[X]_2[\,]_3 \to [[X_{i,0}]_2]_3$ (endo)
2. $[[A]_2]_3 \to [A_{j,0}]_2[\,]_3$ (exo)
3. $[[X_{i,k} \to X_{i,k+1}]_2]_1$, $k < i$ (levol)
4. $[[A_{j,k} \to A_{j,k+1}]_2]_1$, $k < j$ (levol)
5. $[[A_{j,j}X_{i,i} \to xY]_2]_1$, $j = i$ (levol)
6. $[[A_{j,i}X_{i,i} \to \#]_2]_1$, $j > i$ (levol)
7. $[[A_{j,j}X_{i,j} \to \#]_2]_1$, $j < i$ (levol)

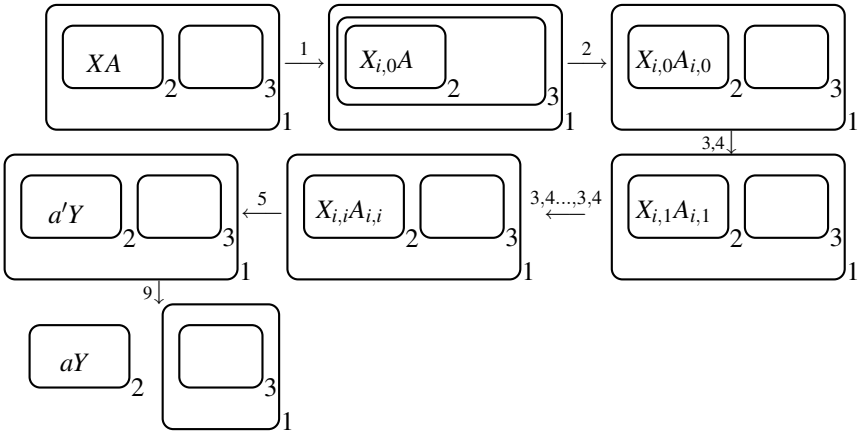
In the initial configuration, we have the objects X and A corresponding to the initial matrix in membrane 2. To simulate a matrix of the above type we start by applying the endocytosis rule 1, thus replacing X with $X_{i,0}$, followed by the exocytosis rule 2, thus replacing a single $A \in N_2$ with $A_{j,0}$. No other $A \in N_2$ can be replaced until membrane 2 enters membrane 3. Rule 3 (for X) and rule 4 (for A) are used to increment the second indices of X and A. This is done to check if the first indices of X and A are the same, and in this case to rewrite A according to the matrix m_i. Once the first indices are equal, rule 5 is applied to complete the simulation of matrix m_i. If the first indices of X and A are not the same, rule 6 (if the first indices of X is lower than the first indices of A) or rule 7 (if the first indices of X is bigger than the first indices of A) is applied, the computation is blocked without producing any output. We now illustrate the evolution of the configurations during one simulation of a type (2) matrix.



(ii) For a terminal matrix $m_i : (X \to a, A \to x)$, $X \in N_1$, $a \in T$, $A \in N_2$, $x \in T^*$, where $1 \leq i \leq n_1$, we use rules 1-7, where rule 5 is replaced by the rules:

8. $[a_{i,i}X_{i,i} \to a'Y]_1$ (levol)
9. $[[a']_2]_1 \to [a]_2[\,]_1$ (exo)

Observe that simulation of a type (4) matrix follows similar steps, except that we have an a in place of Y. During the finishing stages of a type (4) simulation, we use rule 8 to replace $a_{i,i}$ by a', and then to rewrite it to a when sending the membrane 2 out of the skin membrane, namely membrane 1. We now illustrate the evolution of the configurations during one simulation of a type (4) (terminal) matrix.

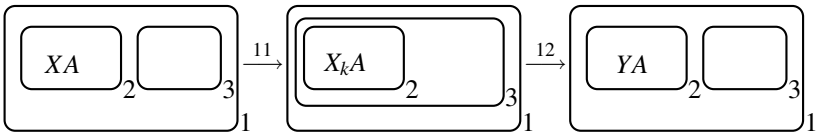


(iii) For each matrix $m'_i : (X \to Y, B^{(k)} \to \#)$, $X, Y \in N_1$, where $n_1 + 1 \leq j \leq n_1 + n_2$, $j \in lab_k$, $k = 1,2$, we consider the rules:

10. $[X]_2[\,]_3 \to [[X_k]_2]_3$, for $i \in lab_k$ (endo)
11. $[[X_k B^{(k)} \to \#]_2]_3$, $k = 1,2$ (levol)
12. $[[X_k]_2]_3 \to [Y]_2[\,]_3$, $k = 1,2$ (exo)

The simulation of matrices of type (3) begins with a rule of type 10. This is followed by a rule 11 in case $B^{(k)}$ exists, blocking membrane 2 inside membrane 3 and the computation stops without producing any output. If no $B^{(k)}$ exists, then rule 12 can be used to send out membrane 2, successfully completing the simula-

tion. We now illustrate the evolution of the configurations during one simulation of a type (3) matrix.



□

2.4.3 Enhanced Mobile Membranes

The operations governing the mobility of enhanced mobile membranes are endocytosis (endo), exocytosis (exo), enhanced endocytosis (fendo) and enhanced exocytosis (fexo). The interplay between these four operations is quite powerful, and the computational power of a Turing machine is obtained using twelve membranes without using the context-free evolution of objects [107].

The family of all sets $Ps(\Pi)$ generated by systems of degree at most n using rules $\alpha \subseteq \{exo, endo, fendo, fexo, pendo, pexo, rendo, rexo, rfendo, rfexo, cevol\}$ is denoted by $PsEMM_n(\alpha)$. Here *cevol* represents contextual evolution. The main results are the following.

Theorem 2.5 ([107]). $PsEMM_{12}(endo, exo, fendo, fexo) = PsRE$.

Theorem 2.6 ([107]). $PsEMM_3(cevol) = PsRE$.

Theorem 2.7 ([107]). $PsEMM_3(endo, exo) = PsEMM_3(fendo, fexo)$.

We improve the result of Theorem 2.5 as follows:

Theorem 2.8. $PsEMM_9(endo, exo, fendo, fexo) = PsRE$.

Proof. Consider a matrix grammar $G = (N, T, S, M, F)$ in the improved strong binary normal form (hence with $N = N_1 \cup N_2 \cup \{S; \#\}$), having n_1 matrices $m_1, \ldots, m_{n_1}$ of types (2) and (4) (that is, not used in the appearance checking mode), and n_2 matrices of type (3) (with appearance checking rules). The initial matrix is m_0, with $m_0 : (S \to XA)$. Let $B^{(1)}$ and $B^{(2)}$ be the two objects in N_2 for which we have rules $B^{(j)} \to \#$ in matrices of M. The matrices of the form $(X \to Y, B^{(j)} \to \#)$ are labelled by m'_i, $1 \leq i \leq n_2$ with $i \in lab_j$, for $j \in \{1,2\}$, such that lab_1, lab_2, and $lab_0 = \{1, 2, \ldots, n_1\}$ are mutually disjoint sets.

We construct a mobile membrane system $\Pi = (V, H, \mu, w_1, \ldots, w_9, R, 7)$ of degree nine, where:

$V = N \cup T \cup \{X'_{0i}, A'_{0i} \mid X \in N_1, A \in N_2, 1 \leq i \leq n_1\}$
$\quad \cup \{X_{ji}, A_{ji} \mid 0 \leq i, j \leq n_1\} \cup \{X_i^{(j)}, X_j \mid X \in N_1, j \in \{1,2\}, 1 \leq i \leq n_2\}$
$H = \{1, \ldots, 9\}$
$\mu = \{(1,7); (1,8); (1,9); (1,2); (2,3); (2,4); 2,5); (2,6)\}$

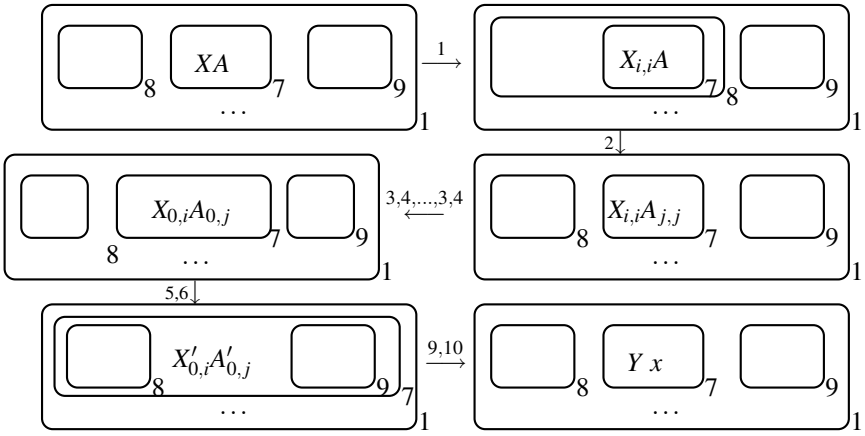
$w_7 = XA$, where $(S \to XA)$ is the initial matrix of G
$w_h = \lambda$, for all $h \in \{1,\ldots,9\}\backslash\{7\}$

The set R of rules is constructed as follows:

(i) For each (nonterminal) matrix $m_i : (X \to Y, A \to x)$, $X, Y \in N_1$, $x \in (N_2 \cup T)^*$, $A \in N_2$, with $1 \le i \le n_1$, we consider the rules:

1. $[X]_7[\ ]_8 \to [[X_{i,i}]_7]_8$ (endo)
2. $[[A]_7]_8 \to [A_{j,j}]_7[\ ]_8$ (exo)
3. $[X_{k,i}]_7[\ ]_9 \to [[X_{k-1,i}]_7]_9$, $k > 0$ (endo)
4. $[[A_{k,j}]_7]_9 \to [A_{k-1,j}]_7[\ ]_9$, $K > 0$ (exo)
5. $[\ ]_8[X_{0,i}]_7 \to [X'_{0,i}[\ ]_8]_7$ (fendo)
6. $[\ ]_9[A_{0,j}]_7 \to [A'_{0,j}[\ ]_9]_7$ (fendo)
7. $[\ ]_8[X_{0,j}]_7 \to [\#[\ ]_8]_7$ (fendo)
8. $[[A_{0,j}]_7]_9 \to [\#]_7[\ ]_9$ (exo)
9. $[X'_{0,j}[\ ]_8]_7 \to [\ ]_8[Y]_7$ (fexo)
10. $[A'_{0,j}[\ ]_9]_7 \to [\ ]_9[x]_7$ (fexo)

We now illustrate the evolution of the configurations during one simulation of a type (2) matrix, when $i = j$.



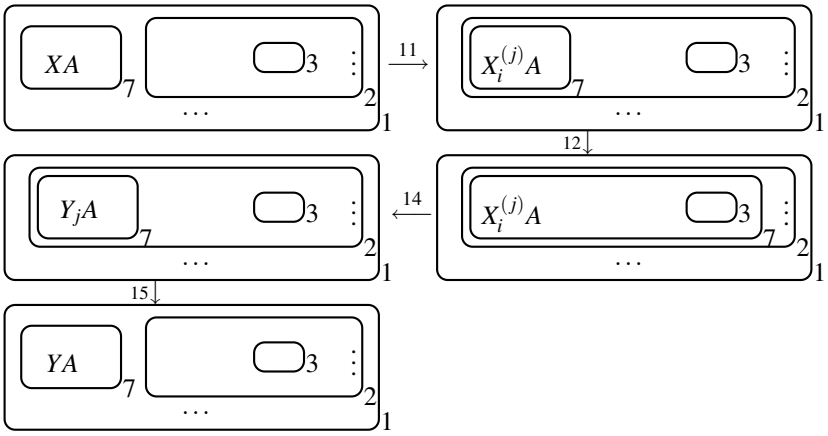
In the initial configuration, we have the objects X, A corresponding to the initial matrix in membrane 7. To simulate a matrix of type (2), we start by applying the endocytosis rule 1, thus replacing X with $X_{i,i}$, followed by the exocytosis rule 2, thus replacing a single $A \in N_2$ with $A_{j,j}$. Rule 3 (for X) and rule 4 (for A) are used to decrement the first indices of X and A. This is done to check if the indices of X and A are the same, and in this case to rewrite A according to the matrix m_i. By using fendo rules 5 and 6, membranes 8 and 9 enter membrane 7 replacing $X_{0,i}$ and $A_{0,j}$ with $X'_{0,i}$ and $A'_{0,j}$, respectively. This is then followed by rules 9 and 10, when membranes 8 and 9 exit membrane 7 by fexo rules replacing $X'_{0,i}$ and $A'_{0,i}$ with Y and x, respectively. If $i > j$, then we obtain $A_{0,j}$ before $X_{0,i}$. In this case, we have a configuration where membrane 7 is inside

membrane 9 containing $A_{0,j}$. Then rule 8 is used, replacing $A_{0,j}$ with #, and an infinite computation is obtained (rule 17). If $j > i$, then we obtain $X_{0,i}$ before $A_{0,j}$. In this case, we reach a configuration with $X_{0,i}A_{k,j}$, $k > 0$ in membrane 7, and membrane 7 placed inside membrane 1. Rule 3 cannot be used now, and the only possibility is to use rule 7, which leads to an infinite computation. Thus, if $i = j$, then we can correctly simulate a matrix of type (2).

(ii) For each matrix $m'_i : (X \to Y, B^{(k)} \to \#)$, $X, Y \in N_1$, $A \in N_2$, $n_1 + 1 \le j \le n_1 + n_2$, $j \in lab_k$, $k = 1,2$, we consider the rules:

11. $[X]_7[\,]_2 \to [[X_i^{(j)}]_7]_2$, $j = 1,2$ (endo)
12. $[\,]_{j+2}[X_i^{(j)}]_7 \to [X_i^{(j)}[\,]_{j+2}]_7$, $j = 1,2$ (fendo)
13. $[\,]_{j+4}[B^{(j)}]_7 \to [\#[\,]_{j+4}]_7$, $j = 1,2$ (fendo)
14. $[X_i^{(j)}[\,]_{j+2}]_7 \to [\,]_{j+2}[Y_j]_7$, $j = 1,2$ (fexo)
15. $[[Y_j]_7]_2 \to [Y]_7[\,]_2$, $j = 1,2$ (exo)

The simulation of matrices of type (3) begins with a rule of type 11. Inside membrane 2, rules 12 and 13 are used, and so membrane $(j+2)$ enters membrane 7, and membrane $(j+4)$ enters membrane 7 if the symbol $B^{(j)}$ is present. In this case, $B^{(j)}$ is replaced with #. Otherwise, membrane $(j+2)$ comes out of the membrane 7 replacing $X_i^{(j)}$ with Y_j. Then membrane 7 exits membrane 2, by replacing Y_j with Y thus successfully simulating a matrix of type (3). We now illustrate (only the membranes appearing in the rules 11-15 and $j = 1$) the evolution of the configurations during one simulation of a type (3) matrix.

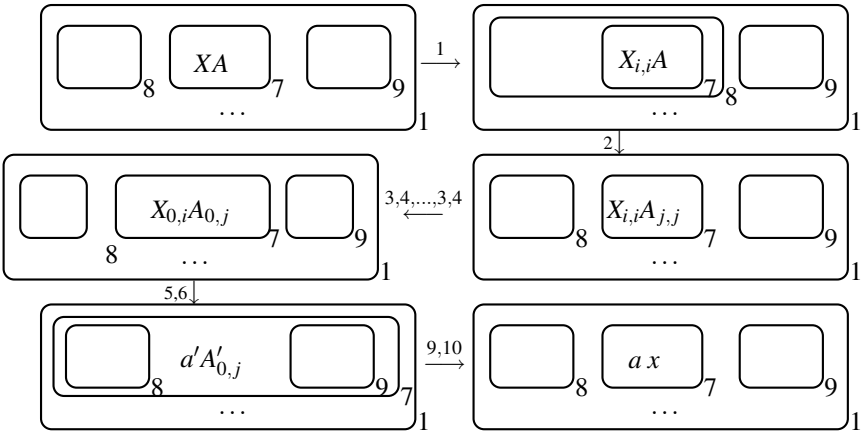


(iii) For a terminal matrix $m_i : (X \to a, A \to x)$, $X \in N_1$, $a \in T$, $A \in N_2$, $x \in T^*$, where $1 \le i \le n_1$:

16. $[[a']_7]_1 \to [a]_7[\,]_1$ (exo)
17. $[\,]_8[\#]_7 \to [\#[\,]_8]_7$ (fendo)
    $[\#[\,]_8]_7 \to [\,]_8[\#]_7$ (fexo)

Observe that simulation of a matrix of type (4) is similar to that of a matrix of type (2), except that we have an a' in place of Y in rule 9. During the finishing

stages of a matrix of type (4) simulation, we use rule 16 to replace a' with a when sending the membrane 7 out of the skin membrane. We now illustrate (only the membranes appearing in the rules 1-8,16-17) the evolution of the configurations during one simulation of a type (4) (terminal) matrix.



□

The family of all sets of numbers $\mathbf{N}(\Pi)$ which are obtained as a result of a halting computation by a P system Π with enhanced mobile membranes of degree at most n using rules $\alpha \subseteq \{exo, endo, fendo, fexo, pendo, pexo\}$, is denoted by $\mathbf{N}EMM_n(\alpha)$. In what follows we present the results obtained in [61].

Theorem 2.9. $\mathbf{N}EMM_5(endo, exo, fendo, fexo) = \mathbf{N}RE$.

Proof. We only prove the assertion $\mathbf{N}RE \subseteq \mathbf{N}EMM_5(endo, exo, fendo, fexo)$, and infer the other inclusion from the Church-Turing thesis. The proof is based on the observation that each set from $\mathbf{N}RE$ is the range of a recursive function. Thus, we will prove that for each recursively enumerable function $f : \mathbf{N} \to \mathbf{N}$, there is a Π with five membranes satisfying the following condition: For any arbitrary $x \in \mathbf{N}$, the system Π first "generates" a multiset of the form c^x and halts if and only if $f(x)$ is defined, and, if so, the result of the computation is $f(x)$.

In order to prove this assertion, we consider a register machine $\mathscr{M}$ with three registers, the last one being a special output register which is never decremented. Let there be a program P consisting of h instructions $l_1, \ldots, l_h$ which computes f. Let l_h correspond to the instruction HALT and l_1 be the first instruction. The input value x is expected to be in register 1 and the output value in register 3. Without loss of generality, we can assume that all registers other than the first one are empty at the beginning of a computation. We construct the membrane system

$\Pi = (V, \{0,1,2,3,4\}, \{(0,1);(0,2);(0,3);(0,4)\}, \emptyset, \{a_0\}, \{a_1\}, \{K_0\}, \emptyset, R, 3)$

with $V = \{l_i, l'_i, l''_i, L_i, L'_i \mid 1 \leq i \leq h\} \cup \{K_0, a_0, a_1, c\}$. The rules R are:

1. $[K_0]_3[\,]_1 \to [[K_0]_3]_1$ (endo)
   $[[\,]_3 a_0]_1 \to [\,]_3[a_0 c]_1$ (fexo)
   $[[K_0]_3]_1 \to [l_1]_3[\,]_1$ (exo)

Generation of c^x, the initial contents of register 1: Membrane 3 with K_0 enters membrane 1, and comes out each time adding a c to membrane 1. To terminate, K_0 is changed to l_1.

2. $[l_i]_3[\,]_1 \to [[l_j]_3]_1$ (endo)
   $[[\,]_3 a_0]_1 \to [\,]_3[a_0 c]_1$ (fexo)
3. $[l_i]_3[\,]_2 \to [\,[l_j]_3]_2$ (endo)
   $[[\,]_3 a_1]_2 \to [\,]_3[a_1 c]_2$ (fexo)
4. $[l_i]_3[\,]_4 \to [[l_j c]_3]_4$ (endo)
   $[[l_j]_3]_4 \to [l_j]_3[\,]_4$ (exo)
   Simulation of an increment instruction: $l_i : (inc(k), l_j)$. An endo and fexo rule given by rule 2 is used to increment counter 1: membrane 3 enters membrane 1 changing the instruction label, and comes out after adding a c to membrane 1. There is a similar rule (rule 3) between membranes 2, 3 for incrementing counter 2, with a_1 playing the role of a_0 for increment. To increment counter 3, we use the rules (rule 4) between membranes 3 and 4.
5. $[\,]_1[l_i]_3 \to [[\,]_1 L_i]_3$ (fendo)
6. $[[c]_1]_3 \to [\,]_1[\,]_3$ (exo)
   $[\,]_4[L_i]_3 \to [[\,]_4 L_i']_3$ (fendo)
7. $[[\,]_1 L_i']_3 \to [\,]_1[l_i'']_3$ (fexo)
8. $[[\,]_4 l_i'']_3 \to [\,]_4[l_i'']_3$ (fexo)
9. $[[\,]_4 L_i']_3 \to [\,]_4[L_i']_3$ (fexo)
10. $[L_i']_3[\,]_4 \to [[l_i']_3]_4$ (endo)
11. $[[l_i']_3]_4 \to [l_i']_3[\,]_4$ (exo)
    Simulation of a decrement instruction $l_i : (dec(1), l_i', l_i'')$. The simulation is initiated by rule 5: when membrane 1 enters membrane 3 by a fendo rule, l_i is replaced with L_i. If there is a c in membrane 1, then membrane 1 exits membrane 3 using rule 6; in parallel, membrane 4 enters membrane 3 using a fendo rule, replacing L_i with L_i'. If there were no c's in membrane 1, then membrane 1 will still be inside membrane 3, hence rule 7 is used, replacing L_i' with l_i'', a fexo rule. Membrane 4 exits membrane 3 irrespective of when membrane 1 exits membrane 3. If the symbol L_i' is present in membrane 3 after both membranes 1 and 4 exit it, then it means that there was a c in membrane 1; this L_i' is now replaced with l_i' using the endo, exo rules 10, 11. Rules for decrementing counter 2 are similar, with membrane 2 playing the role of membrane 1.

If $\mathscr{M}$ halts, then eventually we will have the instruction l_h in membrane 3 and membranes 1, 2 will have the final contents of counters 1, 2. Using the rule $[l_h]_3[\,]_4 \to [[\,]_3]_4$, the label l_h is erased. If we assign 3 as the output membrane, then its contents will be same as the contents of the output counter 3 at the end of a halting computation. □

Theorem 2.10. $\mathbf{N}EMM_{10}(fendo, fexo) = \mathbf{N}RE$.

Proof. The proof is done by simulating a matrix grammar $G = (N, T, S, M, F)$ with appearance checking in the strong binary normal form. We construct the P system

$$\Pi = (V, \{0, 1, 1', 2, 2', 3, 4, 5, 6, 7\}, \mu, \{XA\}, \emptyset, \ldots, \emptyset, R, 0)$$

with $\mu = \{(7,0);(7,3);(7,4);(7,5);(7,1);(7,1');(7,2);(7,2');(7,6)\}$.
$V = N \cup T \cup \{X_{ij}, A_{ij} \mid 0 \leq i,j \leq n_1, X \in N_1, A \in N_2\}$
$\cup \{X'_j, X''_j \mid X \in N_1, n_1+1 \leq j \leq n_1+n_2\} \cup \{Z, \dagger\}$.
Here, XA in membrane 0 corresponds to the initial matrix $(S \to XA)$. Membrane 0 is the output membrane. Let there be n_1 matrices of types (2), (4) in G labelled $1,\ldots n_1$ and n_2 matrices of type (3) in G labelled $n_1+1,\ldots,n_1+n_2$. The rules are

1. $[X]_0[\,]_3 \to [X_{ii}[\,]_3]_0$ (fendo)
   $[A[\,]_3]_0 \to [A_{jj}]_0[\,]_3$ (fexo)
2. $[X_{ki}]_0[\,]_4 \to [X_{k-1i}[\,]_4]_0$ (fendo)
   $[A_{kj}[\,]_4]_0 \to [A_{k-1j}]_0[\,]_4,\ k>0$ (fexo)
3. $[X_{0,i}]_0[\,]_5 \to [Y[\,]_5]_0$ (fendo)
   $[A_{0,j}[\,]_5]_0 \to [x]_0[\,]_5$ (fexo)
4. $[A_{kj}[\,]_5]_0 \to [\dagger]_0[\,]_5,\ k>0$ (fexo)
5. $[A_{0j}[\,]_4]_0 \to [\dagger]_0[\,]_4$ (fexo)
6. $[\dagger]_0[\,]_4 \to [\dagger[\,]_4]_0$ (fendo)
   $[\dagger[\,]_4]_0 \to [\dagger]_0[\,]_4$ (fexo)
   Simulation of a type (2) matrix $m_i : (X \to Y, A \to x)$. Rules 1 are used to remember the matrix m_i to be simulated. If X, A belong to the same matrix, then we obtain X_{ii} and A_{ii}. Rules 2 are then used to check if both X, A belong to the same matrix. If yes, then A_{0i} is generated in membrane 0 in the immediate next step after X_{0i}. This is followed by rule 3, by which X_{0i} and A_{0i} are replaced. In case rule 1 gives rise to X_{ii} and A_{jj} with $i \neq j$, then an infinite computation is triggered by rules 4, 5 and 6.
   For $i \in \{1,2\}$, and a matrix $m_j : (X \to Y, B^{(i)} \to \dagger)$ of type (3),
7. $[X]_0[\,]_i \to [X'_j[\,]_i]_0$ (fendo)
8. $[X'_j]_0[\,]_{i'} \to [X''_j[\,]_{i'}]_0$ (fendo)
   $[B^{(i)}[\,]_i]_0 \to [\dagger]_0[\,]_i$ (fexo)
9. $[X''_j[\,]_{i'}]_0 \to [Y]_0[\,]_{i'}$ (fexo)
10. $[Y[\,]_i]_0 \to [Y]_0[\,]_i$ (fexo)
11. $[\dagger]_0[\,]_i \to [\dagger[\,]_i]_0$ (fendo)
    $[\dagger[\,]_i]_0 \to [\dagger]_0[\,]_i$ (fexo)
    Simulation of a type (3) matrix $m_j : (X \to Y, B^{(i)} \to \dagger)$. The membrane labelled i enters membrane 0 replacing X with X'_j. This is followed by two parallel rules: membrane i' entering membrane 0 replacing X'_j with X''_j, and membrane i exiting membrane 0 in the presence of $B^{(i)}$. If $B^{(i)}$ is present, an infinite computation is triggered by rule 11. Membrane i' exits membrane 0 replacing X''_j with Y. If $B^{(i)}$ is absent, then membrane i will be inside membrane 0. In this case, it exits membrane 0 replacing Y with Y.
12. $[Z]_0[\,]_6 \to [[\,]_6]_0$ (fendo)
    $[A[\,]_6]_0 \to [\dagger]_0[\,]_6, A \in N_2$(fexo)
    Simulation of a type (4) matrix $m_j : (X \to \lambda, A \to x)$. This is done using the rules 1-6, replacing X with a new symbol Z. After this, we check if membrane 0 contains any non-terminals, and if so, an infinite computation is triggered by

rule 6. Otherwise, a halting computation is obtained, with membrane 0 containing the output. □

Theorem 2.11. $\mathbf{N}EMM_9(endo, exo, pendo) = \mathbf{N}RE$.

Proof. The proof is done by simulating a matrix grammar $G = (N, T, S, M, F)$ with appearance checking in the strong binary normal form. As in Theorem 2.10, let there be n_1 matrices of types 2, 4 and n_2 matrices of type 3. We construct the P system

$$\Pi = (V, \{0,1,2,3,4,5,6,7,8\}, \mu, w_0, \ldots, w_8, R, 0)$$

with $\mu = \{(8,0);(8,3);(8,4);(8,5);(8,6);(8,7);(6,1);(6,2)\}$

$w_0 = XA$, $w_1 = \alpha$, $w_2 = B^{(1)}B^{(2)}$, $w_3 = \ldots = w_8 = \emptyset$

$V = N \cup T \cup \{X_{ij}, A_{ij} \mid 0 \leq i, j \leq n_1, X \in N_1, A \in N_2\}$
$\cup \{X_j \mid n_1 + 1 \leq j \leq n_2\} \cup \{\alpha, \beta\} \cup \{Z, \dagger\}$.

Membrane 0 is the output membrane and XA corresponds to the initial matrix $(S \to XA)$. The rules are

1. $[X]_0[\,]_3 \to [[X_{ii}]_0]_3$ (endo)
   $[[A]_0]_3 \to [A_{jj}]_0[\,]_3$ (exo)
2. $[X_{il}]_0[\,]_4 \to [[X_{i-1l}]_0]_4$ (endo)
   $[[A_{jk}]_0]_4 \to [A_{j-1k}]_0[\,]_4$ for $i, j > 0$ (exo)
3. $[X_{0i}]_0[\,]_5 \to [[Y]_0]_5$ (endo)
   $[[A_{0i}]_0]_5 \to [x]_0[\,]_5$ (exo)
4. $[[A_{jk}]_0]_5 \to [\dagger]_0[\,]_5$ if $j > 0$ (exo)
   $[[A_{0k}]_0]_4 \to [\dagger]_0[\,]_4$ (exo)
5. $[\dagger]_0[\,]_5 \to [[\dagger]_0]_5$ (endo)
   $[[\dagger]_0]_5 \to [\dagger]_0[\,]_5$ (exo)
   Simulation of a type (2) matrix $m_i : (X \to Y, A \to x)$. Similar to Theorem 2.10.
6. $[X]_0[\,]_6 \to [[X_j]_0]_6$ (endo)
   $[X_jB^{(i)}]_0[B^{(i)}]_2 \to [[B^{(i)}]_2X_jB^{(i)}]_0$ (pendo)
   $[\alpha]_1[]_0 \to [[\alpha]_1]_0$ (endo)
7. $[[B^{(i)}]_2]_0 \to [\dagger]_2[]_0$ (exo)
   $[[\alpha]_1]_0 \to [\beta]_1[\,]_0$ (exo)
8. $[X_j]_0[\beta]_1 \to [[X_j]_0\beta]_1$ (endo)
   $[[X_j]_0]_1 \to [Y]_0[\,]_1$ (exo)
9. $[\beta]_1[Y]_0 \to [[\beta]_1Y]_0$ (pendo)
   $[[\beta]_1]_0 \to [\alpha]_1[\,]_0$ (exo)
10. $[[Y]_0]_6 \to [Y]_0[\,]_6$ (exo)
11. $[[\beta]_1]_6 \to [\dagger]_1[\,]_6$ (exo)
12. $[[\dagger]_i]_6 \to [\dagger]_i[\,]_6$ (exo)
    $[\dagger]_i[\,]_6 \to [[\dagger]_i]_6$ for $i = 1, 2$ (endo)
    Simulation of a type (3) matrix $m_j : (X \to Y, B^{(i)} \to \dagger)$. Membrane 0 enters membrane 6 replacing X with X_j. This is followed by the *pendo*, *endo* rules 6, by which membranes 1, 2 enter 0. Of course, membrane 2 enters only if there is a $B^{(i)}$ in membrane 0. The presence of a $B^{(i)}$ in membrane 0 triggers an infinite computation. Membrane 1 exits membrane 0 replacing α with β. This is followed by a *pendo* rule in 8, by which membrane 0 enters membrane 1. This

helps in replacing X_j with Y. Next, β is replaced with α by rule 9. If rule 10 is used before rule 9, we get an infinite computation. Membrane 0 comes out using rule 10, and another matrix can be simulated.

13. $[Z]_0[\ ]_7 \to [[\ ]_0]_7$ (endo)
$[[A]_0]_7 \to [\dagger]_0[\ ]_7, A \in N_2$ (exo)
Simulation of a type (4) matrix $m_j : (X \to \lambda, A \to x)$. This is similar to Theorem 2.10. □

Theorem 2.12. $\mathbf{N}EMM_8(fendo, fexo, pendo) = \mathbf{N}RE$.

Theorem 2.13. $\mathbf{N}EMM_{12}(endo, exo, fendo) = \mathbf{N}RE$.

Exercise 2.1. Prove Theorems 2.12 and 2.13 using the techniques from Theorems 2.9 and 2.10.

The following result can be observed from the above Theorems.

Theorem 2.14. $\mathbf{N}MAT \subseteq \mathbf{N}EM_5(endo, exo)$.

Theorem 2.15. *For all* $n \in \mathbf{N}$, *we have*

$$\mathbf{N}EM_n(rendo, rexo, rfendo, rfexo) \subseteq \mathbf{N}MAT_{ac} \subset \mathbf{N}RE.$$

Proof. Consider the membrane system $\Pi = (V, H, \mu, w_1, \ldots, w_n, i_0)$. We construct a random context matrix grammar $G = (N, T, S, M, F)$ with appearance checking but without λ-rules, and use the result $\mathbf{N}RCM(M, CF - \lambda, ac) = \mathbf{N}MAT_{ac} \subset \mathbf{N}RE$. In our construction, $F = \emptyset$, the appearance checking comes from the forbidden sets R used in the matrices.

Let $V_i = \{a_i \mid a \in V\}$, $V_i' = \{a_i' \mid a \in V\}, V_i'' = \{a_i'' \mid a \in V\}$ for $1 \leq i \leq n$, and $H = \{(i, j) \mid 1 \leq i, j \leq n, i \neq j\}$, and $Q = \{E_{j,\emptyset}, E'_{j,\emptyset}, N_{j,list}, N'_{j,list}, N'_{j,list\bar{w}} \mid 1 \leq j \leq n\}$. By $list$, w, and $list\overline{w}$ we denote strings of length at most n over the symbols $1, \ldots, n$, with no repetition of symbols. Then, $N = Q \cup \{C, E, U, Z, X\} \cup \mathscr{P}(H) \cup V_j \cup V_j' \cup V_j''$ for $1 \leq j \leq n$ is the set of non-terminals of G, and $T = V \cup \{Y\}$, is the set of terminal symbols, where Y is a new symbol. $\mathscr{P}(H)$ is the power set of all distinct pairs of labels of membranes.

The idea is to construct a grammar that not only simulates Π, but also keeps track of the membrane structure at each step. Let $E_{i,\emptyset}$ denote that membrane i is elementary, and $N_{j,list}$ denote that membrane j is non-elementary, and $list$ is the list of its children. One step of Π is simulated by G in several steps: G selects pairs of membranes one after the other in random fashion and checks if there is any applicable rule between them; if so, the rule is used, else the next pair of membranes is selected.

We start with the initial matrix

$$(S \to CZE_{1,\emptyset} \ldots N_{i,list} \ldots E_{n,\emptyset} w_1 w_j \ldots w_{i_0}, \emptyset, \emptyset)$$

where

1. C is a symbol to choose randomly a pair (i, j) of membranes whose interaction we are going to simulate,

2. $E_{i,\emptyset}, N_{j,list}$ gives the status of membranes in the initial configuration,
3. w_i is the content of membrane i in the initial configuration,
4. Z is a set that keeps track of the pairs of membranes that have interacted so far in the simulation of a step of Π. To simulate one step of Π, we have to check if all rules applicable to all pairs of membranes $(i, j), i \neq j$ have indeed been applied.

The content of the output membrane w_{i_0} is kept at the tail of the string. Z is initialized to $\emptyset$. We randomly keep selecting pairs of membranes to simulate a rule for a step of Π; this is continued until we have examined all possible pairs. The proof idea is essentially to update the symbols $E_{i,\emptyset}, N_{j,list}$ to reflect (i) whether i has been active in a step : we replace $E_{i,\emptyset}$ with $E'_{i,\emptyset}$ if i is elementary; (ii) if i is elementary and some membrane k entered i, we replace $E_{i,\emptyset}$ with $N'_{i,\bar{k}}$; (iii) if i is non-elementary and some membrane k entered i, we replace one of $N_{i,list}, N'_{i,list}$ with $N'_{i,list\bar{k}}$; (iv) if i is non-elementary, and some membrane k left membrane i, we replace one of $N_{i,list}, N'_{i,list}$ with $N'_{i,list-\{k\}}$. The prime on a symbol just tells us that the corresponding membrane has already been a part of a rule. In the case of $E'_{i,\emptyset}$, this means we can no longer utilize i, in the case of $N'_{j,list}$, since j is passive in the current step, j can be part of more than one rule. Let k_i be the number of symbols in membrane i in the initial configuration. This number remains constant, since the rules do not change the length of symbols evolving. We begin to write matrices that simulate Π and keep track of the membrane structure at every step. Assume that we decide to pick the pair of membranes $(1, i)$ first for simulation. Then we have the following matrices:

1. $((C \to C_{1,i}, Z \to Z \cup \{(1,i)\}, a_1 \to b'_1, E_{1,\emptyset} \to E'_{1,\emptyset}, N_{i,list} \to N'_{i,list\bar{1}},$
   $N_{j,list'} \to N'_{j,list'-\{1\}}), \emptyset, \{(1,i)\})$ if there is a *rendo* rule $[a]_1[\ ]_i \to [[b]_1]_i$, and $list'$ contains 1 and i (hence j is the parent of $1, i$). Here we assume i is non-elementary. In case i is elementary, a similar matrix with $N_{i,list}$ replaced with $E_{i,\emptyset}$ can be considered. In that case, we will have
   $((C \to C_{1,i}, Z \to Z \cup \{(1,i)\}, a_1 \to b'_1, E_{1,\emptyset} \to E'_{1,\emptyset}, E_{i,\emptyset} \to N'_{i,\bar{1}},$
   $N_{j,list'} \to N'_{j,list'-\{1\}}), \emptyset, \{(1,i)\})$.
2. $((C \to C_{1,i}, Z \to Z \cup \{(1,i)\}, a_1 \to b'_1, E_{i,\emptyset} \to E'_{i,\emptyset}, E_{1,\emptyset} \to N'_{1,\bar{i}},$
   $N_{j,list'} \to N'_{j,list'-\{i\}}), \emptyset, \{(1,i)\})$ if there is a $rfendo$ rule $[a]_1[\ ]_i \to [b[\ ]_i]_1$, and $list'$ contains 1 and i. Similarly, we can consider a matrix when membrane 1 is non-elementary.
3. $((C \to C_{1,i}, Z \to Z \cup \{(1,i)\}, a_1 \to b'_1, E_{1,\emptyset} \to E'_{1,\emptyset}, N_{i,list} \to N'_{i,list-\{1\}},$
   $N_{j,list'} \to N'_{j,list'\bar{1}}), \emptyset, \{(1,i)\})$ if there is a *rexo* rule $[[a]_1]_i \to [b]_1[\ ]_i$ and j is the parent of i; i.e, $list'$ contains i.
4. $((C \to C_{1,i}, Z \to Z \cup \{(1,i)\}, a_i \to b'_i, E_{1,\emptyset} \to E'_{1,\emptyset}, N_{i,list} \to N'_{i,list-\{1\}},$
   $N_{j,list'} \to N'_{j,list'\bar{1}}), \emptyset, \{(1,i)\})$ if there is a $rfexo$ rule $[a[\ ]_1]_i \to [\ ]_1[b]_i$ and j is the parent of i; i.e, $list'$ contains i.
5. We can consider $rexo, rfexo$ when i is a child of 1. This is similar to the above and we do not give details.

Now we consider cases when any rule cannot be used for a pair of membranes $(1,i)$. Recall that the number of objects in any membrane remains constant and there are k_i objects in membrane i. In the case when 1, i are adjacent, first check all symbols in membrane 1 to check that they are not involved in any rule. This is followed by checking all symbols of membrane i. A total of $(k_1+k_i)\cdot|R|$ steps are required where R is the set of rules of Π. Assume 1, i are siblings, 1 is elementary and i is non-elementary.

6. $((C \to C^1_{1,i}, Z \to Z \cup \{(1,i)\}, a_1 \to a_1'', \{N_{j,list'}, E_{1,\emptyset}, N_{i,list}\}, \emptyset)$ if $list'$ contains both 1 and i, and there is no *endo* rule involving a in membrane 1 between $1,i$.

   Continue incrementing the superscript $C^l_{1,i}$ to $C^{l+1}_{1,i}$ until we finish checking all symbols of membranes 1 and i.

7. $((C^l_{1,i} \to C^{l+1}_{1,i}, b_1 \to b_1''), \{N_{j,list'}, E_{1,\emptyset}, N_{i,list}\}, \emptyset)$ if $list'$ contains both 1 and i, and there is no *endo* rule involving b in membrane 1 between $1,i$, and $l < k_1$.
8. $((C^{k_1}_{1,i} \to C^{k_1+1}_{1,i}, a_i \to a_i''), \{N_{j,list'}, E_{1,\emptyset}, N_{i,list}\}, \emptyset)$ if $list'$ contains both 1 and i, and there is no $fendo$ rule involving a in membrane i between $1,i$.

   Once the superscript reaches k_1+k_i, we are done in checking. As $(1,i)$ is added to Z, we remember that this pair is checked already. Now, to make the symbols of membranes $1,i$ available for rules with other membranes, we unprime them.

9. $((C^{k_1+k_2}_{1,i} \to D_{1,i}, a_1'' \to a_1), \emptyset, \emptyset)$

10. $((D_{1,i} \to D_{1,i}, a_1'' \to a_1), \emptyset, \emptyset)$

    Rules 6-10 consider the case when membranes $1,i$ are siblings, but cannot participate in a rule with each other. We run through all elements in membranes $1,i$ to make sure of this. A similar set of rules can be written when 1 is non-elementary and i is elementary. If $1,i$ are siblings, and both are non-elementary, then we cannot use them in any mutual rule. In this case, it is enough to have a matrix that directly produces $D_{1,i}$ checking the presence of $N_{1,list}, N_{i,list'}$ (or their primed versions) and $N_{j,list''}$ (or its primed version) such that $list''$ contains both $1,i$. Another possibility to consider is when $1,i$ are such that one is a child of the other and we cannot employ any rules between them. This is done in a similar manner to rules 6-10. The third possibility is that $1,i$ are not adjacent. Then we have

11. $((C \to D_{1,i}, Z \to Z \cup \{(1,i)\}), \emptyset, \{N_{j,list}, N_{1,list'}, N_{i,list''} \mid j \neq 1,i\})$

    We check all the $N_{\alpha,list}$ symbols to ensure that $1,i$ are not adjacent. Thus, we need to put these in the forbidden symbols. Here $list$ contains membranes $1,i$, while $list'$ contains membrane i; $list''$ contains membrane 1.

The next pair of membranes will be chosen by replacing $D_{1,i}$ with an appropriate symbol (either $D_{r,s}$ or $C_{r,s}$ or $C^1_{r,s}$ as is the case); we must check that all double primed symbols are replaced - this is achieved by placing $V_1'' \cup V_i''$ in the forbidden list.

The above rules represent the first choice of a pair of membranes to begin a simulation. In general, for a pair of membranes r,s,

(a) For the case when a rule is applicable between the pair r,s, we will have a matrix

$$((D_{i,j} \to C_{r,s}, Z \to Z \cup \{(r,s)\}, rules\ \ as\ \ appropriate), \emptyset, \{(r,s)\} \cup V_i'' \cup V_j'')$$

or

$$((C_{i,j} \to C_{r,s}, Z \to Z \cup \{(r,s)\}, rules\ \ as\ \ appropriate), \emptyset, \{(r,s)\})$$

or

$$((C \to C_{r,s}, Z \to Z \cup \{(r,s)\}, rules\ \ as\ \ appropriate), \emptyset, \{(r,s)\})$$

The third matrix represents the case when we begin a round of simulation of Π. In all matrices, we will choose rule depending on the current relationship between r,s. The rules are based on the following guidelines:

- When r,s are siblings and not both are non-elementary: A matrix each for the cases when there is a $rendo, rfendo$ rule between r,s. Things to look out for: replace a_r or a_s as is the case by b'_r or b'_s (the prime on the symbols is so that they do not get used by another pair involving r or s); change the status of r, s: (i) if we have $E_{r,\emptyset}$, and if r is active in that step, indicate it by changing it into $E'_{r,\emptyset}$; update the parent information of r, and also the child list of the former parent of r, s; similarly if s is active - this covers the case when r,s are both elementary (ii) if r is passive and s is active, then we may have (a) $N_{r,list}$ (this indicates that so far nothing has entered r or left r) or (b) $N'_{r,list}$ (this indicates that something left r) or (c) $N'_{r,list\bar{w}}$ (this indicates that everything in w entered r previously in this step). Update these symbols appropriately: In cases (a), (b) we get $N'_{r,list\bar{s}}$, and in case (c), we get $N'_{r,list\bar{w}\bar{s}}$. The case when s is active and r is passive is similar. Update the parent j of r,s by deleting from j's list r or s: if we had $N_{j,list}$, it becomes $N'_{j,list-\{r\}}$ or $N'_{j,list-\{s\}}$, if we had $N'_{j,list}$, it becomes $N'_{j,list-\{r\}}$ or $N'_{j,list-\{s\}}$ and if we had $N'_{j,list\bar{w}}$, it becomes $N'_{j,list-\{r\}\bar{w}}$ or $N'_{j,list-\{s\}\bar{w}}$.
- When r is a parent of s and s is elementary: A matrix each for the cases of $rfexo, rexo$. Updates to be done similarly to the above case.
- When s is a parent of r and r elementary: A matrix each for the cases of $rfexo, rexo$. Updates to be done similarly to the above case.

(b) When there is no applicable rule between r,s:

Case (i): r,s are adjacent, and both are non-elementary. In this case, we have a matrix which directly gives $D_{r,s}$; Z is updated to contain (r,s). The permitting context is either $\{N_{r,list}, N_{s,list'}, N_{k,list''}\}$ such that $list''$ contains r,s, or is $\{N_{r,list}, N_{s,list'}\}$ such that $list'$ contains r or $list$ contains s. The permitting context can also be a set consisting of primed versions of these symbols; the only point to note is that we do not have a symbol α and its primed version α' together. The forbidding context contains (r,s).

Case (ii) r,s are adjacent but not both are non-elementary. We have a matrix of one of the forms

$$((C_{i,j} \to C^1_{r,s}, Z \to Z \cup \{(r,s)\}, rules\ \ as\ \ appropriate), A, \{(r,s)\})$$

or

$$((D_{i,j} \to C^1_{r,s}, Z \to Z \cup \{(r,s)\}, rules\ \ as\ \ appropriate), A, \{(r,s)\} \cup V''_i \cup V''_j)$$

or

$$((C \to C^1_{r,s}, Z \to Z \cup \{(r,s)\}, rules\ \ as\ \ appropriate), A, \{(r,s)\})$$

The last matrix represents the beginning of a round of simulation of Π. We are checking all forms of adjacency : siblings, parent or child. Here we consider $A = \{E_{r,\emptyset}, N_{s,list'}, N_{j,list''}\}$ or $\{N_{r,\emptyset}, E_{s,list'}, N_{j,list''}\}$ where $list''$ contains both r, s or $A = \{E_{r,\emptyset}, N_{s,list'}\}$ or $\{E_{s,\emptyset}, N_{r,list''}\}$ where $list'$ contains r and $list''$ contains s. For brevity, we explain only these cases for A here: the case when the members of A are primed also should be included - the only point to notice is that a symbol and its prime are not both together in A. The rules are chosen by the following guidelines: the permitting context A checks that r,s are indeed adjacent. For each symbol a_r, we check that there is no applicable rule with respect to s, and increment the counter till k_r. We have matrices for each of these, until we reach counter k_r. Note that since the number of rules in Π and the number of symbols in membrane r are both finite, this can be done. Then we check for each symbol of s that no rule is applicable till the counter reaches $k_r + k_s$. At this point, we switch to $D_{r,s}$ and unprime the symbols of r, s, and finally $D_{r,s}$ is replaced with some symbol that kick starts the simulation for another pair.

Case (iii) r,s are not adjacent. We check this is the case by having a matrix of the form

$$((C_{i,j} \to D_{r,s}, Z \to Z \cup \{(r,s)\}), \emptyset, \{(r,s)\} \cup B)$$

or

$$((D_{i,j} \to D_{r,s}, Z \to Z \cup \{(r,s)\}), \emptyset, \{(r,s)\} \cup V''_i \cup V''_j \cup B)$$

or

$$((C \to D_{r,s}, Z \to Z \cup \{(r,s)\}), \emptyset, \{(r,s)\} \cup B)$$

The last matrix represents the beginning of a round of simulation of Π. Let

$$B = \{N_{r,list}, N'_{r,list}, N'_{r,list\bar{w}}, N_{s,list'}, N'_{s,list'}, N'_{s,list'\bar{y}}, N_{j,list''}, N'_{j,list''}, N'_{j,list''\bar{x}}\}$$

where j is a membrane different from r,s, and $list$ contains s, $list'$ contains r, and $list''$ contains r,s. The matrix says it all; the absence of symbols of B, which spans all possibilities of r,s being adjacent, is enough. We add (r,s) to Z, so that we know that this pair has already been considered.

To terminate simulation of one step of Π, we do the following:

1. $((C_{p,q} \to E), \{Z \mid |Z| = \binom{n}{2} - n\}, \emptyset)$ or

   $((D_{p,q} \to E), \{Z \mid |Z| = \binom{n}{2} - n\}, V''_p \cup V''_q)$

   At some point, we check that all pairs of membranes have been examined. This is the case if the size of Z is $\binom{n}{2} - n$.

2. $((Z \to \emptyset), \{E\}, \emptyset)$, $((a'_i \to a_i), \{E\}, \emptyset)$ for all i,
3. $((E'_{i,\emptyset} \to E_{i,\emptyset}), \{E\}, \emptyset, \emptyset)$,
   In the presence of E, unprime all the primed symbols, replace Z with $\emptyset$
4. $((N'_{j,list\bar{w}} \to N_{j,list'}), \{E\}, \emptyset)$, where $list'$ is $list$ union w,
5. $((N'_{j,list} \to N_{j,list}), \{E\}, \emptyset)$ if $list \neq \emptyset$, $((N'_{j,list} \to E_{j,\emptyset}), \{E\}, \emptyset)$ if $list = \emptyset$
   Update the lists appropriately after a round of simulation
6. $((E \to C), \emptyset, \{$primed symbols like $a'_i, N'_{j,list}, E'_{j,\emptyset}, N'_{j,list\bar{w}}\})$
   Once all updates are completed, replace E with C to start the next round of simulation. The absence of primed symbols tells us that all updates are complete.

To terminate the simulation of G, we guess that Π has reached a halting configuration and validate our guess.

1. $((C \to U), \emptyset, \emptyset)$,
2. For $i \neq i_0$ and i is elementary: $((a_i \to X), W \cup \{U\}, \emptyset)$ where W is the set of all symbols $E_{i,\emptyset}$, $N_{j,list}$ such that $list$ contains i, and there are no *rexo* rules in Π between a in membrane i and membrane j, and for all $l \in list$, there are no $rendo, rfendo$ rules in Π between a in membrane i and membrane l. This can be checked in finite time: Π has a finite collection of rules, and the number of symbols in the current sentential form of G is finite : $(k_1 + \cdots + k_n) + n + 2$. Comparing each symbol a_i with the rules of Π pertaining to the appropriate membrane l in $list$ and membrane j is enough to apply this matrix.
3. i_0 is elementary: $((a_{i_0} \to a), W \cup \{U\}, \emptyset)$ where W is similar to above,
4. For $i \neq i_0$ and i is non-elementary: $((a_i \to X), W' \cup \{U\}, \emptyset)$ where W' is the set of all symbols $N_{i,list}, N_{k,list''}$ such that $list''$ contains i, and there are no $rfendo$ rules in Π between a in membrane i and membranes in $list''$, there are no $rfexo$ rules between a in membrane i and membranes in $list$. As in rule 1, this can be checked in finite time.
5. i_0 is non-elementary: $((a_{i_0} \to a), W' \cup \{U\}, \emptyset)$ where W' is similar to above,
6. $((U \to X), \emptyset, V_i)$,
   After all symbols a_i are replaced with X or a, we replace U with X.
7. $((N_{j,list} \to X), \{X\}, V_i \cup \{U\})$, $((E_{j,\emptyset} \to X), \{X\}, V_i \cup \{U\})$,
   $((Z \to X), \{X\}, V_i \cup \{U\})$
   After U has also been replaced, replace all the symbols $N_{j,list}$ and $E_{j,\emptyset}, Z$ with X.
8. $((X \to Y), \emptyset, N \backslash \{X\})$
   Replace all occurrences of X in the absence of all non-terminals different from X.

To terminate, we have to make sure that we have reached a halting computation. For this, at some non-deterministic point of time, we replace C with U. Then for all membranes other than the output, we replace the contents with symbols X after making sure that they have no rule to participate in given the current configuration. We replace symbols a_{i_0} of i_0 with a after ensuring the same. We are then left with a string of the form $XZN_{1,list} \ldots E_{n,\emptyset} X \ldots Xw$, where w is the contents of i_0. Then we also replace the symbols $N_{j,list}, E_{i,\emptyset}, Z$ with X. Then the number of symbols X we have is $\kappa = n + 2 + (k_1 + \cdots + k_n) - k_{i_0}$ which is finite. When we get a string of the form $X^\kappa w$, made over X and V, we replace all occurrences of X with Y. Since

the family MAT_{ac} is closed under quotient by letters, we do the quotient operation κ times on $L(G)$. Then we have $\mathbf{N}(\partial_Y^\kappa(L)) = \mathbf{N}(\Pi)$.

Note that if our guess is incorrect, i.e, if we replace C with U before Π has reached a halting configuration, we will not be able to proceed further, since the remaining matrices will not be applicable. In this case, we do not obtain a terminal string.

Note: In the above matrices, for ease of notation, we have kept the symbol Z in the rules and updated it, in an algorithmic style. In pure grammar notation, this can be thought of as replacing the current set with a larger set, and in the last step of a simulation, replacing a set of size $\binom{n}{2} - n$ with $\emptyset$. □

Theorem 2.16. *For all $n \in \mathbf{N}$, we have*

$$\mathbf{N}EM_n(rendo, rexo, rfendo, rfexo, pendo, pexo) \subseteq \mathbf{N}MAT_{ac} \subset \mathbf{N}RE.$$

Proof. A minor change to the proof of Theorem 2.15 will do. The only extra bookkeeping that needs to be done is to keep track of a $pendo, pexo$ rule that has happened. After choosing a pair (r,s) of membranes, we check if there are any $pendo, pexo$ rules which could be simulated as follows:

1. If r,s are siblings: If there exist symbols in membranes r,s such that r is entering s ($pendo$), then prime these symbols, prime the symbol $E_{r,\emptyset}$, change $N_{s,list}$ to $N_{s,list\bar{r}}$ or $E_{s,\emptyset}$ to $N_{s,\bar{r}}$, and continue as usual. The case of a $pexo$ is similar.
2. In case we have chosen r,s and there are no $pendo, pexo$ rules applicable to them, after checking that there are no $rendo, rexo, rfendo, rfexo$ rules applicable to them, do a check for non-applicability of $pendo, pexo$ rules. We have the symbol $D_{r,s}$ after checking for non-applicability of $rendo, rexo, rfendo, rfexo$. In case r, s are adjacent, this will look like

$$((D_{r,s} \to E_{r,s}, a_r'' \to a_r''', N_{j,list} \text{ or } N'_{j,list} \to \text{ itself}), \emptyset, B)$$

where $B = \{b_r'', c_s'' \mid$ there is a pendo rule involving a,b of membrane r and symbol c of membrane $s\}$. Continue this process by replacing a double primed symbol with a triple primed symbol, provided it is not part of a context that will enable a $pendo, pexo$ rule. $E_{r,s}$ is retained until this finishes. The exact description of B depends on the rules of Π. After replacing all symbols of r,s with triple primed symbols, replace $E_{r,s}$ with the next choice of membranes. Similarly check to ensure non-applicability of $pexo$ rules. □

2.4.4 Mutual Mobile Membranes

The family of all sets $Ps(\Pi)$ generated by systems of n mobile membranes using the mutual endocytosis rule $mendo$ and the mutual exocytosis rule $mexo$ is denoted by $PsMM_n(mendo, mexo)$. We denote by $PsRE$ the family of Turing computable sets of vectors generated by arbitrary grammars.

We prove that it is enough to consider only systems with three mobile membranes together with the operations of mutual endocytosis and mutual exocytosis to get the full computational power of a Turing machine. The proof is done in a similar manner to the proof of the computational universality of enhanced mobile membranes [107].

Theorem 2.17. $PsMM_3(mendo, mexo) = PsRE$.

Proof. It is proved in [90] that each recursively enumerable language can be generated by a matrix grammar in the strong binary normal form. We consider a matrix grammar $G = (N,T,S,M,F)$ in the strong binary normal form, having n_1 matrices $m_1,\ldots,m_{n_1}$ of types (2) and (4), and n_2 matrices of type (3) labelled by m'_i, $n_1+1 \le i \le n_1+n_2$ with $i \in lab_j$, for $j \in \{1,2\}$, such that lab_1, lab_2, and $lab_0 = \{1,2,\ldots,n_1\}$ are mutually disjoint sets. The initial matrix is $m_0 : (S \to XA)$.

We construct a system $\Pi = (V,H,\mu,w_1,w_2,w_3,R,2)$ with three mutual mobile membranes, where

$V = N \cup T \cup \{\alpha,\overline{\alpha},\beta,\overline{\beta}\}$
$\quad \cup \{X'_{0i},A'_{0i} \mid X \in N_1, A \in N_2, 1 \le i \le n_1\}$
$\quad \cup \{X_{ji},A_{ji} \mid 0 \le i,j \le n_1\}$
$\quad \cup \{\beta_j,\overline{\beta}_j,X_i^{(j)},Y_j \mid X \in N_1, j \in \{1,2\}, 1 \le i \le n_2\}$
$H = \{1,2,3\}; \qquad \mu = \{(1,2);(1,3)\}$
$w_1 = \emptyset; \qquad w_2 = \overline{\alpha}\,\overline{\alpha}_1\,\overline{\alpha}_2\beta\beta_1\beta_2 XA; \qquad w_3 = \alpha\alpha_1\alpha_2\overline{\beta}\,\overline{\beta}_1\,\overline{\beta}_2$

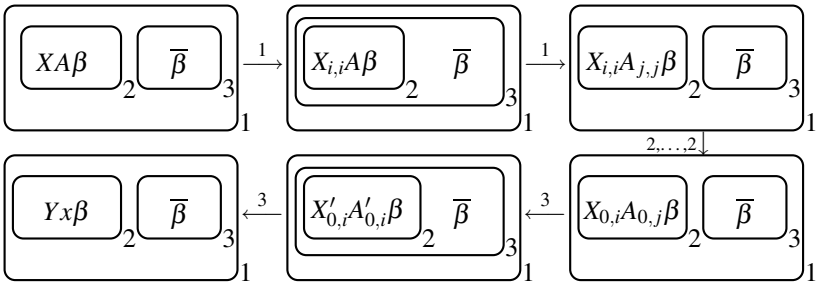
where $(S \to XA)$ is the initial matrix and the set R is constructed as follows:

(i) For each (nonterminal) matrix $m_i : (X \to Y, A \to x)$, $X,Y \in N_1$, $x \in (N_2 \cup T)^*$, $A \in N_2$, with $1 \le i \le n_1$, we consider the rules:

1. $[X\beta]_2[\overline{\beta}]_3 \to [[X_{ii}\beta]_2\overline{\beta}]_3$ (mendo)
   $[[A\beta]_2\overline{\beta}]_3 \to [A_{jj}\beta]_2[\overline{\beta}]_3$ (mexo)
2. $[X_{ki}\beta]_2[\overline{\beta}]_3 \to [[X_{k-1i}\beta]_2\overline{\beta}]_3$, $k > 0$ (mendo)
   $[[A_{kj}\beta]_2]\overline{\beta}]_3 \to [A_{k-1j}\beta]_2[\overline{\beta}]_3$, $k > 0$ (mexo)
3. $[X_{0i}A_{0j}\beta]_2[\overline{\beta}]_3 \to [[X'_{0i}A'_{0j}\beta]_2\overline{\beta}]_3$ (mendo)
   $[[X'_{0i}A'_{0j}\beta]_2\overline{\beta}]_3 \to [Yx\beta]_2[\overline{\beta}]_3$ (mexo)
4. $[[X_{ki}A_{0j}\beta]_2\overline{\beta}]_3 \to [\#\beta]_2[\overline{\beta}]_3$, $k > 0$ (mexo)
5. $[X_{0i}A_{kj}\beta]_2[\overline{\beta}]_3 \to [[\#\beta]_2\overline{\beta}]_3$, $k > 0$ (mendo)

By rule 1, membrane 2 enters membrane 3, replacing $X \in N_1$ with X_{ii}. A symbol $A \in N_2$ is replaced with A_{jj}, and membrane 2 comes out of membrane 3. The subscripts represent the matrices $m_i(m_j)$, $1 \le i,j \le n_1$ corresponding to which X, A have a rule. Next, rule 2 is used until X_{ii} and A_{jj} become X_{0i} and A_{0j}, respectively. If $i = j$, then we have X_{0i} and A_{0j} simultaneously in membrane 2. Then rule 3 is used, by which membrane 2 enters membrane 3 replacing X_{0i} and A_{0j} with X'_{0i} and A'_{0j}, while X'_{0i} and A'_{0j} are replaced with Y and x when membrane 2 exits membrane 3. If $i > j$, then we obtain A_{0j} before X_{0i}. In this case, we have a configuration where membrane 2 is inside membrane 3 containing A_{0j}. Rule 2 cannot be used now, and the only possibility is to use rule 4, replacing X_{ki} and A_{0j} with #, which leads to an infinite computation (due to rule 12). If $j > i$, then we

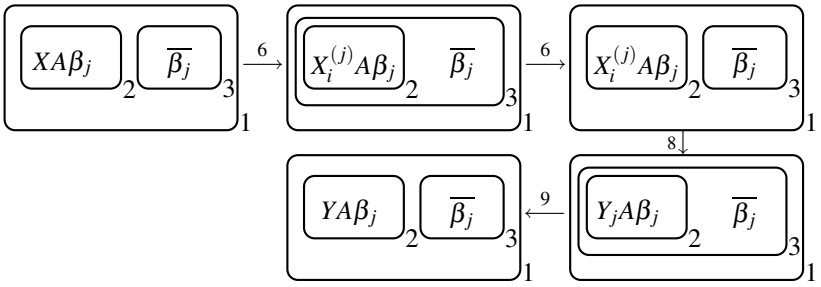
obtain X_{0i} before A_{0j}. In this case, we reach a configuration with $X_{0i}A_{kj}$, $k>0$ in membrane 2, and membrane 2 is in the skin membrane. Rule 2 cannot be used now, and the only possibility is to use rule 5, replacing X_{0i} and A_{kj} with #, which leads to an infinite computation. In this way, we correctly simulate a type (2) matrix whenever $i=j$. We now illustrate the evolution of the configurations during one simulation of a type (2) matrix, for $i=j$.



(ii) For each matrix $m'_i:(X\to Y,B^{(j)}\to\#)$, $X,Y\in N_1$, $B^{(j)}\in N_2$, $n_1+1\le i\le n_1+n_2$, $i\in lab_j$, $j=1,2$, we consider the rules:

6. $[X\beta_j]_2[\overline{\beta}_j]_3\to[[X_i^{(j)}\beta_j]_2\overline{\beta}_j]_3$ (mendo)
   $[[X_i^{(j)}\beta_j]_2\overline{\beta}_j]_3\to[X_i^{(j)}\beta_j]_2[\overline{\beta}_j]_3$ (mexo)
7. $[B^{(j)}\beta_j]_2[\overline{\beta}_j]_3\to[[\#\beta_j]_2\overline{\beta}_j]_3$ (mendo)
8. $[X_i^{(j)}\beta_j]_2[\overline{\beta}_j]_3\to[[Y_j\beta_j]_2\overline{\beta}_j]_3$ (mendo)
9. $[[Y_j\beta_j]_2\overline{\beta}_j]_3\to[Y\beta_j]_2[\overline{\beta}_j]_3$ (mexo)

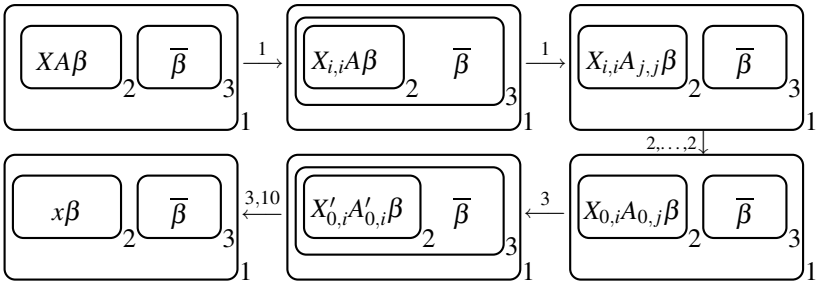
Membrane 2 enters membrane 3 by rule 6, and creates an object $X_i^{(j)}$ depending on whether it has the symbol $B^{(j)}$, $j=1,2$ associated with it, and then exits with the newly created object. Next, by rule 7, membrane 2 enters membrane 3 if the object $B^{(j)}$ is present, replacing it with #. If this rule is applied, membrane 2 exits membrane 3 by applying rule 12. Regardless of the existence of object $B^{(j)}$, membrane 2 enters membrane 3 replacing $X_i^{(j)}$ with Y_j. Membrane 2 exits membrane 3, replacing Y_j with Y, successfully simulating a matrix of type (3). We now illustrate the evolution of the configurations during one simulation of a type (3) matrix.



(iii) For a terminal matrix $m_i : (X \to \lambda, A \to x)$, $X \in N_1$, $A \in N_2$, $x \in T^*$, $1 \leq i \leq n_1$. We begin with rules 1-5 as before and simulate the matrix $(X \to Z, A \to x)$ in place of m_i, where Z is a new symbol.

10. $[\alpha]_3[Z\overline{\alpha}]_2 \to [\lambda\overline{\alpha}[\alpha]_3]_2$ (mendo)
11. $[A\overline{\alpha}[\alpha]_3]_2 \to [\alpha]_3[\#\overline{\alpha}]$ (mexo)
12. $[\#\beta]_2[\overline{\beta}]_3 \to [[\#\beta]_2\overline{\beta}]_3$ (mendo)
    $[[\#\beta]_2\overline{\beta}]_3 \to [\#\beta]_2[\overline{\beta}]_3$ (mexo)

Now we use rule 10 to erase the symbol Z while membrane 3 enters membrane 2. This is followed by rule 11 if there are any more symbols $A \in N_2$ remaining, in which case they are replaced with the trap symbol #. An infinite computation is obtained if the symbol # is present in membrane 2. It is clear that if the computation proceeds correctly, then membrane 2 contains a multiset of terminal symbols $x \in T^*$. In this way we can conclude that $Ps(\Pi)$ equals $Ps(L(G))$. We now illustrate the evolution of the configurations during one simulation of a type (4) (terminal) matrix.



□

It is worth noting that three is the smallest number of membranes when using effectively the movement of membranes given by endocytosis and exocytosis.

It is reasonable to investigate whether we can obtain new computability results using parallel mechanisms instead of sequential mechanisms. For systems of mobile membranes using mutual endocytosis and mutual exocytosis, we get the same computation power, but the results can be obtained more efficiently using parallel mechanisms. The following proof links parallel systems of mutual mobile membranes to sequential register machines. The register machines work in a slow and biologically unrealistic way; the results show that it is possible to get similar results with parallel mechanisms (based on the Church-Turing thesis).

Considering the number of objects and reduction to a register machine, we prove that the family $\mathbf{N}RE$ of all sets of natural numbers generated by arbitrary grammars is the same as the family $\mathbf{N}MM_3(mendo, mexo)$ of all sets of natural numbers generated by systems with three mobile membranes using *mendo* and *mexo* rules. This is calculated by looking at the cardinality of the objects in a specified *output membrane* of the mutual mobile membrane system at the end of a halting configuration.

Theorem 2.18. $\mathbf{N}MM_3(mendo,mexo) = \mathbf{N}RE$.

Proof. In view of the Church-Turing thesis, only $\mathbf{N}RE \subseteq \mathbf{N}MM_3(mendo,mexo)$ has to be proved. The proof is based on the observation that each set from $\mathbf{N}RE$ is the range of a partial recursive function. Thus, we prove that for each partial recursive function $f : \mathbf{N} \to \mathbf{N}$, there is a mutual mobile membrane system Π with three membranes satisfying the following condition: for any arbitrary $x \in \mathbf{N}$, the system first "generates" a multiset of the form o_1^x and halts if and only if $f(x)$ is defined, and, if so, the result is $f(x)$.

In order to prove the assertion, using similar arguments as in Ibarra et al [96], we can assume that the output register is never decremented during computation. This happens without loss of generality. Let there be a program P consisting of h instructions $P_1,\ldots,P_h$ which computes f. Let P_h correspond to the instruction HALT and P_1 be the first instruction. The input value x is expected to be in register 1 and the output value in register 3.

We construct a mutual mobile membrane $\Pi = (V,H,\mu,w_0,w_I,w_{op},R,I)$:

$V = \{s\} \cup \{o_r \mid 1 \le r \le 3\} \cup \{P_k, P'_k \mid 1 \le k \le h\} \cup \{\beta,\overline{\beta},\gamma,\overline{\gamma}\}$
$\quad \cup \{\beta_r \mid 1 \le r \le 3\}$
$H = \{0,I,op\} \quad \mu = \{(0,I);(0,op)\} \quad w_I = s\beta\overline{\gamma} \quad w_0 = \emptyset \quad w_{op} = \overline{\beta}\,\gamma$

(i) Generation of the initial contents x of register 1:

1. $[s\beta]_I[\overline{\beta}]_{op} \to [[s\beta]_I\overline{\beta}]_{op}$ (mendo)
   $[[s\beta]_I\overline{\beta}]_{op} \to [so_1\beta]_I[\overline{\beta}]_{op}$ (mexo)
2. $[[s\beta]_I\overline{\beta}]_{op} \to [P_1\beta]_I[\overline{\beta}]_{op}$ (mexo)

Rule 1 can be used any number of times, generating a number x (o_1^x) as the initial content of register 1. Rule 2 replaces s with the initial instruction P_1, and we are ready for the simulation of the register machine.

(ii) Simulation of an add rule $P_i = (INC(r), j)$, $1 \le r \le 3$, $1 \le i < h$, $1 \le j \le h$

3. $[P_i\beta]_I[\overline{\beta}]_{op} \to [[P_i\beta]_I\overline{\beta}]_{op}$ (mendo)
4. $[[P_i\beta]_I\overline{\beta}]_{op} \to [P_jo_r\beta]_I[\overline{\beta}]_{op}$ (mexo)

Membrane I enters membrane op using rule 3, and then exits it by replacing P_i with P_jo_r (rule 4), thus simulating an add instruction.

(iii) Simulation of a subtract rule $P_i = (DEC(r), j, k)$, $1 \le r \le 3$, $1 \le i < h$, $1 \le j,k \le h$

5. $[[P_i\beta]_I\overline{\beta}]_{op} \to [P'_j\beta_r\beta]_I[\overline{\beta}]_{op}$ (mexo)
6. $[o_r\beta_r\beta]_I[\overline{\beta}]_{op} \to [[\beta]_I\overline{\beta}]_{op}$ (mendo), otherwise
   $[P'_j\beta_r\beta]_I[\overline{\beta}]_{op} \to [[P'_k\beta]_I\overline{\beta}]_{op}$ (mendo)
7. $[[P'_j\beta]_I\overline{\beta}]_{op} \to [P_j\beta]_I[\overline{\beta}]_{op}$ (mexo)
   $[[P'_k\beta]_I\overline{\beta}]_{op} \to [P_k\beta]_I[\overline{\beta}]_{op}$ (mexo)

To simulate a subtract instruction, we start with rule 3, with membrane I entering membrane op. Then rule 5 is used, by which P_i is replaced with $P'_j\beta_r$, and

membrane I exits membrane op. The newly created object β_r denotes the register which has to be decreased. If there is an o_r in membrane I, then by rule 6 the object o_r is erased together with β_r, and membrane I enters membrane op. This is followed by rule 7, where P'_j is replaced with P_j and membrane I is back inside the skin membrane. If there are no o_r's in membrane I, then by applying rule 6, P'_j together with β_r is replaced by P'_k. This is followed by rule 7, where P'_k is replaced with P_k and membrane I is inside the skin membrane, thus simulating a subtract instruction.

(iv) Halting:

8. $[\gamma]_{op}[P_h\overline{\gamma}]_I \rightarrow [[\gamma]_{op}\overline{\gamma}]_I$ (mendo)

To halt the computation, the halt instruction P_h must be obtained. Once we obtain P_h in membrane I, membrane op enters membrane I and the computation stops (rule 8). When the system halts, membrane I contains only o_3's, the content of register 3. □

This result reveals a different technique in proving the computational power of a system with three mutual mobile membranes.

There are many families of languages included in RE. Although Theorem 2.18 is valid for all of them, these families can have particular sets of rules simulating them. We exemplify this aspect by an effective construction of a system with three mutual membranes able to simulate an ET0L system in the normal form.

In order to get the power of an ET0L system by using the operations of mutual endocytosis and mutual exocytosis, we need only three membranes.

Proposition 2.1. $PsET0L \subseteq PsMM_3(mendo, mexo)$.

Proof. In what follows, we use the following normal form: each language $L \in ET0L$ can be generated by $G = (V, T, \omega, R_1, R_2)$. Moreover, from [141], any derivation starts with several steps of R_1, then R_2 is used exactly once, and the process is iterated; the derivation ends by using R_2.

Let $G = (V, T, \omega, R_1, R_2)$ be an ET0L system in the normal form. We construct the mutual mobile membrane system

$$\Pi = (V', H, \mu, w_0, w_1, w_2, R, 0)$$

as follows:

$V' = \{\dagger, \alpha, \overline{\alpha}, \beta, \overline{\beta}\} \cup \{\beta_i, \overline{\beta}_i \mid i = 1,2\} \cup V \cup V_i$, where $V_i = \{a_i \mid a \in V\}$, $i = 1,2$

$H = \{0,1,2\}$ $\quad \mu = \{(2,0);(2,1)\}$ $\quad w_0 = \omega\alpha\beta_1\overline{\beta}$ $\quad w_1 = \overline{\alpha}\beta\overline{\beta}_i$

Simulation of table R_i, $i = 1,2$

1. $[\beta_i]_0[\overline{\beta_i}]_1 \rightarrow [[\beta_i]_0\overline{\beta}_i]_1$ (mendo)
2. $[[a\beta_i]_0\overline{\beta}_i]_1 \rightarrow [w_i\beta_i]_0[\overline{\beta}_i]_1$, if $a \rightarrow w \in R_i$ (mexo)
3. $[\beta]_1[a\overline{\beta}]_0 \rightarrow [[\beta]_1\dagger\overline{\beta}]_0$ (mendo)
4. $[[a_i\beta_i]_0\overline{\beta}_i]_1 \rightarrow [a\beta_i]_0[\overline{\beta}_i]_1$ (mexo)
5. $[\beta]_1[a_i\overline{\beta}]_0 \rightarrow [[\beta]_1\dagger\overline{\beta}]_0$ (mendo)

6. $[[\beta_1\alpha]_0\overline{\alpha}]_1 \to [\beta_i\alpha]_0[\overline{\alpha}]_1$ (mexo)
   $[[\beta_2\alpha]_0\overline{\alpha}]_1 \to [\beta_1\alpha]_0[\overline{\alpha}]_1$ (mexo)
   $[[\beta_2\alpha]_0\overline{\alpha}]_1 \to [\alpha]_0[\overline{\alpha}]_1$ (mexo)
7. $[[\beta]_1\dagger\overline{\beta}]_0 \to [\beta]_1[\dagger\overline{\beta}]_0$ (mexo)
   $[\beta]_1[\dagger\overline{\beta}]_0 \to [[\beta]_1\dagger\overline{\beta}]_0$ (mendo)

In the initial configuration the string $\beta_1\omega$ is in membrane 0, where ω is the axiom, and β_1 indicates that table 1 should be simulated first. The simulation begins with rule 1: membrane 0 enters membrane 1. In membrane 1, the only applicable rule is 2, by which the symbols $a \in V$ are replaced by w_1 corresponding to the rule $a \to w \in R_1$. Rules 1 and 2 can be repeated until all the symbols $a \in V$ have been replaced according to a rule in R_1, thus obtaining only objects from the alphabet V_1. In order to keep track of which table R_i of rules is simulated, each rule of the form $a \to w \in R_i$ is rewritten as $a \to w_i$.

If any symbol $a \in V$ is still present in membrane 0, i.e., if some symbol $a \in V$ has been left out of the simulation, membrane 1 enters membrane 0, replacing it with the trap symbol † (rule 3), and this triggers a never ending computation (rule 7). Otherwise, rules 1 and 4 are applied as long as required until all the symbols of V_1 are replaced with the corresponding symbols of V. Next, if a symbol $a_1 \in V_1$ has not been replaced, membrane 1 enters membrane 0 and the computation stops, replacing it with the trap symbol † (rule 5), and this triggers a never ending computation (rule 7). Otherwise, we have three possible evolutions (rule 6):

(i) if β_1 is in membrane 0, then it is replaced by β_i, and the computation continues with the simulation of table i;
(ii) if β_2 is in membrane 0, then it is replaced by β_1, and the computation continues with the simulation of table 1;
(iii) if β_2 is in membrane 0, then it is deleted, and the computation stops.

It is clear that $Ps(\Pi)$ contains all the vectors in $Ps(L(G))$. □

Corollary 2.2. $PsE0L \subseteq PsMM_3(mendo, mexo)$.

We can interpret the multiset of objects present in the output membrane as a set of strings x such that the multiplicity of symbols in x is the same as the multiplicity of objects in the output membrane. In this way, the multiset of objects in the output membrane generates a language. For a system Π, let $L(\Pi)$ represent this language (all strings computed by Π), and let $LMM_n(\alpha)$ represent the family of languages $L(\Pi)$ generated by systems having $\leq n$ membranes, using a set of operations $\alpha \subseteq \{mendo, mexo\}$. We get the following result.

Lemma 2.1. $LMM_3(mendo, mexo) - ET0L \neq \emptyset$.

Proof. L=$\{x \in \{a,b\}^* \mid |x|_b = 2^{|x|_a}\} \notin ET0L$ [141]. We construct
$\Pi = (\{a, b, b', \dagger, \beta, \beta_1, \overline{\beta}, \overline{\beta}_1, \beta_2, \overline{\beta}_2\}, \beta_3, \overline{\beta}_3\}, \{0, \ldots, 2\}, [[\eta\ b\ in]_1[\overline{in}]_2]_0, R, 1)$
with rules as given below to generate L.

1. $[\ \beta\ ]_1\ [\ \overline{\beta}\ ]_2 \rightarrow [\ [\ \beta\ ]_1\ \overline{\beta}\ ]_2$, (mendo),
   $[\ [\ b\ \beta\ ]_1\ \overline{\beta}\ ]_2 \rightarrow [\ b'\ b'\ \beta\ ]_1 [\ \overline{\beta}\ ]_2$, (mexo),
   $[\ [\ \beta\ ]_1\ \overline{\beta}\ ]_2 \rightarrow [\ \beta\ ]_1\ [\ \overline{\beta}\ ]_2$, (mexo),
2. $[\ \beta\ ]_1\ [\ \overline{\beta}\ ]_2 \rightarrow [\ [\ a\ \beta_1\ ]\ \overline{\beta}_1\ ]_2$, (mendo),
3. $[\ [\ b\ \beta_1\ ]_1\ \overline{\beta}_1\ ]_2 \rightarrow [\ \dagger\ \beta_1\ ]_1\ [\ \overline{\beta}_1\ ]_2$, (mexo),
   $[\ \dagger\ \beta_1\ ]_1\ [\ \overline{\beta}_1\ ]_2 \rightarrow [\ [\ \dagger\ \beta_1\ ]_1\ \overline{\beta}_1\ ]_2$, (mendo),
   $[\ [\ \dagger\ \beta_1\ ]_1\ \overline{\beta}_1\ ]_2 \rightarrow [\ \dagger\ \beta_1\ ]_1\ [\ \overline{\beta}_1\ ]_2$, (mexo),
4. $[\ [\ \beta_1\ ]\ \overline{\beta}_1\ ]_2 \rightarrow [\ \beta_2\ ]_1\ [\ \overline{\beta}_2\ ]_2$, (mexo),
   $[\ b'\ \beta_2\ ]_1\ [\ \overline{\beta}_2\ ]_2 \rightarrow [\ [\ b\ \beta_2\ ]_1\ \overline{\beta}_2\ ]_2$, (mexo),
   $[\ [\ \beta_2\ ]_1\ \overline{\beta}_2\ ]_2 \rightarrow [\ \beta_2\ ]_1\ [\ \overline{\beta}_2\ ]_2$ (mexo)
5. $[\ \beta_2\ ]_1\ [\ \overline{\beta}_2\ ]_2 \rightarrow [\ [\ \beta_3\ ]_1\ \overline{\beta}_3\ ]_2$, (mendo),
6. $[\ [\ b'\ \beta_3\ ]_1\ \overline{\beta}_3\ ]_2 \rightarrow [\ \dagger\ \beta_3\ ]_1\ [\ \overline{\beta}_3\ ]_2$, (mexo),
   $[\ \dagger\ \beta_3\ ]_1\ [\ \overline{\beta}_3\ ]_2 \rightarrow [\ [\ \dagger\ \beta_3\ ]_1\ \overline{\beta}_3\ ]_2$, (mendo),
   $[\ [\ \dagger\ \beta_3\ ]_1\ \overline{\beta}_3\ ]_2 \rightarrow [\ \dagger\ \beta_3\ ]_1\ [\ \overline{\beta}_3\ ]_2$, (mexo),
7. $[\ [\ \beta_3\ ]_1\ \overline{\beta}_3\ ]_2 \rightarrow [\ ]_1\ [\ ]_2$, (mexo),
8. $[\ [\ \beta_3\ ]_1\ \overline{\beta}_3\ ]_2 \rightarrow [\ \beta\ ]_1\ [\ \overline{\beta}\ ]_2$, (mexo).

The system works as follows: Rule 1 is used to replace every b with $b'b'$. Rule 2 can be used at any moment to replace β and $\overline{\beta}$ with β_1 and $\overline{\beta}_1$ (guessing that all b's have been replaced) and also to create an object a. Rule 3 checks that every b has been replaced with $b'b'$, and if not an infinite computation is obtained. If there is no b then rule 4 replaces β_1 and $\overline{\beta}_1$ with β_2 and $\overline{\beta}_2$, and then is used to replace every b' with b. Rule 5 can be used at any moment to replace β_2 and $\overline{\beta}_2$ with β_3 and $\overline{\beta}_3$ (guessing that all b''s have been replaced). Rule 6 checks that every b' has been replaced with b, and if not an infinite computation is obtained. The computation can halt using rule 7, and can continue using rule 8. It is easy to see that membrane 1 contains strings of L at the end of a halting computation. □

Exercise 2.2. What is the minimum number of membranes used to get full computational power for the class of mutual mobile membranes, if instead of the mutual endocytosis rule

$$[uv]_h[\overline{u}v']_m \rightarrow [\ [w]_h w']_m$$

we consider the enhanced endocytosis rule

$$[v]_h[uv']_m \rightarrow [[w]_h w']_m ?$$

Perform the proof using either matrix grammars or register machines.

Exercise 2.3. Consider other combinations of rules from the simple, enhanced and mutual mobile membranes and find the minimal set of ingredients in order to obtain the computational power of a Turing machine.

2.4.5 Mutual Mobile Membranes with Objects on Surface

We explore now the computational power of systems of mutual mobile membranes with objects on surface using $(pino, exo)$ and $(phago, exo)$ as applicable pairs of

rules. The power of these classes was already investigated in [105]; we improve those results with respect to the number of membranes used in computation. A summary of the results (existing results, as well as new ones) is given in Table 2.2.

Table 2.2 Summary of Results

Operations	Number of membranes	Weights	RE	Ref.
Pino, exo	8	4,3	Yes	Theorem 6.1 [105]
Pino, exo	3	5,4	Yes	Theorem 2.19
Phago, exo	9	5,2	Yes	Theorem 6.2 [105]
Phago, exo	9	4,3	Yes	Theorem 6.2 [105]
Phago, exo	4	5,4	Yes	Theorem 2.20

In each of the combinations presented in Table 2.2, the rules *pino* and *phago* are used to increase the number of membranes, while rule *exo* is used to decrease the number of membranes. Thus we combine the rules *pino* and *phago* with *exo* just to balance the number of membranes.

The result of a computation is considered to be the multiplicity vector of the multiset representing the union of the multisets placed on all the membranes of a system. Thus, the result of a halting computation consists of all the vectors describing the multiplicity of objects from all the membranes (a non-halting computation provides no output). The number of objects on the right hand side of a rule is called its *weight*. The family of all sets $Ps(\Pi)$ generated by systems of mutual mobile membranes with objects on surface using at any moment during a computation at most n membranes, and any of the rules $r_1 \in \{pino, phago\}$ and $r_2 \in \{exo\}$ of weight at most r, s respectively, is denoted by $PsM^3OS_n(r_1(r), r_2(s))$.

In what follows, we study the computational power of the $(pino, exo)$ combination of operations, and prove their universality by using during the computation at most three membranes.

Theorem 2.19. *$PsRE = PsM^3OS_m(pino(r), exo(s))$, for all $m \geq 3$, $r \geq 5$, $s \geq 4$.*

Proof. The inclusion of $PsM^3OS_m(pino(r), exo(s))$ into $PsRE$ is assumed true by invoking the Church-Turing thesis. This implies that we have to prove only the inclusion $PsRE \subseteq PsM^3OS_3(pino(5), exo(4))$. For this, we construct a system Π of three mutual mobile membranes with objects on surface,

$$\Pi = (V, \{(3,1);(3,2)\}, A\overline{X}, \overline{A}X, \lambda, R).$$

The finite alphabet V of objects is defined as follows:

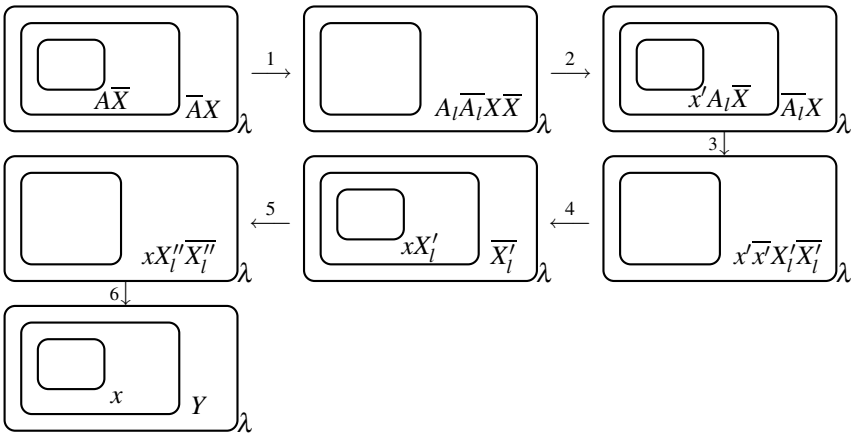
$$V = \{\beta, \overline{\beta}\} \cup \{X, \overline{X}, X'_l, \overline{X'_l}, X^{(j)}, \overline{X^{(j)}}, X_l^{(j)}, \overline{X_l^{(j)}}, X_l^{(j)'}, \overline{X_l^{(j)'}} \mid X \in N_1, 1 \leq l \leq n_1 + n_2, 1 \leq j \leq 2\}$$

The set of types of rules R is constructed as follows:

(i) For each (nonterminal) matrix $m_l : (X \to Y, A \to x)$, $X, Y \in N_1$, $x \in (N_2 \cup T)^*$, $A \in N_2$, $|x| \leq 2$, with $1 \leq l \leq n_1$, we consider the rules:

1. $[\,[\,]_{Au}]_{\overline{A}v} \rightarrow_m [\,]_{A_l\overline{A_l}uv}$ (exo)
2. $[\,]_{X\overline{X}uv} \rightarrow_m [\,[\,]_{x'X_l'u}]_{\overline{X_l'}v}$ (pino)
   (If $m_l : (X \rightarrow Y, A \rightarrow \alpha_1\alpha_2)$ then $x' = \alpha_1'\alpha_2$ or $x' = \alpha_1\alpha_2'$,
   and if $m_l : (X \rightarrow Y, A \rightarrow \alpha_1)$ then $x' = \alpha_1'$)
3. $[\,[\,]_{A_lu}]_{\overline{A_l}v} \rightarrow_m [\,]_{\overline{\alpha'}uv}$ (exo)
4. $[\,]_{\alpha'\overline{\alpha'}uv} \rightarrow_m [\,[\,]_{\alpha u}]_v$ (pino)
5. $[\,[\,]_{X_l'u}]_{\overline{X_l'}v} \rightarrow_m [\,]_{\overline{X_l''}X_l''uv}$ (exo)
6. $[\,]_{\overline{X_l''}X_l''uv} \rightarrow_m [\,[\,]_u]_{Yv}$ (pino)
7. $[\,[\,]_{\overline{X}u}]_{Xv} \rightarrow_m [\,]_{\beta\overline{\beta}uv}$ (exo)
8. $[\,]_{\beta\overline{\beta}uv} \rightarrow_m [\,[\,]_{\beta u}]_{\overline{\beta}v}$ (pino)
9. $[\,[\,]_{\beta u}]_{\overline{\beta}v} \rightarrow_m [\,]_{\beta\overline{\beta}uv}$ (exo)

We start the simulation of a type (2) matrix by using rule 1; A and $\overline{A}$ are replaced by A_l and $\overline{A_l}$, marking the beginning of the simulation. This is followed by rule 2, where X and $\overline{X}$ are replaced by x', X_l' and $\overline{X_l'}$. Next, we apply rule 3 to replace A_l and $\overline{A_l}$ by $\overline{\alpha'}$, in order to prevent replacing more A's from now on. In rule 4 we replace α' and $\overline{\alpha'}$ by α, while rule 5 replaces X_l' and $\overline{X_l'}$ by X_l'' and $\overline{X_l''}$, respectively. Rule 6 is used to replace X_l'' and $\overline{X_l''}$ by Y, thus successfully simulating a type (2) matrix and returning to the initial membrane structure. In case the corresponding symbol $A \in N_2$ is not present (we cannot apply rule 1), rule 7 introduces two symbols β and $\overline{\beta}$ which lead to an infinite computation (by using rules 8 and 9). We now illustrate the evolution of the configurations during one simulation of a type (2) matrix.

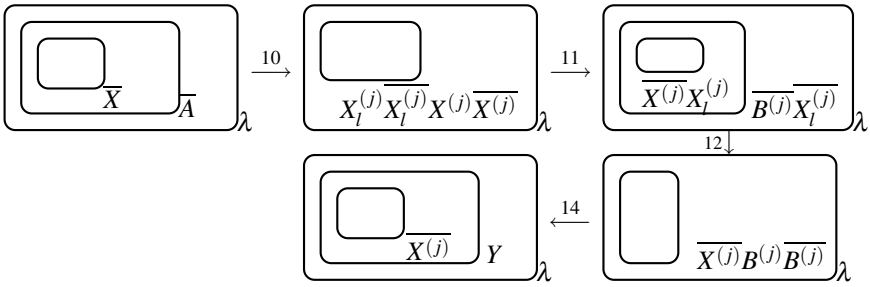


(ii) For each matrix $m_l' : (X \rightarrow Y, B^{(j)} \rightarrow \#)$, $X, Y \in N_1$, $A \in N_2$ and $B^{(j)} \rightarrow \# \in F$, where $n_1 + 1 \leq l \leq n_1 + n_2$, $l \in lab_j$, $j = 1, 2$, we consider the rules:

10. $[\,[\,]_{\overline{X}u}]_{Xv} \rightarrow_m [\,]_{X_l^{(j)}\overline{X_l^{(j)}}X^{(j)}\overline{X^{(j)}}uv}$ (exo)
11. $[\,]_{X^{(j)}\overline{X^{(j)}}uv} \rightarrow_m [\,[\,]_{\overline{X^{(j)}}u}]_{\overline{B^{(j)}}v}$ (pino)
12. $[\,[\,]_{B^{(j)}u}]_{\overline{B^{(j)}}v} \rightarrow_m [\,]_{\beta\overline{\beta}X^{(j)}uv}$ (exo)

13. $[\,[\,]_{X_l^{(j)}u}]_{\overline{X_l^{(j)}}v} \rightarrow_m [\,]_{B^{(j)}uv}$ (exo)
14. $[\,]_{B^{(j)}\overline{B^{(j)}}uv} \rightarrow_m [\,[\,]_u]_{Yv}$ (pino)

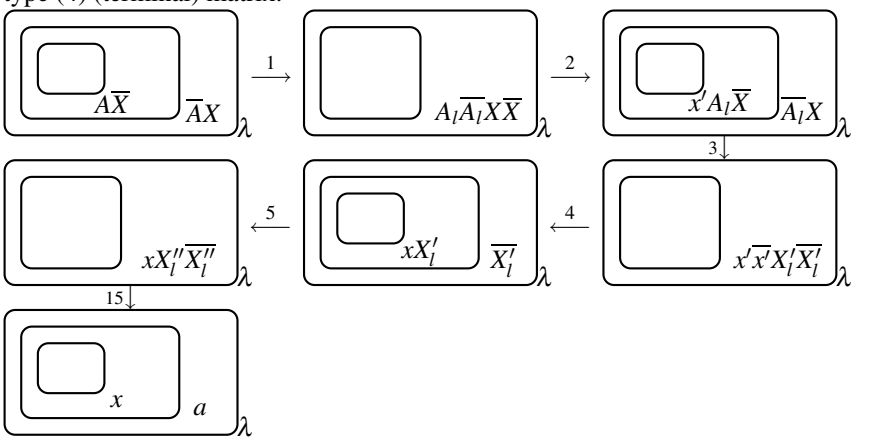
Rule 10 starts the simulation of a type (3) matrix by replacing $\overline{X}$ with $X_l^{(j)}X^{(j)}$ and X by $\overline{X_l^{(j)}}\,\overline{X^{(j)}}$, thereby remembering the index l of the matrix and the index j of the possibly present symbol $B^{(j)}$. This is followed by rule 11 in which $X^{(j)}$ and $\overline{X^{(j)}}$ are replaced by $\overline{X^{(j)}}$ and $\overline{B^{(j)}}$, respectively. At this step we need to check if the corresponding symbol $B^{(j)} \in N_2$ is present. If $B^{(j)}$ is present, rule 12 replaces it and its co-object $\overline{B^{(j)}}$ with $\beta\overline{\beta}$ together with $X^{(j)}$. In this case, by applying rule 11, we go to the configuration obtained before replacing $B^{(j)}$. Regardless of the presence of $B^{(j)}$, rule 13 is applied replacing $X_l^{(j)}$ and $\overline{X^{(j)}}$ by $B^{(j)}$. Rule 14 involves the creation of Y, thus successfully simulating a type (3) matrix and returning to the initial membrane structure. We now illustrate the evolution of the configurations during one simulation of a type (3) matrix.



(iii) For a terminal matrix $m_l : (X \rightarrow a, A \rightarrow x)$, $X \in N_1$, $a \in T$, $A \in N_2$, $x \in T^*$, $|x| \leq 2$, where $1 \leq l \leq n_1$, we consider the rule

15. $[\,]_{\overline{X_l''}X_l''uv} \rightarrow_m [\,[\,]_u]_{av}$ (pino)

We now illustrate the evolution of the configurations during one simulation of a type (4) (terminal) matrix.



By replacing rule 6 with rule 15 in the sequence 1-9, we correctly simulate a terminal matrix. The result of a correct simulation is the multiset of all symbols present on the surfaces of all membranes. □

We also study the computational power of the $(phago, exo)$ combination of operations and prove their universality by using during the computation at most four membranes. We initially consider a system of three membranes. Compared with Theorem 2.19, the higher number of membranes is related to the (use of) *phago* operation.

Theorem 2.20. *$PsRE = PsM^3OS_m(phago(r), exo(s))$, for all $m \geq 4$, $r \geq 5$, $s \geq 4$.*

Proof. The inclusion of $PsM^3OS_m(phago(r), exo(s))$ in $PsRE$ is assumed true by invoking the Church-Turing thesis. This implies that we have to prove only the inclusion $PsRE \subseteq PsM^3OS_4(phago(5), exo(4))$. For this we construct a system Π of three mutual mobile membranes with objects on surface

$$\Pi = (V, \{(3,1);(3,2)\}, A\overline{X}, \overline{A}X, \lambda, R)$$

The finite alphabet V of objects is defined as

$$V = \{\beta, \overline{\beta}\} \cup \{X, \overline{X}, X_l, \overline{X_l}, X_l', \overline{X_l'}, X^{(j)}, \overline{X^{(j)}}, X_l^{(j)}, \overline{X_l^{(j)}}, X_l^{(j)'}, \overline{X_l^{(j)'}} \mid X \in N_1, 1 \leq l \leq n_1 + n_2, 1 \leq j \leq 2\}$$

The set of types of rules R is constructed as follows:

(i) For each (nonterminal) matrix $m_l : (X \to Y, A \to x)$, $X, Y \in N_1$, $x \in (N_2 \cup T)^*$, $A \in N_2$, $|x| \leq 2$, with $1 \leq i \leq n_1$, we consider the rules:

1. $[\,]_{Au}[\,]_{\overline{A}v} \to_m [\,[\,[\,]_{A_l u}]_{\overline{X}}]_v$ (phago)
2. $[\,[\,]_{\overline{X}u}]_{Xv} \to_m [\,]_{\overline{A}_l X_l uv}$ (exo)
3. $[\,]_{A_l u}[\,]_{\overline{A}_l v} \to_m [\,[\,[\,]_{x' u}]_{\overline{X_l}}]_v$ (phago)
   (If $m_l : (X \to Y, A \to \alpha_1\alpha_2)$ then $x' = \alpha_1'\alpha_2$ or $x' = \alpha_1\alpha_2'$,
   and if $m_l : (X \to Y, A \to \alpha_1)$ then $x' = \alpha_1'$)
4. $[\,[\,]_{\overline{X}_l u}]_{X_l v} \to_m [\,]_{\overline{\alpha}' X_l' uv}$ (exo)
5. $[\,]_{\alpha' u}[\,]_{\overline{\alpha}' v} \to_m [\,[\,[\,]_{\alpha u}]_{\overline{X}_l'}]_v$ (phago)
6. $[\,[\,]_{\overline{X}_l' u}]_{X_l' v} \to_m [\,]_{Yuv}$ (exo)
7. $[\,]_{\overline{X}u}[\,]_{Xv} \to_m [\,[\,[\,]_{\overline{\beta}u}]_{\beta}]_v$ (phago)
8. $[\,[\,]_{\beta u}]_{\overline{\beta}v} \to_m [\,]_{\beta\overline{\beta}uv}$ (exo)
9. $[\,]_{\overline{\beta}u}[\,]_{\beta v} \to_m [\,[\,[\,]_{\beta u}]_{\beta}]_v$ (phago)

We start the simulation of a type (2) matrix by using rule 1; A and $\overline{A}$ are replaced by A_l and $\overline{X}$, marking the beginning of the simulation. This is followed by rule 2 replacing X and $\overline{X}$ by $\overline{A_l}$ and X_l, respectively. In rule 3, A_l is replaced by x' in order to prevent replacing more A's from now on, while $\overline{A_l}$ is replaced by $\overline{X_l}$. This is followed by rule 4 in which X_l and $\overline{X_l}$ are replaced by $\overline{\alpha'}$ and X_l', respectively. Rule 5 replaces α' and $\overline{\alpha'}$ by α and X_l'. Rule 6 involves the replacing of X_l' and $\overline{X_l'}$ by Y, thus successfully simulating a type (2) matrix and returning to the initial membrane structure. If the corresponding symbol $A \in N_2$ is not present (i.e., we

cannot apply rule 1), rule 7 introduces two symbols β and $\overline{\beta}$ which lead to an infinite computation (by using rules 8 and 9).

(ii) For each matrix $m'_l : (X \to Y, B^{(j)} \to \#)$, $X, Y \in N_1$, $A \in N_2$ and $B^{(j)} \to \# \in F$, where $n_1 + 1 \leq i \leq n_1 + n_2$, $i \in lab_j$, $j = 1, 2$, we consider the rules:

10. $[\,]_{\overline{X}u}[\,]_{Xv} \to_m [\,[\,[\,]_{X_l^{(j)} X^{(j)} u}]_{\overline{X_l^{(j)}}\,\overline{X^{(j)}}}]_v$ (phago)
11. $[\,[\,]_{X^{(j)}u}]_{\overline{X^{(j)}}v} \to_m [\,]_{\overline{B^{(j)}}\,\overline{X^{(j)}}uv}$ (exo)
12. $[\,]_{B^{(j)}u}[\,]_{\overline{B^{(j)}}v} \to_m [\,[\,[\,]_{\overline{\beta}u}]_{\beta X^{(j)}}]_v$ (phago)
13. $[\,]_{X_l^{(j)}u}[\,]_{\overline{X_l^{(j)}}v} \to_m [\,[\,[\,]_u]_{B^{(j)}}]_v$ (phago)
14. $[\,[\,]_{B^{(j)}u}]_{\overline{B^{(j)}}v} \to_m [\,]_{Yuv}$ (exo)

Rule 10 starts the simulation of a type (3) matrix by first replacing $\overline{X}$ and X by $X_l^{(j)}$, $X^{(j)}$, $\overline{X_l^{(j)}}$ and $\overline{X^{(j)}}$, thereby remembering the index l of the matrix and the index j of the possibly present symbol $B^{(j)}$. This is followed by rule 11 in which $X^{(j)}$ and $\overline{X^{(j)}}$ are replaced by $B^{(j)}$ and $\overline{X^{(j)}}$, respectively. At this step we need to verify if the corresponding symbol $B^{(j)} \in N_2$ is present. If $B^{(j)}$ is present, rule 12 replaces it and its co-object $\overline{B^{(j)}}$ with $\beta\overline{\beta}$ together with $X^{(j)}$. In this case, by applying rule 11, we go to the configuration obtained before replacing $B^{(j)}$. Regardless of the presence of $B^{(j)}$, rule 13 is applied; $X_l^{(j)}$ and $\overline{X^{(j)}}$ are replaced by $B^{(j)}$. Rule 14 involves the creation of Y, successfully simulating a type (3) matrix and returning to the initial membrane structure.

(iii) For a terminal matrix $m_l : (X \to a, A \to x)$, $X \in N_1$, $a \in T$, $A \in N_2$, $x \in T^*$, $|x| \leq 2$ where $1 \leq i \leq n_1$, we consider the rule

15. $[\,[\,]_{\overline{X}'_l u}]_{X'_l v} \to_m [\,]_{auv}$ (exo)

By replacing rule 6 with rule 15 in the sequence 1-9, we correctly simulate a terminal matrix. The result of a correct simulation is the multiset of all symbols present on the surfaces of all membranes. □

Exercise 2.4. Perform the proofs of this subsection using register machines instead of matrix grammars. Do the weights of the used operations differ?

2.5 Complexity of Mutual Mobile Membranes

Regarding the complexity aspects [126], polynomial time solutions to NP-complete problems in the framework of membrane computing are presented comprehensively in [131]. The authors of this survey use P systems with active membranes having associated electrical charges, membrane division and membrane creation. We present solutions to NP-complete problems by using systems of mutual mobile membranes that can perform only mobility and elementary division rules. In order to find such a solution, mutual mobile membranes are treated as deciding devices that respect the

following conditions: (1) all computations halt, (2) two additional objects *yes* and *no* are used, (3) only one of the objects *yes* and *no* appears in the halting configuration. The computation is accepting if the *yes* object is present in the halting configuration, and rejecting if the *no* object is present in the halting configuration.

A family of mutual mobile membrane systems $\{\Pi\}$ solves a decision problem if there is a member of the family to recognize every instance of the problem. To ensure that the construction algorithm of each member of the family does not increase the set of problems decided by the family, we require that the algorithm, is computable within certain restricted resources (time/space). When the algorithm maps an instance size to a membrane system that decides all instances of that length, then the algorithm is called a *uniformity condition*. The notion of *uniformity* was first introduced by Borodin [30] for boolean circuits. When the algorithm maps a single instance to a membrane system that decides that instance, then the algorithm is called a *semi-uniformity* condition. The notions of uniformity and semi-uniformity were first applied to membrane systems in [132].

2.5.1 SAT Problem

The SAT problem checks 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 $\neg x_k$. In this section, we propose a uniform polynomial time solution to the SAT problem using the operations of *mendo*, *mexo* and elementary division (for any instance of SAT we construct a system of mutual mobile membranes which solves it). Consider the formula $\varphi = C_1 \wedge C_2 \wedge \ldots C_m$, over the variables $\{x_1, \ldots, x_n\}$. Consider a system of mutual mobile membranes having the initial configuration

$$[[c_0\ \beta]_J[\overline{\beta}]_K[c\ \overline{d}]_L[g^{n-1}g_0]_1[a_1]_0]_2$$

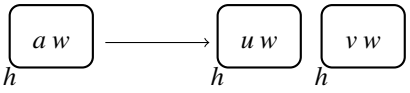
and working over the alphabet:

$$V = \{c, \overline{c}, d, \overline{d}, g, g_0, \beta, \overline{\beta}, yes, no\} \cup \{a_i, t_i, f_i \mid 1 \leq i \leq n\}$$
$$\cup \{\beta_i, \overline{\beta}_i \mid 1 \leq i \leq m\} \cup \{c_i \mid 0 \leq i \leq n+2m+1\}$$

In addition to mutual endocytosis and mutual exocytosis rules, we use elementary division rules to generate all the possible assignments. An elementary division rule has the form:

$$[a]_h \rightarrow [u]_h[v]_h, \text{ for } h \in H,\ a \in V,\ u, v \in V^* \qquad \text{(div)}$$

where a copy of each object from membrane h is placed inside the newly created membranes, except for object a which is replaced by the multisets of objects u and v. If w is the multiset of objects from h except the a object, then the rule is illustrated as:



The system of mutual mobile membranes solving the SAT problem uses the rules:

(i) $[a_i]_0 \to [t_i\, a_{i+1}]_0[f_i\, a_{i+1}]_0$, for $1 \le i \le n-1$ (div)
$[a_n]_0 \to [tn\, \beta_1]_0[f_n\, \beta_1]_0$ (div)
$[g]_1 \to [\,]_1[\,]_1$ (div)
$[g_0]_1 \to [\overline{\beta_1}]_1[\overline{\beta_1}]_1$ (div)
The first two rules create 2^n membranes labelled by 0 containing all the possible assignments over variables $\{x_1, \ldots, x_n\}$. In each membrane labelled by 0 is placed also a symbol β_1 . The next two rules create 2^n membranes labelled by 1 each containing an object $\overline{\beta_1}$. The symbols β_1 and $\overline{\beta_1}$ are used to determine in two steps which assignments are true for C_1.

(ii) $[t_j\, \beta_i]_0[\overline{\beta_i}]_1 \to [[t_j\, \beta_i]_0\overline{\beta_i}]_1$ (mendo)
$[[t_j\, \beta_i]_0\overline{\beta_i}]_1 \to [t_j\, \beta_{i+1}]_0[\overline{\beta_{i+1}}]_1$, $1 \le i \le m-1$, $1 \le j \le n$ (mexo)
(if clause C_i contains the literal x_j)
$[f_j\, \beta_i]_0[\overline{\beta_i}]_1 \to [[f_j\, \beta_i]_0\overline{\beta_i}]_1$ (mendo)
$[[f_j\, \beta_i]_0\overline{\beta_i}]_1 \to [f_j\, \beta_{i+1}]_0[\overline{\beta_{i+1}}]_1$, $1 \le i \le m-1$, $1 \le j \le n$ (mexo)
(if clause C_i contains the literal $\neg x_j$)
$[t_j\, \beta_m]_0[\overline{\beta_m}]_1 \to [[t_j\, \beta_m]_0\overline{\beta_m}]_1$ (mendo)
$[[t_j\, \beta_m]_0\overline{\beta_m}]_1 \to [t_j\, \overline{c}]_0[\overline{\beta_m}]_1$, $1 \le j \le n$ (mexo)
(if clause C_m contains the literal x_j)
$[f_j\, \beta_m]_0[\overline{\beta_m}]_1 \to [[f_j\, \beta_m]_0\overline{\beta_m}]_1$ (mendo)
$[[f_j\, \beta_m]_0\overline{\beta_m}]_1 \to [f_j\, \overline{c}]_0[\overline{\beta_m}]_1$, $1 \le j \le n$ (mexo)
(if clause C_m contains the literal $\neg x_j$)
If some assignments satisfy the clause C_i, $1 \le i < m$, then the objects β_i from the corresponding membranes 0 are replaced by β_{i+1}. The assignments from the membranes containing β_{i+1} satisfy the clauses $C_1, \ldots, C_i$, the object β_{i+1} marking the fact that in the next step the clause C_{i+1} is checked. If there exist assignments which satisfy all the clauses, then the membranes containing these assignments contain an object $\overline{c}$ after $n+2m$ steps.

(iii) $[c_i\, \beta]_J[\overline{\beta}]_K \to [[c_{i+1}\, \beta]_J\overline{\beta}]_K$ (mendo)
$[[c_i\, \beta]_J\overline{\beta}]_K \to [c_{i+1}\, \beta]_J[\overline{\beta}]_K$, $0 \le i \le n+2m$ (mexo)
$[[c_{n+2m+1}\, \beta]_J\overline{\beta}]_K \to [d\, \beta]_J[\overline{\beta}]_K$ (mexo)
$[c_{n+2m+1}\, \beta]_J[\overline{\beta}]_K \to [[c_{n+2m+1}\, \beta]_J\overline{\beta}]_K$ (mendo)
These rules trace the number of steps performed. If this number is greater than $n+2m+1$, then an object d is created, which will subsequently create an object *no*; $n+2m+1$ is determined by: generating space (n steps), verifying assignments ($2m$ steps), creating a *yes* object (1 step). An extra step can be performed, such that membrane J containing the object c_{n+2m+1} becomes sibling with membrane K, thus increasing the number of steps needed to create d to $n+2m+2$.

(iv) $[\overline{c}]_0[c]_L \to [[yes]_L]_0$ (mendo)
$[d]_J[\overline{d}]_L \to [[no]_J]_L$ (mendo)
A *yes* object is created whenever membrane L enters some membrane 0 in the $(2m+n+1)$-th step. If no membrane 0 contains an object $\overline{c}$, then a *no* object is created, in step $(2m+n+2)$ or $(2m+n+3)$, whenever membrane J enters membrane L. By applying one of these two rules, the other one cannot by applied

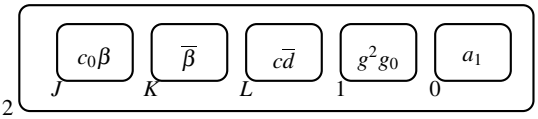
anymore, so at the end of the computation the system contains either an object *yes* or an object *no*.

The number of membranes in the initial configuration is 6, and the number of objects is $n+6$. The size of the working alphabet is $4n+4m+13$. The number of rules in the above system: $n+2$ rules of type (i), $4nm$ rules of type (ii), $n+2m+3$ rules of type (iii), and 2 rules of type (iv). Hence, the size of the constructed system of mutual mobile membranes is $\mathcal{O}(mn)$.

Example 2.2. Consider the SAT problem with $\phi = C_1 \wedge C_2 \wedge C_3$ and $X = \{x_1, x_2, x_3\}$, $C_1 = x_1 \vee \neg x_3$, $C_2 = \neg x_1 \vee \neg x_2$ and $C_3 = x_2$. In this case, $n = 3$, $m = 3$ and

$$[[c_0\ \beta]_J[\overline{\beta}]_K[c\ \overline{d}]_L[g^2 g_0]_1[a_1]_0]_2$$

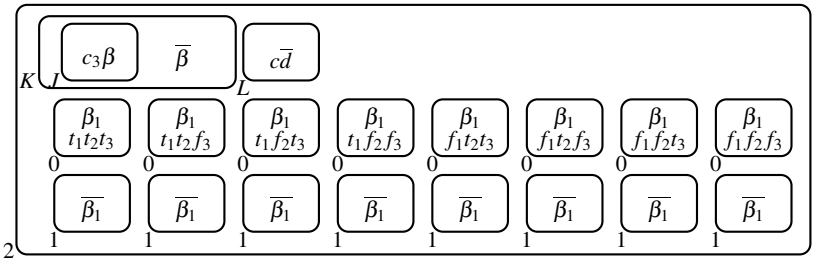
Graphically this is illustrated as:



The evolution of the system is described by the following steps, where $[w]_i^n$ stands for n membranes $[w]_i$. The working space is created in $n = 3$ steps leading from the initial configuration 1 to configuration 4:

1. $[[c_0\ \beta]_J[\overline{\beta}]_K[c\ \overline{d}]_L[g^2 g_0]_1[a_1]_0]_2$
2. $[[[c_1\ \beta]_J\overline{\beta}]_K[c\ \overline{d}]_L[g^2\overline{\beta_1}]_1^2[t_1\ a_2]_0[f_1\ a_2]_0]_2$
3. $[[c_2\ \beta]_J[\overline{\beta}]_K[c\ \overline{d}]_L[g\overline{\beta_1}]_1^4[t_1\ t_2\ a_3]_0[t_1\ f_2\ a_3]_0[f_1\ t_2\ a_3]_0[f_1\ f_2\ a_3]_0]_2$
4. $[[[c_3\ \beta]_J\overline{\beta}]_K[c\ \overline{d}]_L[\overline{\beta_1}]_1^8[t_1\ t_2\ t_3\ \beta_1]_0[t_1\ t_2\ f_3\ \beta_1]_0[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0$
$[f_1\ t_2\ t_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0[f_1\ f_2\ f_3\ \beta_1]_0]_2$

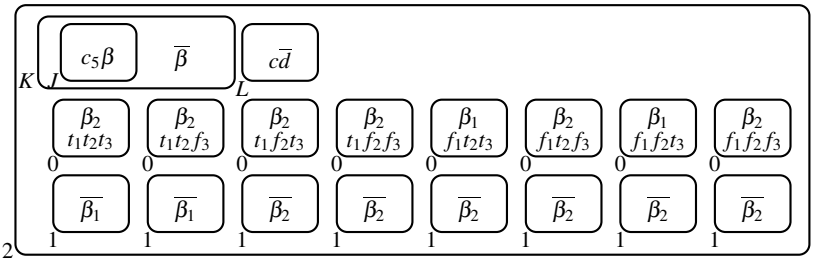
Graphically the working space is described by the following picture:



The next two steps mark the solutions of C_1 by replacing β_1 by β_2:

5. $[[c_4\ \beta]_J[\overline{\beta}]_K[c\ \overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_1}[t_1\ t_2\ t_3\ \beta_1]_0]_1[\overline{\beta_1}[t_1\ t_2\ f_3\ \beta_1]_0]_1[\overline{\beta_1}[t_1\ f_2\ t_3\ \beta_1]_0]_1$
$[\overline{\beta_1}[t_1\ f_2\ f_3\ \beta_1]_0]_1[\overline{\beta_1}[f_1\ t_2\ f_3\ \beta_1]_0]_1[\overline{\beta_1}[f_1\ f_2\ f_3\ \beta_1]_0]_1[f_1\ t_2\ t_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0]_2$
6. $[[[c_5\ \beta]_J\overline{\beta}]_K[c\ \overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ t_2\ t_3\ \beta_2]_0[t_1\ t_2\ f_3\ \beta_2]_0[t_1\ f_2\ t_3\ \beta_2]_0[t_1\ f_2\ f_3\ \beta_2]_0$
$[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0[f_1\ t_2\ t_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0]_2$

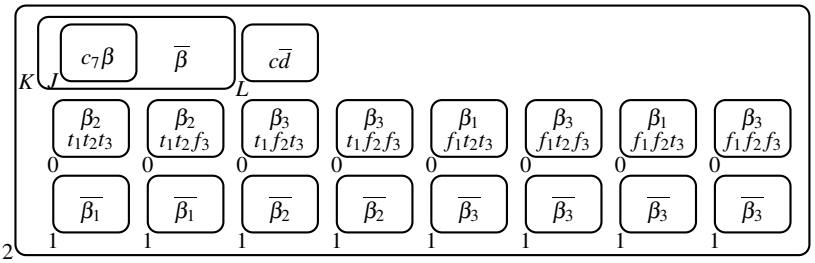
The new configuration is graphically represented by:



The next two steps mark the solutions of C_2 by replacing β_2 by $\underline{\beta_3}$:

7. $[[c_6\ \beta]_J[\overline{\beta}]_K[c\ \overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^2[\overline{\beta_2}[t_1\ f_2\ t_3\ \beta_2]_0]_1[\overline{\beta_2}[t_1\ f_2\ f_3\ \beta_2]_0]_1[\overline{\beta_2}[f_1\ t_2\ f_3\ \beta_2]_0]_1$
$[\overline{\beta_2}[f_1\ f_2\ f_3\ \beta_2]_0]_1[\underline{t_1\ t_2\ t_3\ \beta_2}]_0[t_1\ t_2\ f_3\ \beta_2]_0[f_1\ t_2\ t_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0]_2$
8. $[[[c_7\ \beta]_J\overline{\beta}]_K[c\ \overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^2[\overline{\beta_3}]_1^4[t_1\ f_2\ t_3\ \beta_3]_0[t_1\ f_2\ f_3\ \beta_3]_0[f_1\ t_2\ f_3\ \beta_3]_0$
$[f_1\ f_2\ f_3\ \beta_3]_0[t_1\ t_2\ t_3\ \beta_2]_0[t_1\ t_2\ f_3\ \beta_2]_0[f_1\ t_2\ t_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0]_2$

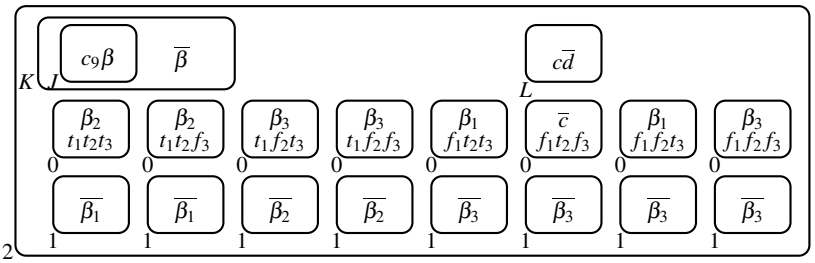
The new configuration is graphically represented by:



The next two steps mark the solutions of C_3 by replacing β_3 by $\overline{c}$:

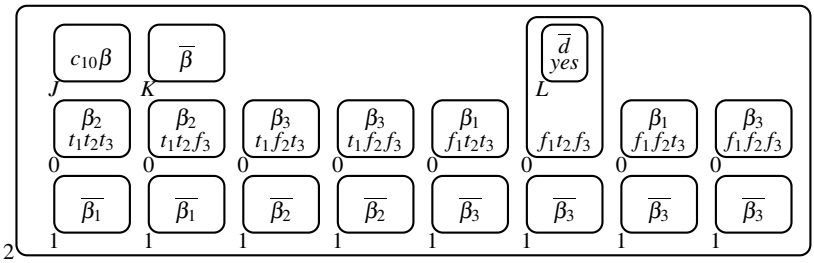
9. $[[c_8\ \beta]_J[\overline{\beta}]_K[c\ \overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^2[\overline{\beta_3}]_1^3[\overline{\beta_3}[f_1\ t_2\ f_3\ \beta_3]_0]_1[t_1\ f_2\ t_3\ \beta_2]_0[t_1\ f_2\ f_3\ \beta_2]_0$
$[f_1\ f_2\ f_3\ \underline{\beta_2}]_0[t_1\ t_2\ t_3\ \beta_2]_0[\underline{t_1\ t_2\ f_3}\ \beta_2]_0[t_1\ t_2\ t_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0]_2$
10. $[[[c_9\ \beta]_J\overline{\beta}]_K[c\ \overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^2[\overline{\beta_3}]_1^4[f_1\ t_2\ f_3\ \overline{c}]_0[t_1\ f_2\ t_3\ \beta_3]_0[t_1\ f_2\ f_3\ \beta_3]_0$
$[f_1\ f_2\ f_3\ \beta_3]_0[t_1\ t_2\ t_3\ \beta_2]_0[t_1\ t_2\ f_3\ \beta_2]_0[f_1\ t_2\ t_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0]_2$

The new configuration is graphically illustrated below, where we have placed the membrane labelled by L near the membrane labelled by 0 containing the symbol $\overline{c}$ to illustrate that an interaction is possible:



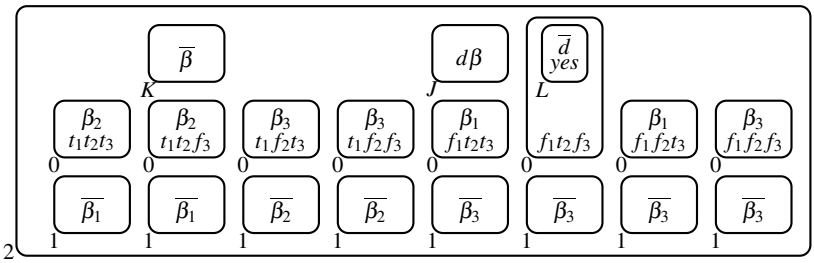
In the next step, an object *yes* is created and placed in membrane L, marking the fact that there exists an assignment such that the formula $(C_1 \wedge C_2 \wedge C_3)$ holds. The number of steps needed to create an object *yes* is $n+2m+1 = 3+6+1 = 10$.

11. $[[c_{10}\ \beta]_J[\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^2[\overline{\beta_3}]_1^4[f_1\ t_2\ f_3\ [yes\ \overline{d}]_L]_0[t_1\ f_2\ t_3\ \beta_3]_0[t_1\ f_2\ f_3\ \beta_3]_0$
$[f_1\ f_2\ f_3\ \beta_3]_0[t_1\ t_2\ t_3\ \beta_2]_0[t_1\ t_2\ f_3\ \beta_2]_0[f_1\ t_2\ t_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0]_2$
The new configuration is graphically illustrated as below:



An object d used to create an object *no* is created after performing the steps:

12. $[[[c_{10}\ \beta]_J\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^2[\overline{\beta_3}]_1^4[f_1\ t_2\ f_3\ [yes\ \overline{d}]_L]_0[t_1\ f_2\ t_3\ \beta_3]_0[t_1\ f_2\ f_3\ \beta_3]_0$
$[f_1\ f_2\ f_3\ \beta_3]_0[t_1\ t_2\ t_3\ \beta_2]_0[t_1\ t_2\ f_3\ \beta_2]_0[f_1\ t_2\ t_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0]_2$
13. $[[d\ \beta]_J[\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^2[\overline{\beta_3}]_1^4[f_1\ t_2\ f_3\ [yes\ \overline{d}]_L]_0[t_1\ f_2\ t_3\ \beta_3]_0[t_1\ f_2\ f_3\ \beta_3]_0$
$[f_1\ f_2\ f_3\ \beta_3]_0[t_1\ t_2\ t_3\ \beta_2]_0[t_1\ t_2\ f_3\ \beta_2]_0[f_1\ t_2\ t_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0]_2$
The new configuration is graphically illustrated below, where we place the membrane labelled by J near the membrane 0 containing membrane L to illustrate that an interaction between membranes J and L is not possible, and so the computation stops after $n+2m+3=12$ steps.



The fact that the computation ends in $n+2m+3$ steps is given by the fact that $n+2m$ is an odd number, and thus we had to perform an extra step before generating d from c_{n+2m+1}. If instead $n+2m$ is an even number, then d is created after $n+2m+2$ steps.

Exercise 2.5. Solve the SAT problem using other classes of mobile membranes.

2.5.2 2QBF Problem

In this section, we propose a polynomial time solution for solving satisfiability of 2QBF using mutual mobile membranes using the operations *mendo*, *mexo* and *div*. A quantified boolean formula is said to be in 2QBF if it is of the form $\varphi = \forall X \exists Y\, \psi$

or $\exists X \forall Y \psi$ where ψ is in CNF and X, Y partition the variables of ψ. By ψ is denoted the quantifier free part of φ. For 2QBF formulae of the form $\exists X \forall Y \psi$, satisfiability simplifies to the SAT problem $\exists X \psi'$ where ψ' is the CNF obtained from ψ by removing all occurrences of universal literals. *Hence we deal only with 2QBF of the form* $\varphi = \forall X \exists Y \psi$.

Consider the formula $\varphi = \forall X \exists Y \psi$ where $\psi = (C_1 \wedge C_2 \cdots \wedge C_m)$. ψ is a propositional logic formula in CNF. Let $X = \{x_1, \ldots, x_k\}$ and $Y = \{x_{k+1}, \ldots, x_n\}$, $X \cap Y = \emptyset$, and each C_i be a clause (disjunction of literals x_i or $\neg x_i$). Consider a system of mutual mobile membranes having the initial configuration

$$[[d\ c_0\ \beta]_J[\overline{\beta}]_K[\overline{d}]_L[g^{n-1}g_0]_1[a_1]_0]_2$$

and working over the alphabet:

$$V = \{b, c, d, \overline{d}, z, \overline{z}, no, yes, g, g_0, \alpha, \overline{\alpha}, \beta, \overline{\beta}\} \cup \{a_i, t_i, f_i \mid 1 \leq i \leq n\}$$
$$\cup \{\overline{t_i}, \overline{f_i} \mid 1 \leq i \leq k\} \cup \{\beta_i, \overline{\beta_i} \mid 1 \leq i \leq m\}$$
$$\cup \{c_i \mid 0 \leq i \leq n+2m+4k+4\} \cup \{d_i, d_i', d_i'', d_i''', \overline{d_i}''' \mid 1 \leq i \leq k\} \cup \{\overline{d_k}''\}$$

In addition to mutual endocytosis and mutual exocytosis rules, we use elementary division rules to generate all the possible assignments. The system of mutual mobile membranes solving the 2QBF problem uses the rules:

(i) $[a_i]_0 \to [t_i\ a_{i+1}]_0[f_i\ a_{i+1}]_0$, for $1 \leq i \leq n-1$ (div)
$[a_n]_0 \to [tn\ \beta_1]_0[f_n\ \beta_1]_0$ (div)
$[g]_1 \to [\,]_1[\,]_1$ (div)
$[g_0]_1 \to [\overline{\beta_1}]_1[\overline{\beta_1}]_1$ (div)
The first two rules generate 2^n membranes labelled by 0 containing all the possible assignments over variables $\{x_1, \ldots, x_n\}$. In each membrane labelled by 0 is placed also a symbol β_1 . The next two rules generate 2^n membranes labelled by 1 each containing an object $\overline{\beta_1}$. The symbols β_1 and $\overline{\beta_1}$ are used in mobility, where the membranes containing the object β_1 are the ones that move. These objects are used to determine in two steps which assignments are true for C_1.

(ii) $[t_j\ \beta_i]_0[\overline{\beta_i}]_1 \to [[t_j\ \beta_i]_0\overline{\beta_i}]_1$ (mendo)
$[[t_j\ \beta_i]_0\overline{\beta_i}]_1 \to [t_j\ \beta_{i+1}]_0[\overline{\beta_{i+1}}]_1$, $1 \leq i \leq m-1$, $0 \leq j \leq n$ (mexo)
(if clause C_i contains the literal x_j)
$[f_j\ \beta_i]_0[\overline{\beta_i}]_1 \to [[f_j\ \beta_i]_0\overline{\beta_i}]_1$ (mendo)
$[[f_j\ \beta_i]_0\overline{\beta_i}]_1 \to [f_j\ \beta_{i+1}]_0[\overline{\beta_{i+1}}]_1$, $1 \leq i \leq m-1$, $0 \leq j \leq n$ (mexo)
(if clause C_i contains the literal $\neg x_j$)
$[t_j\ \beta_m]_0[\overline{\beta_m}]_1 \to [[t_j\ \beta_m]_0\overline{\beta_m}]_1$ (mendo)
$[[t_j\ \beta_m]_0\overline{\beta_m}]_1 \to [t_j\ c]_0[\overline{\beta_m}]_1$, $0 \leq j \leq n$ (mexo)
(if clause C_m contains the literal x_j)
$[f_j\ \beta_m]_0[\overline{\beta_m}]_1 \to [[f_j\ \beta_m]_0\overline{\beta_m}]_1$ (mendo)
$[[f_j\ \beta_m]_0\overline{\beta_m}]_1 \to [f_j\ c]_0[\overline{\beta_m}]_1$, $0 \leq j \leq n$ (mexo)
(if clause C_m contains the literal $\neg x_j$)
If some assignments satisfy the clause C_i, $1 \leq i < n$, then the objects β_i from the corresponding membranes 0 are replaced by β_{i+1}. The object β_{i+1} marks the fact that the assignment satisfies clauses $C_1, \ldots, C_i$ and that in the next step the clause C_{i+1} is checked. If there exist assignments which verify all the clauses,

then the membranes containing these assignments contain an object $\overline{c}$ after $n+2m$ steps.

(iii) $[c_i\ \beta]_J[\overline{\beta}]_K \to [[c_{i+1}\ \beta]_J\overline{\beta}]_K$ (mendo)
$[[c_i\ \beta]_J\overline{\beta}]_K \to [c_{i+1}\ \beta]_J[\overline{\beta}]_K$, $0 \le i \le n+2m-1$ (mexo)
$[[c_{n+2m}\ \beta]_J\overline{\beta}]_K \to [c_{n+2m}\ \beta]_J[\overline{\beta}]_K$ (mexo)
$[c_{n+2m}d]_J[\overline{d}]_L \to [[c_{n+2m}d]_J\overline{d}]_L$ (mendo)
$[[c_{n+2m}d]_J\overline{d}]_L \to [c_{n+2m+3}]_J[d_1]_L$ (mexo)
These rules trace the number of steps performed. If this number is greater than $n+2m$, then an object d_1 is created in membrane L that marks the end of checking ψ. If there are solutions for ψ, the corresponding membranes contain the object c. The number $n+2m+3$ is determined by: generating space (n steps), verifying assignments ($2m$ steps), creating a d_1 object (2 steps) and eventually one step to perform the third rule if necessary.

(iv) $[d_i]_L \to [d_i']_L[d_i'']_L$, for $1 \le i \le k$ (div)
$[d_i']_L \to [\overline{t_i}^{2^{n-i}}\overline{d_i'''}\overline{z}]_L[\overline{f_i}^{2^{n-i}}\overline{d_i'''}\overline{z}]_L$, for $1 \le i \le k-1$ (div)
$[d_k']_L \to [\overline{t_k}^{2^{n-k}}d_k''']_L[\overline{f_k}^{2^{n-k}}d_k''']_L$ (div)
$[d_i'']_L \to [d_i''']_L[d_i''']_L$, for $1 \le i \le k-1$ (div)
$[d_k'']_L \to [\overline{d_k''}]_L[\overline{d_k''}]_L$ (div)
$[\overline{d_k'''}]_L \to [\overline{d_k''''}]_L[\overline{d_k''''}]_L$ (div)
$[d_i''']_L[\overline{d_i'''}]_L \to [[d_{i+1}]_L]_L$, for $1 \le i \le k-1$ (mendo)
$[t_ic]_0[\overline{t_i}]_L \to [[t_ic]_0\overline{t_i}]_L$, for $1 \le i \le k$ (mendo)
$[f_ic]_0[\overline{f_i}]_L \to [[f_ic]_0\overline{f_i}]_L$, for $1 \le i \le k$ (mendo)
Next, after finding the solutions of ψ, the $\forall$ part of the formula over variables $x_1,\ldots,x_k$ is checked. This amounts to checking if all the 2^k combinations of $t_i, f_i, 1 \le i \le k$ contain an object c. If so, then φ is true, and any last $n-k$ symbols will suffice for a solution. In order to check that all the 2^k combinations are in membranes containing a c object, membrane L is divided and a membrane structure is created in $3k$ steps. First, d_1 is replaced with d_1', d_1'' in two membranes. This is followed by the division of the membrane containing d_1' into two new membranes in which this object is replaced by a multiset containing $\overline{t_1}$, respectively $\overline{f_1}$. The membrane containing d_1'' is used to obtain two new membranes that are sent inside the membranes containing $\overline{t_1}$, respectively $\overline{f_1}$, in order to continue the construction until membranes containing $\overline{t_k}$, respectively $\overline{f_k}$, are obtained. In parallel, the membranes 0 containing an object c, enter the newly created structure.

(v) $[\overline{t_k}d_k''']_L[\overline{d_k'''}]_L \to [[b\ z]_L\overline{z}]_L$ (mendo)
$[\overline{f_k}d_k''']_L[\overline{d_k'''}]_L \to [[b\ z]_L\overline{z}]_L$ (mendo)
If at the end of the construction from step (iv) there exists an elementary membrane L containing an object $\overline{t_k}$ or $\overline{f_k}$, it means that not all possible assignments over the variables $x_1,\ldots,x_k$ are solutions, so the object b is created which will subsequently create an object *no*.

(vi) $[[z]_L\overline{z}]_L \to [z]_L[\overline{z}]_L$ (mexo)
$[[z]_L\overline{z}\,\overline{t_1}]_L \to [z\overline{\alpha}]_L[\overline{z}\,\overline{t_1}]_L$ (mexo)
$[[z]_L\overline{z}\overline{f_1}]_L \to [z\overline{\alpha}]_L[\overline{z}\overline{f_1}]_L$ (mexo)

If there exists a membrane L containing an object z together with the object *no*, then in k steps it reaches membrane 2, in order to deliver the answer inside membrane J.

(vii) $[c_i\ \beta]_J[\overline{\beta}]_K \rightarrow [[c_{i+1}\ \beta]_J\overline{\beta}]_K$ (mendo)
$[[c_i\ \beta]_J\overline{\beta}]_K \rightarrow [c_{i+1}\ \beta]_J[\overline{\beta}]_K$, $n+2m+3 \leq i \leq n+2m+4k+3$ (mexo)
In parallel, the counter in membrane J evolves until it reaches $n+2m+4k+4$. The extra $4k+1$ steps from $n+2m+3$ to $n+2m+4k+4$ is determined by: generating the structure starting from membrane L and movement of membranes 0 containing a c object inside membranes L containing $\overline{t_k}$ or $\overline{f_k}$ ($3k$ steps), creating a *no* object (1 step), and moving the membrane L containing an object z to membrane 2 (k steps).

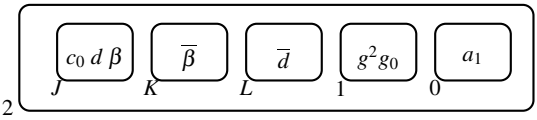
(viii) $[[c_{n+2m+4k+4}\ \beta]_J\overline{\beta}]_K \rightarrow [c_{n+2m+4k+4}\ \beta\ \alpha]_J[\overline{\beta}]_K$ (mexo)
$[c_{n+2m+4k+4}\ \alpha]_J[\overline{\alpha}\ b]_L \rightarrow [[no]_J\overline{\alpha}]_L$ (mendo)
$[\overline{\beta}]_K \rightarrow [\overline{\alpha}]_K[\overline{\alpha}]_K$ (div)
$[c_{n+2m+4k+4}\ \alpha]_J[\overline{\alpha}]_K \rightarrow [[yes]_J\overline{\alpha}]_K$ (mendo)
In case all the 2^k assignments do not contain an object c, a membrane L containing an object b and an object z reaches membrane 2. Membrane J with the counter value $n+m+4k+4$ exits membrane K, and in the next step enters membrane L containing a b object and creates an object *no* inside J, deleting the one in K. In case all the 2^k assignments contain an object c, then there is no membrane L containing a b object that will ever reach membrane 2. In this case, a *yes* object is created in membrane J by allowing it to enter membrane K after a determined period of time. The maximum number of steps needed to obtain a *no* object is $n+2m+4k+6$, while to obtain a *yes* object is $n+2m+4k+7$.

The number of membranes in the initial configuration is 6, and the number of objects is $n+6$. The size of the working alphabet is $4n+4m+11k+18$. The number of rules in the above system is: $n+2$ rules of type (i), $2(n+1)(2m-1)$ rules of type (ii), $2(n+2m)+3$ rules of type (iii), $6k$ rules of type (iv), 2 rules of type (v), 3 of type (vi), $4k+1$ rules of type (vii) and 4 rules of type $(viii)$. Hence, the size of the constructed system of mutual mobile membranes is $\mathcal{O}(mn)$.

Example 2.3. Consider the 2QBF problem with $\phi = \forall X \exists Y (C_1 \wedge C_2)$, $X = \{x_1, x_2\}$, $Y = \{x_3\}$, $C_1 = \neg x_1 \vee x_2$, $C_2 = x_1 \vee x_3$. In this case, $n = 3$, $m = 2$, $k = 2$ and

$$[[c_0\ d\ \beta]_J[\overline{\beta}]_K[\overline{d}]_L[g^2 g_0]_1[a_1]_0]_2$$

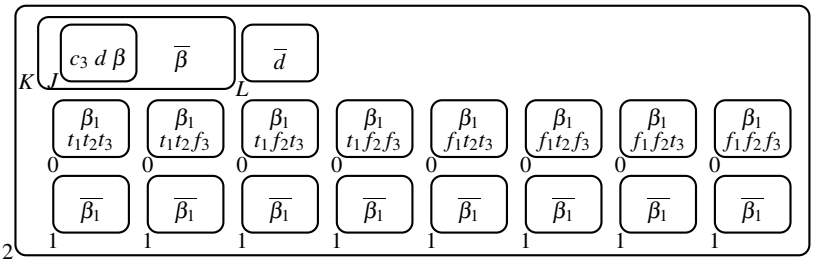
Graphically this is illustrated as:



The evolution of the system is described by the following steps. The working space is generated in $n = 3$ steps leading from the initial configuration 1 to configuration 4:

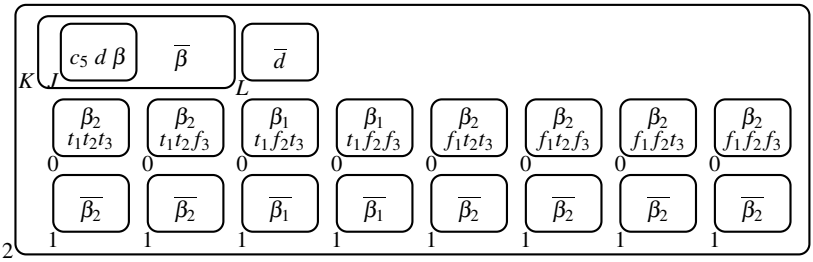
1. $[[c_0\ d\ \beta]_J[\overline{\beta}]_K[\overline{d}]_L[g^2 g_0]_1[a_1]_0]_2$
2. $[[[c_1\ d\ \beta]_J\overline{\beta}]_K[\overline{d}]_L[g^2\overline{\beta}_1]_1^2[t_1\ a_2]_0[f_1\ a_2]_0]_2$

3. $[[c_2\ d\ \beta]_J[\overline{\beta}]_K[\overline{d}]_L[g\overline{\beta_1}]_1^4[t_1\ t_2\ a_3]_0[t_1\ f_2\ a_3]_0[f_1\ t_2\ a_3]_0[f_1\ f_2\ a_3]_0]_2$
4. $[[[c_3\ d\ \beta]_J\overline{\beta}]_K[\overline{d}]_L[\overline{\beta_1}]_1^8[t_1\ t_2\ t_3\ \beta_1]_0[t_1\ t_2\ f_3\ \beta_1]_0[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0$
$[f_1\ t_2\ t_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_1]_0[f_1\ f_2\ t_3\ \beta_1]_0[f_1\ f_2\ f_3\ \beta_1]_0]_2$
Graphically the working space is described by the following picture:



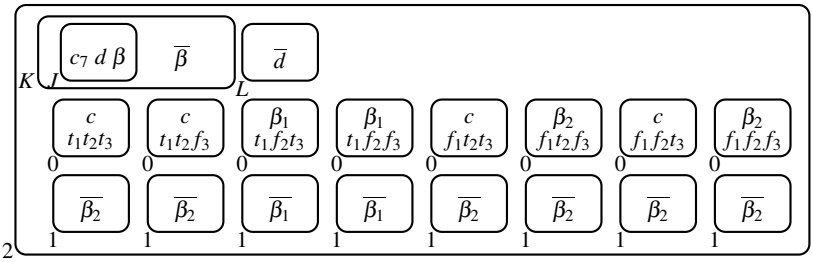
The next two steps mark the solutions of C_1 by replacing $\underline{\beta_1}$ by β_2:

5. $[[c_4\ d\ \beta]_J[\overline{\beta}]_K[\overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_1}[t_1\ t_2\ t_3\ \beta_1]_0]_1[\overline{\beta_1}[t_1\ t_2\ f_3\ \beta_1]_0]_1$
$[\overline{\beta_1}[f_1\ f_2\ f_3\ \beta_1]_0]_1[\overline{\beta_1}[f_1\ t_2\ t_3\ \beta_1]_0]_1[\overline{\beta_1}[f_1\ f_2\ t_3\ \beta_1]_0]_1[t_1\ f_2\ f_3\ \beta_1]_0[t_1\ f_2\ t_3\ \beta_1]_0]_2$
6. $[[[c_5\ d\ \beta]_J\overline{\beta}]_K[\overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ t_2\ t_3\ \beta_2]_0[t_1\ t_2\ f_3\ \beta_2]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0$
$[f_1\ t_2\ t_3\ \beta_2]_0[f_1\ f_2\ t_3\ \beta_2]_0[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0]_2$
The new configuration is graphically represented by:



The next two steps mark the solutions of C_2 by replacing $\underline{\beta_2}$ by c:

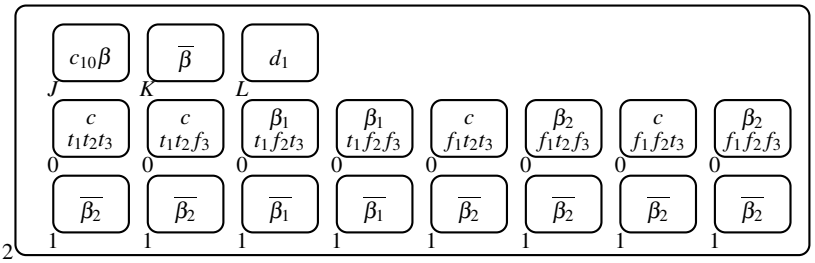
7. $[[c_6\ d\ \beta]_J[\overline{\beta}]_K[\overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^2[\overline{\beta_2}[t_1\ t_2\ t_3\ \beta_2]_0]_1[\overline{\beta_2}[t_1\ t_2\ f_3\ \beta_2]_0]_1[\overline{\beta_2}[f_1\ t_2\ t_3\ \beta_2]_0]_1$
$[\overline{\beta_2}[f_1\ f_2\ t_3\ \underline{\beta_2}]_0]_1[\underline{f_1\ t_2\ f_3}\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0]_2$
8. $[[[c_7\ d\ \beta]_J\overline{\beta}]_K[\overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0[f_1\ t_2\ t_3\ c]_0[f_1\ f_2\ t_3\ c]_0$
$[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0]_2$
The new configuration is graphically represented by:



In the next step, an object d_1 is generated. The number of steps needed to create this object is $n+2m+3 = 3+4+3 = 10$.

9. $[[c_7\ d\ \beta]_J[\overline{\beta}]_K[\overline{d}]_L[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0[f_1\ t_2\ t_3\ c]_0[f_1\ f_2\ t_3\ c]_0$
$[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0]_2$
10. $[[[c_7\ d\ \beta]_Jd]_L[\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0[f_1\ t_2\ t_3\ c]_0[f_1\ f_2\ t_3\ c]_0$
$[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0]_2$
11. $[[c_{10}\ \beta]_J[d_1]_L[\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0[f_1\ t_2\ t_3\ c]_0[f_1\ f_2\ t_3\ c]_0$
$[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0]_2$

The new configuration is graphically illustrated as below:

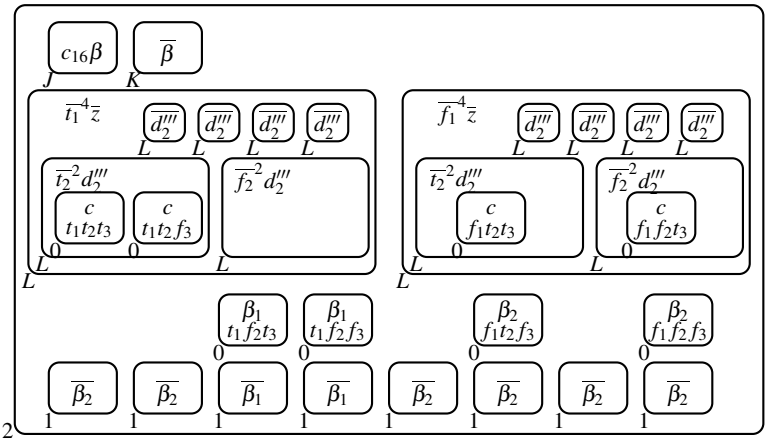


In what follows we check the $\forall$ part of the formula, namely whether all assignments over variables $x_1,\ldots,x_k$ appear in the existing solutions (the membranes 0 containing an object c). We need $3k = 3*2 = 6$ steps to see if there exists a combination that is missing from the membranes containing an object c, in order to create an object *no*.

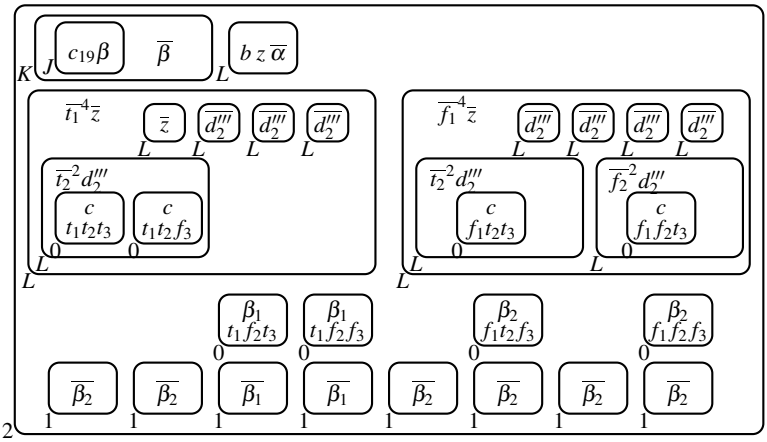
12. $[[[c_{11}\ \beta]_J\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[d_1']_L[d_1'']_L[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0[f_1\ t_2\ t_3\ c]_0[f_1\ f_2\ t_3\ c]_0$
$[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0]_2$
13. $[[c_{12}\ \beta]_J[\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[\overline{t_1}^4\overline{d_1'''}\overline{z}]_L[\overline{f_1}^4\overline{d_1'''}\overline{z}]_L[d_1''']_L[d_1''']_L[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0$
$[f_1\ t_2\ t_3\ c]_0[f_1\ f_2\ t_3\ c]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0]_2$
14. $[[[c_{13}\ \beta]_J\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0$
$[\overline{t_1}^4\overline{z}[d_2]_L[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0]_L[\overline{f_1}^4\overline{z}[d_2]_L[f_1\ t_2\ t_3\ c]_0[f_1\ f_2\ t_3\ c]_0]_L]_2$
15. $[[c_{14}\ \beta]_J[\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0$
$[\overline{t_1}^4\overline{z}[d_2']_L[d_2'']_L[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0]_L$
$[\overline{f_1}^4\overline{z}[d_2']_L[d_2'']_L[f_1\ t_2\ t_3\ c]_0[f_1\ f_2\ t_3\ c]_0]_L]_2$
16. $[[[c_{15}\ \beta]_J\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0$
$[\overline{t_1}^4\overline{z}[\overline{t_2}^2d_2''']_L[\overline{f_2}^2d_2''']_L[\overline{d_2''}]_L^2[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0]_L$
$[\overline{f_1}^4\overline{z}[\overline{t_2}^2d_2''']_L[\overline{f_2}^2d_2''']_L[\overline{d_2''}]_L^2[f_1\ t_2\ t_3\ c]_0[f_1\ f_2\ t_3\ c]_0]_L]_2$
17. $[[c_{16}\ \beta]_J[\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0$
$[\overline{t_1}^4\overline{z}[\overline{t_2}^2d_2'''[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0]_L[\overline{f_2}^2d_2''']_L[\overline{d_2'''}]_L^4]_L$
$[\overline{f_1}^4\overline{z}[\overline{t_2}^2d_2'''[f_1\ t_2\ t_3\ c]_0]_L[\overline{f_2}^2d_2'''[f_1\ f_2\ t_3\ c]_0]_L[\overline{d_2'''}]_L^4]_L]_2$

Since not all assignments for the variables x_1 and x_2 are present in the membranes containing a c object, then the answer provided is *no*. This is achieved by an elementary membrane containing an object $\overline{t_2}$ or $\overline{f_2}$ entering a membrane containing the object $\overline{d_2'''}$ in 1 step. The membrane containing the object b is sent,

in k steps to membrane 2 (skin membrane). The new configuration is graphically illustrated as below:



18. $[[[c_{17}\ \beta]_J\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0$
$[\overline{t_1}^4\overline{z}[\overline{t_2}^2 d_2'''[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0]_L[\overline{z}[b\ z]_L]_L[\overline{d_2'''}]_L^3]_L$
$[\overline{f_1}^4\overline{z}[\overline{t_2}^2 d_2'''[f_1\ t_2\ t_3\ c]_0]_L[\overline{f_2}^2 d_2'''[f_1\ f_2\ t_3\ c]_0]_L[\overline{d_2'''}]_L^4]_2$
19. $[[c_{18}\ \beta]_J[\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0$
$[\overline{t_1}^4\overline{z}[\overline{t_2}^2 d_2'''[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0]_L[\overline{z}]_L[b\ z]_L[\overline{d_2'''}]_L^3]_L$
$[\overline{f_1}^4\overline{z}[\overline{t_2}^2 d_2'''[f_1\ t_2\ t_3\ c]_0]_L[\overline{f_2}^2 d_2'''[f_1\ f_2\ t_3\ c]_0]_L[\overline{d_2'''}]_L^4]_2$
20. $[[[c_{19}\ \beta]_J\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0$
$[b\ z\overline{\alpha}]_L[\overline{t_1}^4\overline{z}[\overline{t_2}^2 d_2'''[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0]_L[\overline{z}]_L[\overline{d_2'''}]_L^3]_L$
$[\overline{f_1}^4\overline{z}[\overline{t_2}^2 d_2'''[f_1\ t_2\ t_3\ c]_0]_L[\overline{f_2}^2 d_2'''[f_1\ f_2\ t_3\ c]_0]_L[\overline{d_2'''}]_L^4]_2$
The new configuration is graphically illustrated as below:



If a membrane L contains a b object and has reached membrane 2, then an object *no* is created inside membrane J. If not, an extra step is performed in order to

have a *yes* object in membrane J. The maximum number of steps performed for an *no* object is $n+2m+4k+5=21$. For a positive answer the number of steps needed is $21+1=22$.

21. $[[c_{19}\,\beta\,\alpha]_J[\overline{\beta}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0$
$[b\ z\overline{\alpha}]_L[\overline{t_1}^4\overline{z}[\overline{t_2}^2 d_2'''[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0]_L[\overline{z}]_L[\overline{d_2'''}]_L^3]_L$
$[\overline{f_1}^4\overline{z}[\overline{t_2}^2 d_2'''[f_1\ t_2\ t_3\ c]_0]_L[\overline{f_2}^2 d_2'''[f_1\ f_2\ t_3\ c]_0]_L[\overline{d_2'''}]_L^4]_2$
22. $[[\overline{\alpha}]_K[\overline{\alpha}]_K[\overline{\beta_1}]_1^2[\overline{\beta_2}]_1^6[t_1\ f_2\ t_3\ \beta_1]_0[t_1\ f_2\ f_3\ \beta_1]_0[f_1\ t_2\ f_3\ \beta_2]_0[f_1\ f_2\ f_3\ \beta_2]_0$
$[[no]_J\ z\overline{\alpha}]_L[\overline{t_1}^4\overline{z}[\overline{t_2}^2 d_2'''[t_1\ t_2\ t_3\ c]_0[t_1\ t_2\ f_3\ c]_0]_L[\overline{z}]_L[\overline{d_2'''}]_L^3]_L$
$[\overline{f_1}^4\overline{z}[\overline{t_2}^2 d_2'''[f_1\ t_2\ t_3\ c]_0]_L[\overline{f_2}^2 d_2'''[f_1\ f_2\ t_3\ c]_0]_L[\overline{d_2'''}]_L^4]_2$

Exercise 2.6. Solve the 2QBF problem using other classes of mobile membranes.

2.5.3 Bin Packing Problem

The Bin Packing problem can be stated as follows: given a finite set A, a weight function $g: A \to \mathbb{N}$, and two constants $b \in \mathbb{N}$ and $c \in \mathbb{N}$, decide whether or not there exists a partition of A into b subsets such that the weight of each subset does not exceed c.

Consider $A = \{a_1, \ldots, a_n\}$, and a system of mutual mobile membranes having the initial configuration

$$[[\overline{\alpha}]_M[\alpha e_0]_J[\overline{\alpha}]_K[d^{n-1}e]_1 \ldots [d^{n-1}e]_b[a_1 \ldots a_n c_{1,0} \ldots c_{b,0}\overline{\beta_{1,0}} \ldots \overline{\beta_{b,0}}\beta]_0]_L$$

and working over the alphabet

$$\begin{aligned} V = {} & \{\alpha, \overline{\alpha}, \alpha_1, \beta, \overline{\beta}, no, yes, e, d\} \cup \{\gamma_i \mid 1 \le i \le n-1\} \cup \{d_i \mid 1 \le i \le b-2\} \\ & \cup \{w_i \mid 1 \le i \le b\} \cup \{a_i \mid 1 \le i \le n\} \cup \{e_i \mid 1 \le i \le 2bn+2c+3\} \\ & \cup \{\psi_{i,j}, \beta_{i,j}, x_{i,j}, y_{i,j}, z_{i,j} \mid 1 \le i \le b, 1 \le j \le n\} \\ & \cup \{c_{i,j} \mid 1 \le i \le b, 0 \le j \le 2bn+2c+1\} \end{aligned}$$

In addition to mutual endocytosis and mutual exocytosis rules, elementary division rules are used to generate all the possible subsets. The system of mutual mobile membranes solving the bin packing problem uses the rules:

(i) If $b = 2$ we have the rules:
$[a_i]_0 \to [x_{1,i}]_0[x_{2,i}]_0$, for $1 \le i \le n$ (div)
$[d]_j \to [\,]_j[\,]_j$, for $1 \le j \le 2$ (div)
$[e]_j \to [\beta_{j,0}]_j[\beta_{j,0}]_j$, for $1 \le j \le 2$ (div)
If $b > 2$ we have the rules:
$[a_i]_0 \to [x_{1,i}]_0[y_{1,i}]_0$, for $1 \le i \le n$ (div)
$[y_{j,i}]_0 \to [x_{j+1,i}]_0[y_{j+1,i}]_0$, for $1 \le j \le b-3$ and $1 \le i \le n$ (div)
$[y_{b-2,i}]_0 \to [x_{b-1,i}]_0[x_{b,i}]_0$, for $1 \le i \le n$ (div)
$[d]_j \to [\,]_j[d_1]_j$, for $1 \le j \le b$ (div)
$[d_k]_j \to [\,]_j[d_{k+1}]_j$, for $1 \le k \le b-3$ and $1 \le j \le b$ (div)
$[d_{b-2}]_j \to [\,]_j[\,]_j$, for $1 \le j \le b$ (div)
$[e]_j \to [\beta_{j,0}]_j[e_1]_j$ (div)
$[e_k]_j \to [\beta_{j,0}]_j[e_{k+1}]_j$, for $1 \le k \le b-3$ and $1 \le j \le b$ (div)

$[e_{b-2}]_j \to [\beta_{j,0}]_j[\beta_{j,0}]_j$, for $1 \le j \le b$ (div)

Different sets of rules are needed, depending on the number of bins (2 or more). The rules containing the objects a, x or y are used to create b^n membranes labelled by 0 containing all the possible subsets over $\{a_1,\ldots,a_n\}$. In $x_{j,i}$, the bin is denoted by j, while i denotes the variable placed in bin j. The objects $x_{j,i}$ are used to introduce objects that represent the weight of the object a_i. In each membrane are placed also the objects $\overline{\beta_{j,0}}$, $1 \le j \le b$, that are used to count the weight of each bin. The next rules create, for each $1 \le j \le b$, b^n membranes labelled by j each containing an object $\beta_{j,0}$.

(ii) $[\beta_{j,i}]_j[x_{j,i}\overline{\beta_{j,i}}]_0 \to [[\beta_{j,i+1}]_j x_{j,i}\overline{\beta_{j,i+1}}]_0$, for $0 \le i \le n-1$ and $1 \le j \le b$ (mendo)

$[[\beta_{j,i}]_j x_{j,i}\overline{\beta_{j,i}}]_0 \to [\beta_{j,i}]_j[z_{j,i}\, w_j^{g(a_i)}\overline{\beta_{j,i}}]_0$, for $1 \le i \le n$ and $1 \le j \le b$ (mexo)

$[[\beta_{j,i}]_j\overline{\beta_{j,i}}]_0 \to [\beta_{j,i}]_j[\overline{\beta_{j,i}}]_0$, for $1 \le i \le n$ and $1 \le j \le b$ (mexo)

These rules are used to replace $x_{j,i}$ by $z_{j,i}\, w_j^{g(a_i)}$. If $x_{j,i}$ is not present in a membrane 0, then that membrane just increments the second index associated to β. After applying these rules, each membrane 0 contains a number of objects w_j equal to the weight contained in the bin j, and also objects $z_{j,i}$ used to remember which objects are contained in each bin.

(iii) $[\beta_{j,n}]_j[w_j\,\overline{\beta_{j,n}}]_0 \to [[\beta_{j,n}]_j\overline{\beta_{j,n}}]_0$ (mendo)

$[[\beta_{j,n}]_j c_{j,i}\,\overline{\beta_{j,n}}]_0 \to [\beta_{j,n}]_j[c_{j,i+1}\,\overline{\beta_{j,n}}]_0$, for $0 \le i$ and $1 \le j \le b$ (mexo)

These rules are used to calculate the weights of each bin j, by using the objects $c_{j,0}$ that appear in all 0 membranes.

(iv) $[\alpha\, e_i]_J[\overline{\alpha}]_K \to [[\alpha\, e_{i+1}]_J\overline{\alpha}]_K$ for $0 \le i < (b+1)n+2c-1$ (mendo)

$[[\alpha\, e_i]_J\overline{\alpha}]_K \to [\alpha\, e_{i+1}]_J[\overline{\alpha}]_K$, for $0 \le i < (b+1)n+2c-1$ (mexo)

$[[\alpha\, e_i]_J\overline{\alpha}]_K \to [\alpha\, e_{i+1}]_J[\alpha e_{i+1}]_K$, for $i = (b+1)n+2c-1$ (mexo)

The above rules are used in parallel to calculate the number of steps performed. The number $(b+1)n+2c$ is determined by: generating space ($(b-1)n$ steps), replacing $x_{j,i}$ by corresponding weight ($2n$ steps) and calculating weights ($2c$ steps).

(v) $[[\alpha\, e_{(b+1)n+2c}]_J\overline{\alpha}]_K \to [\alpha\, e_{(b+1)n+2c}]_J[\alpha e_{(b+1)n+2c}]_K$ (mexo)

If $b = 2$ we have the rules

$[\alpha\, e_{(b+1)n+2c}]_J \to [\gamma_1]_J[\psi_1]_J$ (div)

$[\gamma_i]_J \to [\gamma_{i+1}]_J[\gamma_{i+1}]_J$, for $1 \le i \le n-2$ (div)

$[\gamma_{n-1}]_J \to [\overline{\beta}]_J[\overline{\beta}]_J$ (div)

If $b > 2$ we have the rules

$[\alpha\, e_{(b+1)n+2c}]_J \to [\gamma_1]_J[\psi_{1,1}]_J$ (div)

$[\psi_{i,j}]_J \to [\gamma_i]_J[\psi_{i,j+1}]_J$, for $1 \le j \le b-3$ and $1 \le i \le n-1$ (div)

$[\psi_{i,b-2}]_J \to [\gamma_i]_J[\gamma_i]_J$, for $1 \le i \le n-1$ (div)

$[\gamma_i]_J \to [\gamma_{i+1}]_J[\psi_{i+1,1}]_J$, for $1 \le i \le n-2$ (div)

$[\psi_{n,j}]_J \to [\overline{\beta}]_J[\psi_{n,j+1}]_J$, for $1 \le j \le b-3$ (div)

$[\psi_{n,b-2}]_J \to [\overline{\beta}]_J[\overline{\beta}]_J$ (div)

For $b \ge 2$ we have the rules

$[\overline{\beta}]_J[c_{k,j_k}\beta]_0 \to [[\,]_0\,]_J$, if $j_k > c$ (mendo)

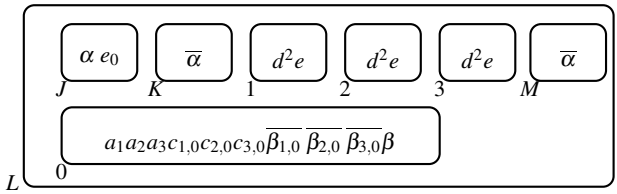
After an extra step needed to prepare the membrane J for division, b^n membranes J are created ($(b-1)n$ steps) to check which membranes respect the

weight condition. All membranes 0 that do not respect the condition are blocked inside the J membranes. In this way by choosing any membrane that is not placed inside a J membrane, we obtain a solution to the problem.

(vi) $[\alpha\ e_i]_K[\overline{\alpha}]_M \rightarrow [[\alpha\ e_{i+1}]_K\overline{\alpha}]_M$ for $(b+1)n+2c \leq i \leq 2bn+2c+2$ (mendo)
$[[\alpha\ e_i]_K\overline{\alpha}]_M \rightarrow [\alpha\ e_{i+1}]_K[\overline{\alpha}]_M$, for $(b+1)n+2c \leq i < 2bn+2c+2$ (mexo)
$[[\alpha\ e_i]_K\overline{\alpha}]_M \rightarrow [\overline{\beta}\beta]_K[\alpha_1]_M$, for $i = 2bn+2c+2$ or $i = 2bn+2c+3$ (mexo)
$[\overline{\beta}]_K[\beta]_0 \rightarrow [yes[\ ]_0]_K$ (mendo)
$[\alpha_1]_M \rightarrow [\overline{\beta}]_M[\overline{\beta}]_M$ (mendo)
$[\beta]_K[\overline{\beta}]_M \rightarrow [[no]_K]_M$ (mendo)
The above rules are used in parallel to calculate the number of steps performed. The number $2bn+2c+2$ is determined by: generating space ($(b-1)n$ steps), replacing $x_{j,i}$ by corresponding weight ($2n$ steps), calculating weights ($2c$ steps), generating J membranes ($(b-1)n$ steps), preparing division of J (1 step), blocking all membranes 0 that do not satisfy conditions (1 step). If there still exists a membrane 0 that is not inside a membrane J, then the object *yes* is created inside membrane K. Otherwise, after one more step, the *no* object is created inside membrane K. The computation stops after $2bn+2c+5$ steps, with the answer placed inside membrane K.

The number of membranes in the initial configuration is $b+5$, and the number of objects is $nb+n+2b+3$. The size of the working alphabet is $7bn+3b+2n+2c+2b^2n+2bc+9$. The number of rules in the above system is: $n+4$ rules of type (i) if $b=2$ or $3bn-7n+4b$ if $b>2$, $3bn-b$ rules of type (ii), $2cb$ rules of type (iii), $2(b+1)n+4c$ rules of type (iv), and $n+1$ rules of type (v) if $b=2$ or $bn+bc-n+b$ if $b>2$. Hence, the size of the constructed system of mutual mobile membranes is $max\{\mathcal{O}(bn), \mathcal{O}(bc)\}$.

Example 2.4. Consider the bin packing problem with $A=\{a_1,a_2,a_3\}$, $g(a_1)=1$, $g(a_2)=3$, $g(a_3)=2$, $b=3$ and $c=3$. In this case, $n=b=c=3$ and
$[[\overline{\alpha}]_M[\alpha e_0]_J[\overline{\alpha}]_K[d^2e]_1[d^2e]_2[d^2e]_3[a_1a_2a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0]_L$
Graphically this is illustrated as:



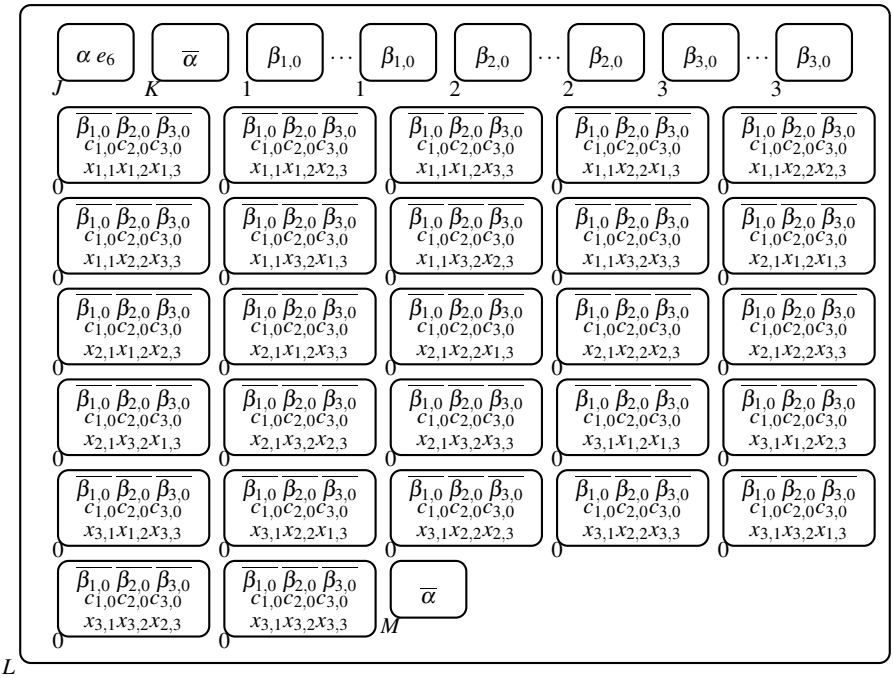
The evolution of the system is described by the following steps. The working space is created in $2n=2*3=6$ steps leading from the initial configuration 1 to configuration 7.

1. $[[\overline{\alpha}]_M[\alpha e_0]_J[\overline{\alpha}]_K[d^2e]_1[d^2e]_2[d^2e]_3[a_1a_2a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0]_L$
2. $[[\overline{\alpha}]_M[[\alpha e_1]_J\overline{\alpha}]_K[d\ e]_1[d_1d\ e]_1[d\ e]_2[d_1d\ e]_2[d\ e]_3[d_1d\ e]_3$
$[x_{1,1}a_2a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0[y_{1,1}a_2a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0]_L$

3. $[[\overline{\alpha}]_M[\alpha e_2]_J[\overline{\alpha}]_K[e]_1[d_1\ e]_1[d\ e]_1^2[e]_2[d_1\ e]_2[d\ e]_2^2[e]_3[d_1\ e]_3[d\ e]_3^2$
$[x_{1,1}x_{1,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}y_{1,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}a_2a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{3,1}a_2a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0]_L$
4. $[[\overline{\alpha}]_M[[\alpha e_3]_J\overline{\alpha}]_K[\beta_{1,0}]_1[e_1]_1[e]_1^4[d_1\ e]_1^2[\beta_{2,0}]_2[e_1]_2[e]_2^4[d_1\ e]_2^2[\beta_{3,0}]_3[e_1]_3[e]_3^4[d_1\ e]_3^2$
$[x_{1,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{1,2}y_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{2,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{3,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}x_{1,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}y_{1,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{1,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{3,1}y_{1,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0]_L$
5. $[[\overline{\alpha}]_M[\alpha e_4]_J[\overline{\alpha}]_K[\beta_{1,0}]_1^7[e_1]_1^4[e]_1^4[\beta_{2,0}]_2^7[e_1]_2^4[e]_2^4[\beta_{3,0}]_3^7[e_1]_3^4[e]_3^4$
$[x_{1,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{2,2}y_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{3,2}y_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}x_{1,2}y_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{2,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}x_{3,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{3,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{1,2}y_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{3,1}x_{2,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{3,2}a_3c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0]_L$
6. $[[\overline{\alpha}]_M[[\alpha e_5]_J\overline{\alpha}]_K[\beta_{1,0}]_1^{19}[e_1]_1^4[\beta_{2,0}]_2^{19}[e_1]_2^4[\beta_{3,0}]_3^{19}[e_1]_3^4$
$[x_{1,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{2,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{2,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{3,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{3,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{2,2}y_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{3,2}y_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{3,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{3,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{2,2}y_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{3,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{3,2}y_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0]_L$
7. $[[\overline{\alpha}]_M[\alpha e_6]_J[\overline{\alpha}]_K[\beta_{1,0}]_1^{27}[\beta_{2,0}]_2^{27}[\beta_{3,0}]_3^{27}$
$[x_{1,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{2,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{2,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{1,1}x_{3,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{1,1}x_{3,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{2,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}x_{2,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{2,1}x_{3,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{2,1}x_{3,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0[x_{3,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\ \overline{\beta_{2,0}}\ \overline{\beta_{3,0}}\beta]_0$

$[x_{3,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0[x_{3,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{2,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0[x_{3,1}x_{2,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0[x_{3,1}x_{3,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\beta_{1,0}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0$
$[x_{3,1}x_{3,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,0}}\,\overline{\beta_{2,0}}\,\overline{\beta_{3,0}}\beta]_0]_L$

Graphically the working space is described by the following picture, where for membranes labelled by 1, 2 and 3 we draw only two representatives:



In what follows we replace $x_{j,i}$ by $z_{j,i}w_j^{g(a_i)}$, such that in each membrane we obtain a number of objects w_j equal to the weight of the objects contained in the bin j. The objects $z_{j,i}$ are used to remember which objects are contained in each membrane 0.

8. $[[\overline{\alpha}]_M[[\alpha e_7]_J\overline{\alpha}]_K$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3x_{1,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3x_{1,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3x_{1,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3x_{1,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3x_{1,1}x_{2,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3x_{1,1}x_{2,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3x_{1,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3x_{1,1}x_{3,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3x_{1,1}x_{3,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$

$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{2,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{2,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{2,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{2,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{2,1}x_{2,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{2,1}x_{2,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{2,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{2,1}x_{3,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{2,1}x_{3,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{3,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{3,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{3,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{3,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{3,1}x_{2,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{3,1}x_{2,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{3,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{3,1}x_{3,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[[\beta_{1,1}]_1[\beta_{2,1}]_2[\beta_{3,1}]_3 x_{3,1}x_{3,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0]_L$

9. $[[\overline{\alpha}]_M[\alpha e_8]_J[\overline{\alpha}]_K[\beta_{1,1}]_1^{27}[\beta_{2,1}]_2^{27}[\beta_{3,1}]_3^{27}$
$[w_1z_{1,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_1z_{1,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_1z_{1,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_1z_{1,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_1z_{1,1}x_{2,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_1z_{1,1}x_{2,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_1z_{1,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_1z_{1,1}x_{3,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_1z_{1,1}x_{3,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_2z_{2,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_2z_{2,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_2z_{2,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_2z_{2,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_2z_{2,1}x_{2,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_2z_{2,1}x_{2,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_2z_{2,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_2z_{2,1}x_{3,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_2z_{2,1}x_{3,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_3z_{3,1}x_{1,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_3z_{3,1}x_{1,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_3z_{3,1}x_{1,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_3z_{3,1}x_{2,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_3z_{3,1}x_{2,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_3z_{3,1}x_{2,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_3z_{3,1}x_{3,2}x_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0[w_3z_{3,1}x_{3,2}x_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0$
$[w_3z_{3,1}x_{3,2}x_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,1}}\,\overline{\beta_{2,1}}\,\overline{\beta_{3,1}}\beta]_0]_L$

By applying in a similar way the set of rules (ii) for $2 \leq i \leq 3$, after 4 steps we obtain the configuration.

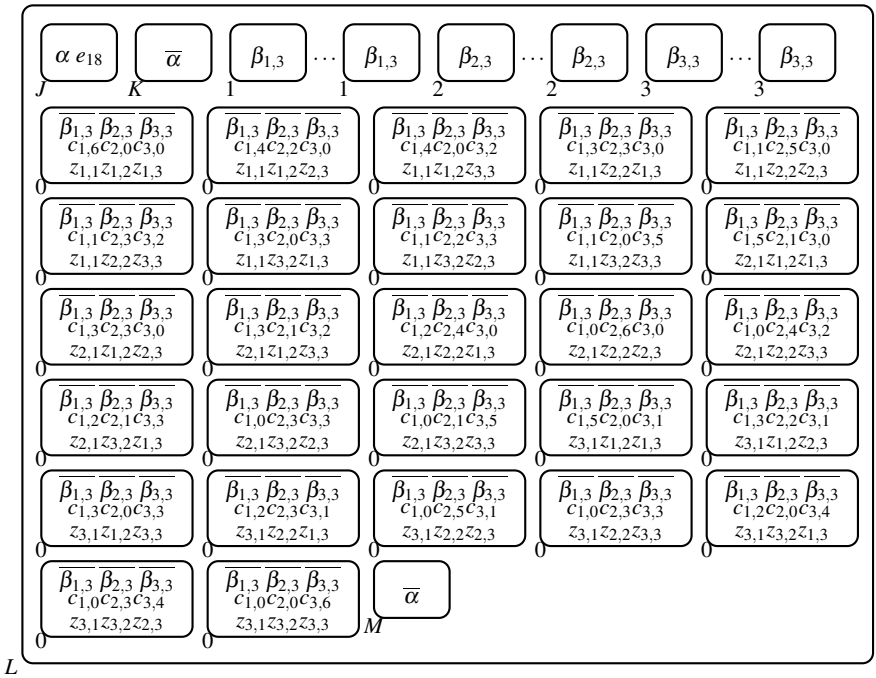
13. $[[\overline{\alpha}]_M[\alpha e_{12}]_J[\overline{\alpha}]_K[\beta_{1,3}]_1^{27}[\beta_{2,3}]_2^{27}[\beta_{3,3}^{27}]_3$
$[w_1^6z_{1,1}z_{1,2}z_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0$
$[w_1^4w_2^2z_{1,1}z_{1,2}z_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0$
$[w_1^4w_3^2z_{1,1}z_{1,2}z_{3,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0$
$[w_1^3w_2^3z_{1,1}z_{2,2}z_{1,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0$
$[w_1w_2^5z_{1,1}z_{2,2}z_{2,3}c_{1,0}c_{2,0}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0$

$[w_1 w_2^3 w_3^2 z_{1,1} z_{2,2} z_{3,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_1^3 w_3^3 z_{1,1} z_{3,2} z_{1,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_1 w_3^3 w_2^2 z_{1,1} z_{3,2} z_{2,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_1 w_3^5 z_{1,1} z_{3,2} z_{3,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_2 w_1^5 z_{2,1} z_{1,2} z_{1,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_2^3 w_1^3 z_{2,1} z_{1,2} z_{2,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_2 w_1^3 w_3^2 z_{2,1} z_{1,2} z_{3,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_2^4 w_1^2 z_{2,1} z_{2,2} z_{1,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_2^6 z_{2,1} z_{2,2} z_{2,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_2^4 w_3^2 z_{2,1} z_{2,2} z_{3,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_2 w_3^3 w_1^2 z_{2,1} z_{3,2} z_{1,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_2 w_3^3 w_1^2 z_{2,1} z_{3,2} z_{2,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_2 w_3^5 z_{2,1} z_{3,2} z_{3,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_3 w_1^5 z_{3,1} z_{1,2} z_{1,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_3 w_1^3 w_2^2 z_{3,1} z_{1,2} z_{2,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_3^3 w_1^3 z_{3,1} z_{1,2} z_{3,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_3 w_3^2 w_1^2 z_{3,1} z_{2,2} z_{1,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_3 w_2^5 z_{3,1} z_{2,2} z_{2,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_3^3 w_2^3 z_{3,1} z_{2,2} z_{3,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_3^4 w_1^2 z_{3,1} z_{3,2} z_{1,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_3^4 w_2^2 z_{3,1} z_{3,2} z_{2,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[w_3^6 z_{3,1} z_{3,2} z_{3,3} c_{1,0} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0]_L$

By applying in a similar way the set of rules (iii) for $1 \leq i \leq 3$, after 6 steps we obtain the next configuration, in which the second index of $c_{l,m}$ objects equals the number of w_j objects.

19. $[[\overline{\alpha}]_M [\alpha e_{18}]_J [\overline{\alpha}]_K [\beta_{1,3}]_1^{27} [\beta_{2,3}]_2^{27} [\beta_{3,3}^{27}]_3$
$[z_{1,1} z_{1,2} z_{1,3} c_{1,6} c_{2,0} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{1,1} z_{1,2} z_{2,3} c_{1,4} c_{2,2} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{1,1} z_{1,2} z_{3,3} c_{1,4} c_{2,0} c_{3,2} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{1,1} z_{2,2} z_{1,3} c_{1,3} c_{2,3} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{1,1} z_{2,2} z_{2,3} c_{1,1} c_{2,5} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{1,1} z_{2,2} z_{3,3} c_{1,1} c_{2,3} c_{3,2} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{1,1} z_{3,2} z_{1,3} c_{1,3} c_{2,0} c_{3,3} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{1,1} z_{3,2} z_{2,3} c_{1,1} c_{2,2} c_{3,3} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{1,1} z_{3,2} z_{3,3} c_{1,1} c_{2,0} c_{3,5} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{2,1} z_{1,2} z_{1,3} c_{1,5} c_{2,1} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{2,1} z_{1,2} z_{2,3} c_{1,3} c_{2,3} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{2,1} z_{1,2} z_{3,3} c_{1,3} c_{2,1} c_{3,2} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{2,1} z_{2,2} z_{1,3} c_{1,2} c_{2,4} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{2,1} z_{2,2} z_{2,3} c_{1,0} c_{2,6} c_{3,0} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{2,1} z_{2,2} z_{3,3} c_{1,0} c_{2,4} c_{3,2} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{2,1} z_{3,2} z_{1,3} c_{1,2} c_{2,1} c_{3,3} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{2,1} z_{3,2} z_{2,3} c_{1,0} c_{2,3} c_{3,3} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{2,1} z_{3,2} z_{3,3} c_{1,0} c_{2,1} c_{3,5} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{3,1} z_{1,2} z_{1,3} c_{1,5} c_{2,0} c_{3,1} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{3,1} z_{1,2} z_{2,3} c_{1,3} c_{2,2} c_{3,1} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{3,1} z_{1,2} z_{3,3} c_{1,3} c_{2,0} c_{3,3} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{3,1} z_{2,2} z_{1,3} c_{1,2} c_{2,3} c_{3,1} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{3,1} z_{2,2} z_{2,3} c_{1,0} c_{2,5} c_{3,1} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{3,1} z_{2,2} z_{3,3} c_{1,0} c_{2,3} c_{3,3} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{3,1} z_{3,2} z_{1,3} c_{1,2} c_{2,0} c_{3,4} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0 [z_{3,1} z_{3,2} z_{2,3} c_{1,0} c_{2,2} c_{3,4} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0$
$[z_{3,1} z_{3,2} z_{3,3} c_{1,0} c_{2,0} c_{3,6} \overline{\beta_{1,3}}\ \overline{\beta_{2,3}}\ \overline{\beta_{3,3}} \beta]_0]_L$

Graphically the working space is described by the following picture:



The next steps are used to create a *yes* object inside membrane K. We present only the final configuration:

29. $[[\overline{\beta}]^2_M[\overline{\beta}]^{12}_J[\beta_{1,3}]^{27}_1[\beta_{2,3}]^{27}_2[\beta^{27}_{3,3}]_3$
$[[z_{1,1}z_{1,2}z_{1,3}c_{1,6}c_{2,0}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J[[z_{1,1}z_{1,2}z_{2,3}c_{1,4}c_{2,2}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J$
$[[z_{1,1}z_{1,2}z_{3,3}c_{1,4}c_{2,0}c_{3,2}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J[[z_{1,1}z_{2,2}z_{2,3}c_{1,1}c_{2,5}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J$
$[z_{1,1}z_{2,2}z_{3,3}c_{1,1}c_{2,3}c_{3,2}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0[z_{1,1}z_{3,2}z_{1,3}c_{1,3}c_{2,0}c_{3,3}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0$
$[z_{1,1}z_{3,2}z_{2,3}c_{1,1}c_{2,2}c_{3,3}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0[[z_{1,1}z_{3,2}z_{3,3}c_{1,1}c_{2,0}c_{3,5}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J$
$[[z_{2,1}z_{1,2}z_{1,3}c_{1,5}c_{2,1}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J[z_{2,1}z_{1,2}z_{2,3}c_{1,3}c_{2,3}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0$
$[z_{2,1}z_{1,2}z_{3,3}c_{1,3}c_{2,1}c_{3,2}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0[[z_{2,1}z_{2,2}z_{1,3}c_{1,2}c_{2,4}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J$
$[[z_{2,1}z_{2,2}z_{2,3}c_{1,0}c_{2,6}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J[[z_{2,1}z_{2,2}z_{3,3}c_{1,0}c_{2,4}c_{3,2}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J$
$[z_{2,1}z_{3,2}z_{1,3}c_{1,2}c_{2,1}c_{3,3}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0[z_{2,1}z_{3,2}z_{2,3}c_{1,0}c_{2,3}c_{3,3}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0$
$[[z_{2,1}z_{3,2}z_{3,3}c_{1,0}c_{2,1}c_{3,5}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J[[z_{3,1}z_{1,2}z_{1,3}c_{1,5}c_{2,0}c_{3,1}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J$
$[z_{3,1}z_{1,2}z_{2,3}c_{1,3}c_{2,2}c_{3,1}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0[z_{3,1}z_{1,2}z_{3,3}c_{1,3}c_{2,0}c_{3,3}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0$
$[z_{3,1}z_{2,2}z_{1,3}c_{1,2}c_{2,3}c_{3,1}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0[[z_{3,1}z_{2,2}z_{2,3}c_{1,0}c_{2,5}c_{3,1}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J$
$[z_{3,1}z_{2,2}z_{3,3}c_{1,0}c_{2,3}c_{3,3}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}\beta]_0[[z_{3,1}z_{3,2}z_{1,3}c_{1,2}c_{2,0}c_{3,4}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J$
$[[z_{3,1}z_{3,2}z_{2,3}c_{1,0}c_{2,2}c_{3,4}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J[[z_{3,1}z_{3,2}z_{3,3}c_{1,0}c_{2,0}c_{3,6}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0]_J$
$[yes[z_{1,1}z_{2,2}z_{1,3}c_{1,3}c_{2,3}c_{3,0}\overline{\beta_{1,3}}\,\overline{\beta_{2,3}}\,\overline{\beta_{3,3}}]_0\beta]_K]_L$

Exercise 2.7. Solve the Bin Packing problem using other classes of mobile membranes.

2.5.4 Subset Sum Problem

The problem can be enounced as follows: given a finite set, A, a weight function, $g : A \to \mathbb{N}$, and a constant s, determine whether or not there exists a non-empty subset B of A such that $g(B) = s$.

Consider $A = \{a_1, \ldots, a_n\}$, and a system of mutual mobile membranes having the initial configuration

$$[[\overline{\alpha}]_M [\alpha\ e_0]_J [\overline{\alpha}]_K [d^{n-1} d_0]_1 [a_1\ c_0\ \beta_0]_0]_2$$

and working over the alphabet

$$V = \{no, yes, d, d_0, b, \alpha, \overline{\alpha}, \alpha_1, \beta, \overline{\beta}\} \cup \{\beta_i, \overline{\beta_i} \mid 0 \le i \le n\}$$
$$\cup \{a_i, x_i, y_i \mid 1 \le i \le n\} \cup \{\gamma_i \mid 1 \le i \le n-1\}$$
$$\cup \{e_i \mid 0 \le i \le 4n + 2s + 1\} \cup \{c_i \mid 0 \le i \le g(A)\}$$

In addition to mutual endocytosis and mutual exocytosis rules, elementary division rules are used to generate all the possible subsets. The system of mutual mobile membranes solving the Subset problem uses the rules:

(i) $[a_i]_0 \to [x_i a_{i+1}]_0 [a_{i+1}]_0$, for $1 \le i \le n-1$ (div)
$[a_n]_0 \to [x_n \beta]_0 [\beta]_0$ (div)
$[d]_1 \to [\ ]_1 [\ ]_1$ (div)
$[d_0]_1 \to [\overline{\beta}]_1 [\overline{\beta}]_1$ (div)
The first two rules generate 2^n membranes labelled by 0 containing all the possible subsets over variables $\{x_1, \ldots, x_n\}$. In each membrane labelled by 0 is placed also a symbol β . The next two rules generate 2^n membranes labelled by 1 each containing an object $\overline{\beta}$. The symbols β and $\overline{\beta}$ are used in mobility, where the membranes containing the object β are the ones that move.

(ii) $[\beta]_0 [\overline{\beta}]_1 \to [[\beta_1]_0 \overline{\beta_1}]_1$ (mendo)
$[\beta_i]_0 [\overline{\beta_i}]_1 \to [[\beta_{i+1}]_0 \overline{\beta_{i+1}}]_1$, for $1 \le i \le n-1$ (mendo)
$[[x_i \beta_i]_0 \overline{\beta_i}]_1 \to [y_i\ b^{g(a_i)} \beta_i]_0 [\overline{\beta_i}]_1$, for $1 \le i \le n$ (mexo)
$[[\beta_i]_0 \overline{\beta_i}]_1 \to [\beta_i]_0 [\overline{\beta_i}]_1$, for $1 \le i \le n$ (mexo)
These rules are used to replace x_i by $y_i\ b^{g(a_i)}$, for $1 \le i \le n$. If x_i is not present in a membrane 0, then the indexes of β and $\overline{\beta}$ are incremented. After applying these rules, each membrane 0 contains a number of objects b equal to the weight of the contained subset of A, and also objects y_i used to remember which objects are contained in this membrane.

(iii) $[b\ \beta_n]_0 [\overline{\beta_n}]_1 \to [[\beta_n]_0 \overline{\beta_n}]_1$ (mendo)
$[[c_i\ \beta_n]_0 \overline{\beta_n}]_1 \to [c_{i+1}\ \beta_n]_0 [\overline{\beta_n}]_1$, for $0 \le i$ (mexo)
These rules are used to calculate the weights of the subsets B of A, by using the objects c_0 that appear in all 0 membranes. For each b present in a membrane 0, the subscript of c, present in the same membrane, is incremented.

(iv) $[\alpha\ e_i]_J [\overline{\alpha}]_K \to [[\alpha\ e_{i+1}]_J \overline{\alpha}]_K$, for $0 \le i < 3n + 2s - 1$ (mendo)
$[[\alpha\ e_i]_J \overline{\alpha}]_K \to [\alpha\ e_{i+1}]_J [\overline{\alpha}]_K$, for $0 \le i < 3n + 2s - 1$ (mexo)
$[[\alpha\ e_i]_J \overline{\alpha}]_K \to [\alpha\ e_{i+1}]_J [\alpha\ e_{i+1}]_K$, for $i = 3n + 2s - 1$ (mexo)
These rules are used in parallel to calculate the number of steps performed. The counting stops after $3n + 2s$ steps, a number determined by: generating space (n

steps), replacing x_i by corresponding weight ($2n$ steps) and calculating weights ($2s$ steps).

(v) $[[\alpha\ e_{3n+2s}]_J\overline{\alpha}]_K \rightarrow [\alpha\ e_{3n+2s}]_J[\alpha e_{3n+2s}]_K$ (mexo)
$[\alpha\ e_{3n+2s}]_J \rightarrow [\gamma_1]_J[\gamma_1]_J$ (div)
$[\gamma_i]_J \rightarrow [\gamma_{i+1}]_J[\gamma_{i+1}]_J$, for $1 \leq i \leq n-2$ (div)
$[\gamma_{n-1}]_J \rightarrow [\overline{\beta_0}]_J[\overline{\beta_0}]_J$ (div)
$[c_i\beta_0]_0[\overline{\beta_0}]_J \rightarrow [[\ ]_0]_J$, for $0 \leq i, i \neq s$ (mendo)
An extra step, needed to prepare the membrane J for division, is performed if needed. If membrane J contains an object e_{3n+2s}, and is placed near membrane K, then 2^n membranes are generated (n steps) in order to check which membranes contain the object c_s. All membranes 0 that do not respect the condition are blocked inside a J membrane. In this way by choosing any membrane that it is not placed inside a J membrane, we obtain a solution to the problem.

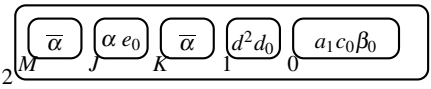
(vi) $[\alpha\ e_i]_K[\overline{\alpha}]_M \rightarrow [[\alpha\ e_{i+1}]_K\overline{\alpha}]_M$ for $3n+2s \leq i \leq 4n+2s$ (mendo)
$[[\alpha\ e_i]_K\overline{\alpha}]_M \rightarrow [\underline{\alpha}\ e_{i+1}]_K[\overline{\alpha}]_M$, for $3n+2s \leq i < 4n+2s$ (mexo)
$[[\underline{\alpha}\ e_i]_K\overline{\alpha}]_M \rightarrow [\overline{\beta_0}\beta_0]_K[\alpha_1]_M$, for $i = 4n+2s$ or $i = 4n+2s+1$ (mexo)
$[\overline{\beta_0}]_K[\beta_0]_0 \rightarrow [\underline{yes}[\ ]_0]_K$ (mendo)
$[\alpha_1]_M \rightarrow [\overline{\beta_0}]_M[\overline{\beta_0}]_M$ (div)
$[\beta_0]_K[\overline{\beta_0}]_M \rightarrow [[no]_K]_M$ (mendo)
The above rules are used in parallel to calculate the number of steps performed. The number $4n+2s+1$ is determined by: generating space (n steps), replacing x_i by corresponding weight ($2n$ steps),calculating weights ($2s$ steps), generating the J membranes ($n-1$ steps), preparing division of J (1 step), blocking all membranes 0 that do not satisfy conditions (1 step). If there still exists a membrane 0 that is not inside a membrane J, then the object *yes* is created inside membrane K. Otherwise, after one more step, the *no* object is created inside membrane K. The computation stops after $4n+2s+3$ steps, with the answer placed inside membrane K.

The number of membranes in the initial configuration is 6, and the number of objects is $n+7$. The size of the working alphabet is $10n+4s+12$. The number of rules in the above system is: $n+2$ rules of type (i), $3n$ rules of type (ii), $2(s+n-1)$ rules of type (iii), $3n+2s$ rules of type (iv), $n+2$ rules of type (v) and $2n+5$ rules of type (vi). Hence, the size of the constructed system is $\mathcal{O}(n)$.

Example 2.5. Consider the Subset problem with $A = \{a_1, a_2, a_3\}$, $s = 2$, $g(a_1) = 1$, $g(a_2) = 2$ and $g(a_3) = 1$. In this case, $n = 3$, $s = 2$, and the initial configuration of the system of mutual mobile membranes

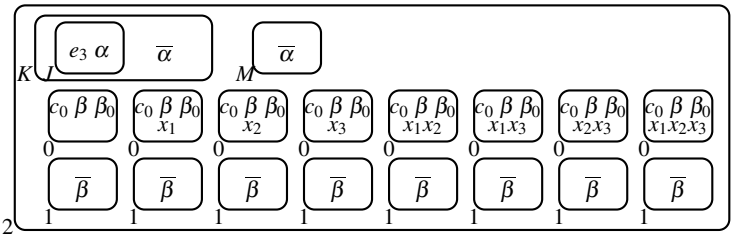
$$[[\overline{\alpha}]_M[\alpha\ e_0]_J[\overline{\alpha}]_K[d^2d_0]_1[a_1\ c_0\ \beta_0]_0]_2$$

Graphically this is illustrated as:



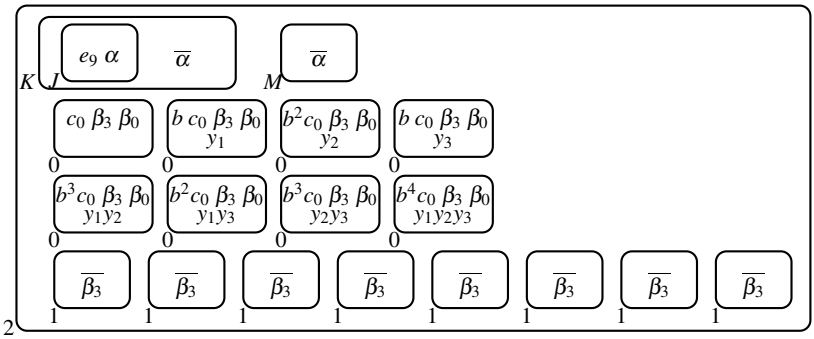
The evolution of the system is described by the following steps. The working space is generated in $n = 3$ steps leading from the initial configuration 1 to configuration 4:

1. $[[\overline{\alpha}]_M[\alpha\ e_0]_J[\overline{\alpha}]_K[d^2\ d_0]_1[a_1\ c_0\ \beta_0]_0]_2$
2. $[[\overline{\alpha}]_M[[\alpha\ e_1]_J\overline{\alpha}]_K[d\ d_0]_1^2[x_1\ a_2\ c_0\ \beta_0]_0[a_2\ c_0\ \beta_0]_0]_2$
3. $[[\overline{\alpha}]_M[\alpha\ e_2]_J[\overline{\alpha}]_K[d_0]_1^4[x_1\ x_2\ a_3\ c_0\ \beta_0]_0[x_1\ a_3\ c_0\ \beta_0]_0[x_2\ a_3\ c_0\ \beta_0]_0[a_3\ c_0\ \beta_0]_0]_2$
4. $[[\overline{\alpha}]_M[[\alpha\ e_3]_J\overline{\alpha}]_K[\overline{\beta}]_1^8[x_1\ x_2\ x_3\ c_0\ \beta\beta_0]_0[x_1\ x_2\ c_0\ \beta\beta_0]_0[x_1\ x_3\ c_0\ \beta\beta_0]_0[x_1\ c_0\ \beta\beta_0]_0$
$[x_2\ x_3\ c_0\ \beta\beta_0]_0[x_2\ c_0\ \beta\beta_0]_0[x_3\ c_0\ \beta\beta_0]_0[c_0\ \beta\beta_0]_0]_2$
Graphically the working space is described by the following picture:



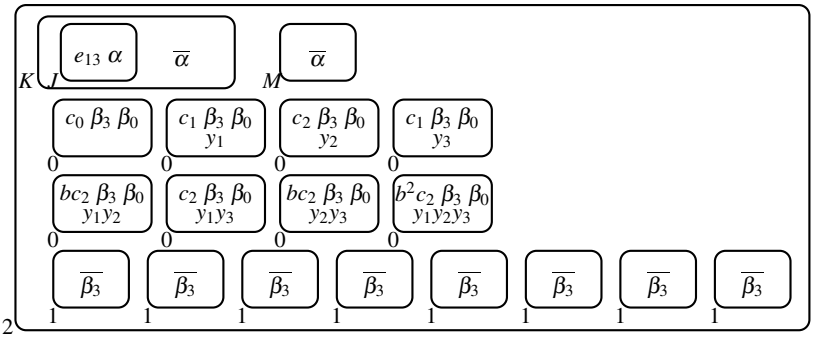
In the next steps we replace x_i by y_i and $b^{g(a_i)}$. We use y_i to mark the fact that in the membrane 0 containing it there is a subset containing a_i. The multiset of objects $b^{g(a_i)}$ is used to denote the weight of object a_i.

5. $[[\overline{\alpha}]_M[\alpha\ e_4]_J[\overline{\alpha}]_K[[x_1\ x_2\ x_3\ c_0\ \beta_1\ \beta_0]_0\underline{\overline{\beta_1}}]_1[[x_1\ x_2\ c_0\ \beta_1\ \beta_0]_0\underline{\overline{\beta_1}}]_1[[x_1\ x_3\ c_0\ \beta_1\ \beta_0]_0\overline{\beta_1}]_1$
$[[\overline{\alpha}]_M[x_1\ c_0\ \beta_1]_0\overline{\beta_1}]_1[[x_2\ x_3\ c_0\ \underline{\beta_1}\ \beta_0]_0\overline{\beta_1}]_1[[x_2\ c_0\ \beta_1\ \beta_0]_0\overline{\beta_1}]_1$
$[[x_3\ c_0\ \beta_1\ \beta_0]_0\overline{\beta_1}]_1[[c_0\ \beta_1\ \beta_0]_0\overline{\beta_1}]_1]_2$
6. $[[\overline{\alpha}]_M[[\alpha\ e_5]_J\overline{\alpha}]_K[\overline{\beta_1}]_1^8[b\ y_1\ x_2\ x_3\ c_0\ \beta_1\ \beta_0]_0[b\ y_1\ x_2\ c_0\ \beta_1\ \beta_0]_0[b\ y_1\ x_3\ c_0\ \beta_1\ \beta_0]_0$
$[b\ y_1\ c_0\ \beta_1\ \beta_0]_0[x_2\ x_3\ c_0\ \beta_1\ \beta_0]_0[x_2\ c_0\ \beta_1\ \beta_0]_0[x_3\ c_0\ \beta_1]_0[c_0\ \beta_1\ \underline{\beta_0}]_0]_2$
7. $[[\overline{\alpha}]_M[\alpha\ e_6]_J[\overline{\alpha}]_K[[\underline{b\ y_1}\ x_2\ x_3\ c_0\ \beta_2\ \beta_0]_0\overline{\beta_2}]_1[[b\ y_1\ x_2\ c_0\ \beta_2\ \beta_0]_0\underline{\beta_0}\overline{\beta_2}]_1$
$[[b\ y_1\ x_3\ c_0\ \beta_2\ \underline{\beta_0}]_0\overline{\beta_2}]_1[[b\ y_1\ c_0\ \underline{\beta_2}\ \beta_0]_0\overline{\beta_2}]_1[[x_2\ \underline{x_3}\ c_0\ \beta_2\ \beta_0]_0\overline{\beta_2}]_1$
$[[x_2\ c_0\ \beta_2\ \beta_0]_0\overline{\beta_2}]_1[[\underline{x_3}\ c_0\ \beta_2\ \beta_0]_0\overline{\beta_2}]_1[[c_0\ \beta_2\ \beta_0]_0\overline{\beta_2}]_1]_2$
8. $[[\overline{\alpha}]_M[[\alpha\ e_7]_J\overline{\alpha}]_K[\overline{\beta_2}]_1^8[b^3\ y_1\ y_2\ x_3\ c_0\ \beta_2\ \beta_0]_0[b^3\ y_1\ y_2\ c_0\ \beta_2\ \beta_0]_0[b\ y_1\ x_3\ c_0\ \beta_2\ \beta_0]_0$
$[b\ y_1\ c_0\ \beta_2\ \beta_0]_0[b^2\ y_2\ x_3\ c_0\ \beta_2\ \beta_0]_0[b^2\ \underline{y_2}\ c_0\ \beta_2\ \beta_0]_0[x_3\ c_0\ \beta_2\ \beta_0]_0\underline{[c_0\ \beta_2\ \beta_0]_0}]_2$
9. $[[\overline{\alpha}]_M[\alpha\ e_8]_J[\overline{\alpha}]_K[[\underline{b\ y_1}\ x_2\ x_3\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[[b\ y_1\ x_2\ c_0\ \beta_3\ \beta_0]_0\underline{\beta_0}\overline{\beta_3}]_1$
$[[b\ y_1\ x_3\ c_0\ \beta_3\ \underline{\beta_0}]_0\overline{\beta_3}]_1[[b\ y_1\ c_0\ \underline{\beta_3}\ \beta_0]_0\overline{\beta_3}]_1[[x_2\ \underline{x_3}\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1$
$[[x_2\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[[\underline{x_3}\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[[c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1]_2$
10. $[[\overline{\alpha}]_M[[\alpha\ e_9]_J\overline{\alpha}]_K[\overline{\beta_3}]_1^8[b^4\ y_1\ y_2\ y_3\ c_0\ \beta_3\ \beta_0]_0[b^3\ y_1\ y_2\ c_0\ \beta_3\ \beta_0]_0[b^2\ y_1\ y_3\ c_0\ \beta_3\ \beta_0]_0$
$[b\ y_1\ c_0\ \beta_3\ \beta_0]_0[b^3\ y_2\ y_3\ c_0\ \beta_3\ \beta_0]_0[b^2\ y_2\ c_0\ \beta_3\ \beta_0]_0[b\ y_3\ c_0\ \beta_3\ \beta_0]_0[c_0\ \beta_3\ \beta_0]_0]_2$
Graphically the working space is described by the following picture:



In what follows we calculate the weight of each membrane 0 by using the object c_0, until we reach c_2 since $s = 2$.

11. $[[\overline{\alpha}]_M[\alpha\ e_{10}]_J[\overline{\alpha}]_K[\overline{\beta_3}]_1[[b^3\ y_1\ y_2\ y_3\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[[b^2\ y_1\ y_2\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1$
$[[b\ y_1\ y_3\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[[y_1\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[[b^2\ y_2\ y_3\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1$
$[[b\ y_2\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[[y_3\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[c_0\ \beta_3]_0]_2$
12. $[[\overline{\alpha}]_M[[\alpha\ e_{11}]_J\overline{\alpha}]_K[\overline{\beta_3}]_1^8[b^3\ y_1\ y_2\ y_3\ c_1\ \beta_3\ \beta_0]_0[b^2\ y_1\ y_2\ c_1\ \beta_3\ \beta_0]_0[b\ y_1\ y_3\ c_1\ \beta_3\ \beta_0]_0$
$[y_1\ c_1\ \beta_3\ \beta_0]_0[b^2\ y_2\ y_3\ c_1\ \beta_3\ \beta_0]_0[b\ y_2\ c_1\ \beta_3\ \beta_0]_0[y_3\ c_1\ \beta_3\ \beta_0]_0[c_0\ \beta_3\ \beta_0]_0]_2$
13. $[[\overline{\alpha}]_M[\alpha\ e_{12}]_J[\overline{\alpha}]_K[\overline{\beta_3}]_1^3[[b^2\ y_1\ y_2\ y_3\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[[b\ y_1\ y_2\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1$
$[[y_1\ y_3\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[y_1\ c_0\ \beta_3\ \beta_0]_0[[b\ y_2\ y_3\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1$
$[[y_2\ c_0\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[y_3\ c_0\ \beta_3\ \beta_0]_0[c_0\ \beta_3\ \beta_0]_0]_2$
14. $[[\overline{\alpha}]_M[[\alpha\ e_{13}]_J\overline{\alpha}]_K[\overline{\beta_3}]_1^8[b^2\ y_1\ y_2\ y_3\ c_2\ \beta_3\ \beta_0]_0[b\ y_1\ y_2\ c_2\ \beta_3\ \beta_0]_0[y_1\ y_3\ c_2\ \beta_3\ \beta_0]_0$
$[y_1\ c_1\ \beta_3\ \beta_0]_0[b\ y_2\ y_3\ c_2\ \beta_3\ \beta_0]_0[y_2\ c_2\ \beta_3\ \beta_0]_0[y_3\ c_1\ \beta_3\ \beta_0]_0[c_0\ \beta_3\ \beta_0]_0]_2$
Graphically the working space is described by the following picture:



Now we divide membrane J, after performing an extra step that moves membrane J out of membrane K, in order to check if the index of c_i from the 0 membranes equals s. In parallel, if there still exist membranes 0 containing b objects, then the subscript of c_i is increased.

15. $[[\overline{\alpha}]_M[\alpha\ e_{13}]_J[\alpha\ e_{13}]_K[\overline{\beta_3}]_1^5[[b\ y_1\ y_2\ y_3\ c_2\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[[y_1\ y_2\ c_2\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1$
$[y_1\ y_3\ c_2\ \beta_3\ \beta_0]_0[y_1\ c_1\ \beta_3\ \beta_0]_0[[y_2\ y_3\ c_2\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[y_2\ c_2\ \beta_3\ \beta_0]_0[y_3\ c_1\ \beta_3\ \beta_0]_0$
$[c_0\ \beta_3\ \beta_0]_0]_2$
16. $[[\overline{\alpha}]_M[\overline{\alpha}[\alpha\ e_{14}]_K]_M[\gamma_1]_J^2[\overline{\beta_3}]_1^8[b\ y_1\ y_2\ y_3\ c_3\ \beta_3\ \beta_0]_0[y_1\ y_2\ c_3\ \beta_3\ \beta_0]_0[y_1\ y_3\ c_2\ \beta_3\ \beta_0]_0$
$[y_1\ c_1\ \beta_3\ \beta_0]_0[y_2\ y_3\ c_3\ \beta_3\ \beta_0]_0[y_2\ c_2\ \beta_3\ \beta_0]_0[y_3\ c_1\ \beta_3\ \beta_0]_0[c_0\ \beta_3\ \beta_0]_0]_2$
17. $[[\overline{\alpha}]_M[\gamma_2]_J^4[\alpha\ e_{15}]_K[\overline{\beta_3}]_1^7[[y_1\ y_2\ y_3\ c_3\ \beta_3\ \beta_0]_0\overline{\beta_3}]_1[y_1\ y_2\ c_3\ \beta_3\ \beta_0]_0[y_1\ y_3\ c_2\ \beta_3\ \beta_0]_0$
$[y_1\ c_1\ \beta_3\ \beta_0]_0[y_2\ y_3\ c_3\ \beta_3\ \beta_0]_0[y_2\ c_2\ \beta_3\ \beta_0]_0[y_3\ c_1\ \beta_3\ \beta_0]_0[c_0\ \beta_3\ \beta_0]_0]_2$
18. $[[\overline{\alpha}[\alpha\ e_{16}]_K]_M[\overline{\beta_0}]_J^8[\overline{\beta_3}]_1^8[y_1\ y_2\ y_3\ c_4\ \beta_3\ \beta_0]_0[y_1\ y_2\ c_3\ \beta_3\ \beta_0]_0[y_1\ y_3\ c_2\ \beta_3\ \beta_0]_0$
$[y_1\ c_1\ \beta_3\ \beta_0]_0[y_2\ y_3\ c_3\ \beta_3\ \beta_0]_0[y_2\ c_2\ \beta_3\ \beta_0]_0[y_3\ c_1\ \beta_3\ \beta_0]_0[c_0\ \beta_3\ \beta_0]_0]_2$
19. $[[\alpha_1]_M[\beta_0\overline{\beta_0}]_K[\overline{\beta_0}]_J^8[\overline{\beta_3}]_1^8[[y_1\ y_2\ y_3\ \beta_3]_0]_J[[y_1\ y_2\ \beta_3]_0]_J[y_1\ y_3\ c_2\ \beta_3\ \beta_0]_0$
$[[y_1\ \beta_3]_0]_J[[y_2\ y_3\ \beta_3]_0]_J[y_2\ c_2\ \beta_3\ \beta_0]_0[[y_3\ \beta_3]_0]_J[[\beta_3]_0]_J]_2$
The next steps are used to generate a *yes* object inside membrane K. We present only the final configuration obtained after $4n + 2s + 3 = 12 + 4 + 3 = 19$ steps:
20. $[[\overline{\beta}_1]_M^2[\beta_0\ yes[y_1\ y_3\ c_2\ \beta_3]_0]_K[\overline{\beta_0}]_J^8[\overline{\beta_3}]_1^8[[y_1\ y_2\ y_3\ \beta_3]_0]_J[[y_1\ y_2\ \beta_3]_0]_J$
$[[y_1\ \beta_3]_0]_J[[y_2\ y_3\ \beta_3]_0]_J[y_2\ c_2\ \beta_3\ \beta_0]_0[[y_3\ \beta_3]_0]_J[[\beta_3]_0]_J]_2$

Exercise 2.8. Solve the Subset Sum problem using other classes of mobile membranes.

2.5.5 Knapsack Problem (0/1)

The decision Knapsack problem can be stated as follows: Given a knapsack of capacity $k \in \mathbb{N}$, a set A of n elements, a weight function $g : A \to \mathbb{N}$, a value function $r : A \to \mathbb{N}$, and a constant $l \in \mathbb{N}$, decide whether or not there exists a subset of A such that its weight does not exceed k and its value is greater than or equal than l.

Consider $A = \{a_1, \ldots, a_n\}$, and a system of mutual mobile membranes having the initial configuration

$$[[\overline{\psi_n}]_M[\alpha\ e_0]_J[\overline{\alpha}]_K[d^{n-1}d_0]_1[d^{n-1}d_0]_2[a_1\ b_0\ c_0\ \beta_0]_0]_3$$

and working over the alphabet

$$V = \{yes, no, d, d_0, b, c, \varphi, \alpha, \overline{\alpha}, \beta, \overline{\beta}, \psi, \overline{\psi}\}$$
$$\cup\{a_i, x_i, y_i, \psi_i, \overline{\psi_i}, \varphi_i, \gamma_i \mid 1 \leq i \leq n\} \cup \{\beta_i, \overline{\beta_i} \mid 0 \leq i \leq n\}$$
$$\cup\{b_i, c_i \mid 0 \leq i \leq n + max\{k,l\}\} \cup \{e_i \mid 0 \leq i \leq 3n + 2max\{k,l\}\}$$

In addition to mutual endocytosis and mutual exocytosis rules, elementary division rules are used to generate all the possible subsets. The system of mutual mobile membranes solving the Knapsack problem uses the rules:

(i) $[a_i]_0 \to [x_i z_i a_{i+1}]_0[a_{i+1}]_0$, for $1 \leq i \leq n-1$ (div)
$[a_n]_0 \to [x_n z_n \overline{\beta}\,\overline{\psi}]_0[\overline{\beta}\,\overline{\psi}]_0$ (div)
$[d]_j \to [\,]_j[\,]_j$, for $1 \leq j \leq 2$ (div)
$[d_0]_1 \to [\beta]_1[\beta]_1$ (div)
$[d_0]_2 \to [\psi]_2[\psi]_2$ (div)
The first two rules generate 2^n membranes labelled by 0 containing all the possible subsets over variables $\{x_1, \ldots, x_n\}$. When an object x_i appears in a membrane 0, there appears also an object z_i. The objects x_i and z_i are used to introduce objects that represent the weight and value, respectively, of each object a_i from a subset of A. In each membrane are placed also two symbols $\overline{\beta}$ and $\overline{\psi}$. The next two rules generate 2^n membranes labelled by 1 each containing an object β and 2^n membranes labelled by 2 each containing an object ψ. The symbols β, $\overline{\beta}$, ψ, and $\overline{\psi}$ are used in the mobility of membranes 1 and 2.

(ii) $[\beta]_1[\overline{\beta}]_0 \to [[\beta_1]_1\overline{\beta_1}]_0$ (mendo)
$[\beta_i]_1[\overline{\beta_i}]_0 \to [[\beta_{i+1}]_1\overline{\beta_{i+1}}]_0$, for $1 \leq i \leq n-1$ (mendo)
$[[\beta_i]_1 x_i\overline{\beta_i}]_0 \to [\beta_i]_1[y_i\, b^{g(a_i)}\overline{\beta_i}]_0$, for $1 \leq i \leq n$ (mexo)
$[[\beta_i]_1\overline{\beta_i}]_0 \to [\beta_i]_1[\overline{\beta_i}]_0$, for $1 \leq i \leq n$ (mexo)
These rules are used to replace x_i by $y_i\, b^{g(a_i)}$, for $1 \leq i \leq n$. If x_i is not present in a membrane 0, then the indexes of β and $\overline{\beta}$ are incremented. After applying these rules, each membrane 0 contains a number of objects b equal to the weight of the contained subset of A, and also objects y_i used to remember which objects are contained in this membrane.

(iii) $[\psi]_2[\overline{\psi}]_0 \to [[\psi_1]_2\overline{\psi_1}]_0$ (mendo)

$[\psi_i]_2[\overline{\psi_i}]_0 \rightarrow [[\psi_{i+1}]_2\overline{\psi_{i+1}}]_0$, for $1 \leq i \leq n-1$ (mendo)
$[[\psi_i]_2 z_i \overline{\psi_i}]_0 \rightarrow [\psi_i]_2[c^{r(a_i)}\overline{\psi_i}]_0$, for $1 \leq i \leq n$ (mexo)
$[[\psi_i]_2\overline{\psi_i}]_0 \rightarrow [\psi_i]_2[\overline{\psi_i}]_0$, for $1 \leq i \leq n$ (mexo)
In parallel with the rules (ii), these rules are used to replace z_i by $c^{r(a_i)}$, for $1 \leq i \leq n$. If z_i is not present in a membrane 0, then the indexes of ψ and $\overline{\psi}$ are incremented. After applying these rules, each membrane 0 contains a number of objects c equal to the value of the contained subset of A.

(iv) $[\beta_n]_1[b\ \overline{\beta_n}]_0 \rightarrow [[\beta_n]_1\overline{\beta_n}]_0$ (mendo)
$[[\beta_n]_1 b_i\ \overline{\beta_n}]_0 \rightarrow [\beta_n]_1[b_{i+1}\ \overline{\beta_n}]_0$, for $0 \leq i$ (mexo)
These rules are used to calculate the weights of the subsets B of A, by using the objects b_0 that appear in all 0 membranes. If an object b is present, then the index of b_i, placed in the same membrane as b, is incremented.

(v) $[\psi_n]_2[c\ \overline{\psi_n}]_0 \rightarrow [[\psi_n]_2\overline{\psi_n}]_0$ (mendo)
$[[\psi_n]_2 c_i\ \overline{\psi_n}]_0 \rightarrow [\psi_n]_2[c_{i+1}\ \overline{\psi_n}]_0$, for $0 \leq i$ (mexo)
In parallel with the rules (iv), these rules are used to calculate the value of the subsets B of A, by using the objects c_0 that appear in all 0 membranes. If an object c is present, then the index of c_i, placed in the same membrane as c, is incremented.

(vi) $[\alpha\ e_i]_J[\overline{\alpha}]_K \rightarrow [[\alpha\ e_{i+1}]_J\overline{\alpha}]_K$ (mendo)
$[[\alpha\ e_i]_J\overline{\alpha}]_K \rightarrow [\alpha\ e_{i+1}]_J[\overline{\alpha}]_K$, for $0 \leq i \leq 3n+2max\{k,l\}-1$ (mexo)
We use these rules in parallel to calculate the number of steps performed. The counting stops after $3n+2max\{k,l\}$ steps, a number determined by: generating space (n steps), replacing x_i by the corresponding weight ($2n$ steps) and calculating weights and values ($2max\{k,l\}$ steps).

(vii) $[\alpha\ e_{3n+2max\{k,l\}}]_J[\overline{\alpha}]_K \rightarrow [[\alpha\ e_{3n+2max\{k,l\}}]_J\overline{\alpha}]_K$ (mendo)
$[[\alpha\ e_{3n+2max\{k,l\}}]_J\overline{\alpha}]_K \rightarrow [\alpha\ e_{3n+2max\{k,l\}}]_J[\varphi]_K$ (mexo)
$[\alpha\ e_{3n+2max\{k,l\}}]_J \rightarrow [\gamma_1]_J[\gamma_1]_J$ (div)
$[\gamma_i]_J \rightarrow [\gamma_{i+1}]_J[\gamma_{i+1}]_J$, for $1 \leq i \leq n-2$ (div)
$[\gamma_{n-1}]_J \rightarrow [\beta_n]_J[\beta_n]_J$ (div)
$[\varphi]_K \rightarrow [\varphi_1]_K[\varphi_1]_K$ (div)
$[\varphi_i]_K \rightarrow [\varphi_{i+1}]_K[\varphi_{i+1}]_K$, for $1 \leq i \leq n-2$ (div)
$[\varphi_{n-1}]_K \rightarrow [\psi_n]_K[\psi_n]_K$ (div)
$[\beta_n]_J[b_i\overline{\beta_n}]_0 \rightarrow [[\ ]_J]_0$, for $k < i$ (mendo)
$[\psi_n]_K[c_i\overline{\psi_n}]_0 \rightarrow [[\ ]_K]_0$, for $0 \leq i < l$ (mendo)
One or two extra steps, needed to prepare the membranes J and K for division, are performed if needed. If membrane J contains two objects α and $3n+2max\{k,l\}$, and is placed near membrane K containing an object ϕ, then 2^n membranes J are generated to check which membranes respect the weight condition, and 2^n membranes K are generated to check which membranes respect the value condition. All membranes 0 that do not respect the conditions are blocked inside a J or K membrane. In this way by choosing any membrane that is not placed inside a J or K membrane, we obtain a solution to the problem.

(viii) $[\overline{\psi_n}]_M[\psi_n]_K \rightarrow [\overline{\psi_n}[\psi_n]_K]_M$ (mendo)
$[\overline{\psi_n}[\psi_n]_K]_M \rightarrow [\overline{\beta_0\beta_0}]_M[\beta_0]_K$ (mexo)
$[\beta_0]_0[\overline{\beta_0}]_M \rightarrow [yes[\ ]_0]_M$ (mendo)

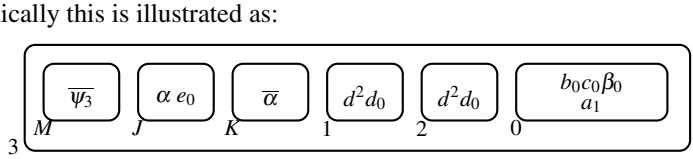
$[\underline{\beta_0}]_K \rightarrow [\overline{\beta_0}]_K[\overline{\beta_0}]_K$ (div)
$[\beta_0]_K[\beta_0]_M \rightarrow [[no]_M]_K$ (mendo)
The computation stops after 2 more steps (maximum $4n+2max\{k,l\}+5$ steps in total). If there still exists a membrane 0 that is not blocked inside a J or K membrane, then the *yes* object is created inside membrane M. Otherwise, after one more step, the *no* object is created inside membrane M.

The number of membranes in the initial configuration is 7, and the number of objects is $2n+9$. The size of the working alphabet is $14n+3max\{k,l\}+15$. The number of rules in the above system is: $n+4$ rules of type (i), $3n$ rules of type (ii), $3n$ rules of type (iii), $k+n-1$ rules of type (iv), $l+n-1$ rules of type (v), $3n+2max\{k,l\}$ rules of type (vi), $4n+2max\{k,l\}+2$ rules of type (vii) and 5 rules of type $(viii)$. Hence, the size of the constructed system is $max\{\mathcal{O}(n),\mathcal{O}(max\{k,l\})\}$.

Example 2.6. Consider the Knapsack problem with $A=\{a_1,a_2,a_3\}$, $g(a_1)=1$, $g(a_2)=2$, $g(a_3)=1$, $k=2$, $r(a_1)=2$, $r(a_2)=1$, $r(a_3)=3$ and $l=3$. In this case, $n=3$, $k=2$, $l=3$, and the initial configuration of the system is

$$[[\overline{\psi_3}]_M[\alpha\ e_0]_J[\overline{\alpha}]_K[d^2d_0]_1[d^2d_0]_2[a_1\ b_0\ c_0\ \beta_0]_0]_3$$

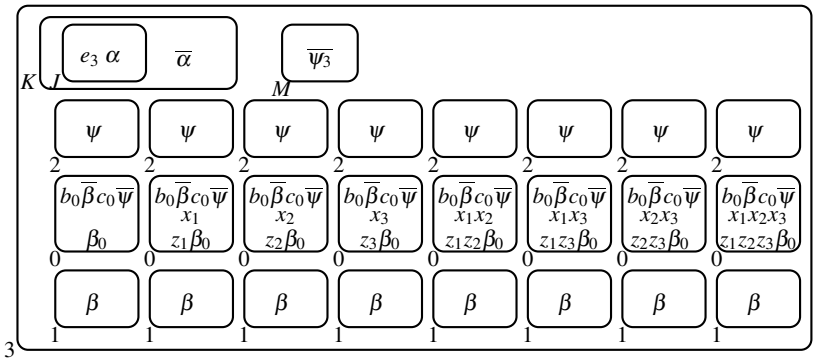
Graphically this is illustrated as:



The evolution of the system is described by the following steps. The working space is generated in $n=3$ steps leading from the initial configuration 1 to configuration 4:

1. $[[\overline{\psi_3}]_M[\alpha\ e_0]_J[\overline{\alpha}]_K[d^2d_0]_1[d^2d_0]_2[a_1\ b_0\ c_0\ \beta_0]_0]_3$
2. $[[\overline{\psi_3}]_M[[\alpha\ e_1]_J\overline{\alpha}]_K[d\ d_0]_1^2[d\ d_0]_2^2[x_1z_1a_2\ b_0\ c_0\ \beta_0]_0[a_2\ b_0\ c_0\ \beta_0]_0]_3$
3. $[[\overline{\psi_3}]_M[\alpha\ e_2]_J[\overline{\alpha}]_K[d_0]_1^4[d_0]_2^4[x_1z_1x_2z_2a_3\ b_0\ c_0\ \beta_0]_0[x_1z_1a_3\ b_0\ c_0\ \beta_0]_0$
$[x_2z_2a_3\ b_0\ c_0\ \beta_0]_0[a_3\ b_0\ c_0\ \beta_0]_0]_3$
4. $[[\overline{\psi_3}]_M[[\alpha\ e_3]_J\overline{\alpha}]_K[\beta]_1^8[\psi]_2^8[x_1z_1x_2z_2x_3z_3\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\beta_0]_0[x_1z_1x_2z_2\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\beta_0]_0$
$[x_1z_1x_3z_3\ b_0\ c_0\overline{\beta}\ \overline{\psi}\beta_0]_0[x_1z_1\ b_0\ c_0\overline{\beta}\ \overline{\psi}\beta_0]_0[x_2z_2x_3z_3\ b_0\ c_0\overline{\beta}\ \overline{\psi}\beta_0]_0[x_2z_2b_0c_0\overline{\beta}\ \overline{\psi}\beta_0]_0$
$[x_3z_3\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\beta_0]_0[b_0\ c_0\ \overline{\beta}\ \overline{\psi}\beta_0]_0]_3$

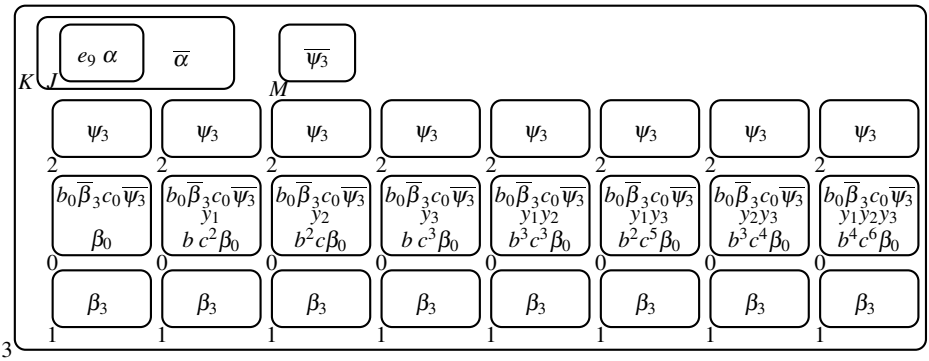
Graphically the working space is described by the following picture:



In the next steps we replace x_i by y_i and $b^{g(a_i)}$. We use y_i to mark the fact that in the membrane 0 containing it there is a subset containing a_i. The multiset of objects $b^{g(a_i)}$ is used to denote the weight of object a_i. In parallel we replace z_i by $c^{r(a_i)}$. The multiset of objects $c^{r(a_i)}$ is used to denote the value of object a_i.

5. $[[\overline{\psi_3}]_M[\alpha\ e_4]_J[\overline{\alpha}]_K[[\beta_1]_1[\underline{\psi_1}]_2 x_1z_1x_2z_2x_3z_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}]_0$
$[[\beta_1]_1[\psi_1]_2 x_1z_1x_2z_2\ \underline{b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0[[\beta_1]_1[\psi_1]_2 x_1z_1x_3z_3\ \underline{b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0$
$[[\beta_1]_1[\psi_1]_2 x_1z_1\ b_0\ c_0\ \underline{\overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0[[\beta_1]_1[\psi_1]_2 x_2z_2x_3z_3\ \underline{b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0$
$[[\beta_1]_1[\psi_1]_2 x_2z_2\ \underline{b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0[[\beta_1]_1[\psi_1]_2 x_3z_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0$
$[[\beta_1]_1[\psi_1]_2 b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0]_3$
6. $[[\overline{\psi_3}]_M[[\alpha\ e_5]_J\overline{\alpha}]_K[\beta_1]_1^8[\psi_1]_2^8[b\ c^2y_1x_2z_2x_3z_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0$
$[b\ c^2y_1x_2z_2\ \underline{b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0[b\ c^2y_1x_3z_3\ \underline{b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0$
$[b\ c^2y_1b_0\ c_0\ \underline{\overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0[x_2z_2x_3z_3\ \underline{b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0$
$[x_2z_2\ b_0\ c_0\ \underline{\overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0[x_3z_3\ b_0\ c_0\ \underline{\overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0[b_0\ c_0\ \underline{\overline{\beta_1}\ \overline{\psi_1}}\beta_0]_0]_3$
7. $[[\overline{\psi_3}]_M[\alpha\ e_6]_J[\overline{\alpha}]_K[[\beta_2]_1[\underline{\psi_2}]_2 b\ c^2y_1x_2z_2x_3z_3\ b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}]_0$
$[[\beta_2]_1[\psi_2]_2 b\ c^2x_2z_2\ \underline{b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0[[\beta_2]_1[\psi_2]_2 b\ c^2x_3z_3\ \underline{b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0$
$[[\beta_2]_1[\psi_2]_2 b\ c^2\ b_0\ c_0\ \underline{\overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0[[\beta_2]_1[\psi_2]_2 x_2z_2x_3z_3\ \underline{b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0$
$[[\beta_2]_1[\psi_2]_2 x_2z_2\ \underline{b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0[[\beta_2]_1[\psi_2]_2 x_3z_3\ b_0\ c_0\ \underline{\overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0$
$[[\beta_2]_1[\psi_2]_2 b_0\ c_0\ \underline{\overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0]_3$
8. $[[\overline{\psi_3}]_M[[\alpha\ e_7]_J\overline{\alpha}]_K[\beta_2]_1^8[\psi_2]_2^8[b^3\ c^3y_1y_2x_3z_3\ b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}\beta_0]_0$
$[b^3\ c^3y_1y_2\ \underline{b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0[b\ c^2y_1x_3z_3\ \underline{b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0$
$[b\ c^2y_1b_0\ c_0\ \underline{\overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0[b^2c\ y_2x_3z_3\ \underline{b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0$
$[b^2c\ y_2\ b_0\ c_0\ \underline{\overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0[x_3z_3\ b_0\ c_0\ \underline{\overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0[b_0\ c_0\ \underline{\overline{\beta_2}\ \overline{\psi_2}}\beta_0]_0]_3$
9. $[[\overline{\psi_3}]_M[\alpha\ e_8]_J[\overline{\alpha}]_K[[\beta_3]_1[\underline{\psi_3}]_2 b^3\ c^3y_1y_2x_3z_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2 b^3\ c^3y_1y_2\ \underline{b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0[[\beta_3]_1[\psi_3]_2 b\ c^2y_1x_3z_3\ \underline{b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2 b\ c^2y_1b_0\ c_0\ \underline{\overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0[[\beta_3]_1[\psi_3]_2 b^2c\ y_2x_3z_3\ \underline{b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2 b^2c\ y_2\ b_0\ c_0\ \underline{\overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0[[\beta_3]_1[\psi_3]_2 x_3z_3\ b_0\ c_0\ \underline{\overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2 b_0\ c_0\ \underline{\overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0]_3$
10. $[[\overline{\psi_3}]_M[[\alpha\ e_9]_J\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b^4\ c^6y_1y_2y_3\ b_0\ c_0\overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b^3\ c^3y_1y_2\ b_0\ c_0\overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[b^2\ c^5y_1y_3\ b_0\ \underline{c_0\ \overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0[b\ c^2y_1b_0\ \underline{c_0\ \overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0[b^3c^4\ y_2y_3\ b_0\ c_0\ \underline{\overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0$
$[b^2c\ y_2\ b_0\ c_0\ \underline{\overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0[b\ c^3y_3\ b_0\ c_0\ \underline{\overline{\beta_3}\ \overline{\psi_3}}\beta_0]_0[b_0\ c_0\ \underline{\overline{\beta_3}\ \overline{\psi_3}}]_0\beta_0]_3$

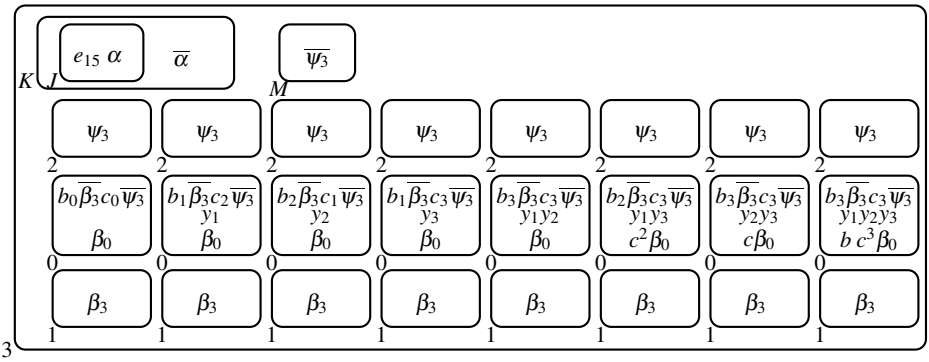
Graphically the working space is described by the following picture:



In what follows we begin to calculate, for $2*max\{k,l\} = 2*max\{2,3\} = 2*3 =$ 6 steps, the weight and value of each membrane 0 by using the objects b_0 and c_0.

11. $[[\overline{\psi_3}]_M[\alpha\, e_{10}]_J[\overline{\alpha}]_K[\beta_3]_1[\psi_3]_2[[\beta_3]_1[\psi_3]_2 b^3\, c^5 y_1y_2y_3\, b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2 b^2\, c^2 y_1y_2\, b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2 b\, c^4 y_1y_3\, b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2 c\, y_1 b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2 b^2 c^3\, y_2y_3\, b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2 b\, y_2\, b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2 c^2 y_3\, b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0]_3$
12. $[[\overline{\psi_3}]_M[[\alpha\, e_{11}]_J\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b^3\, c^5 y_1y_2y_3\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b^2\, c^2 y_1y_2\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[b\, c^4 y_1y_3\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[c\, y_1 b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b^2 c^3\, y_2y_3\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[b\, y_2\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[c^2 y_3\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0]_3$
13. $[[\overline{\psi_3}]_M[\alpha\, e_{12}]_J[\overline{\alpha}]_K[\beta_3]_1^3[\psi_3]_2^2[[\beta_3]_1[\psi_3]_2 b^2\, c^4 y_1y_2y_3\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2 b\, c\, y_1y_2\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2 c^3 y_1y_3\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2 y_1 b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2 b\, c^2\, y_2y_3\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1 y_2\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[[\psi_3]_2 c\, y_3\, b_1\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0]_3$
14. $[[\overline{\psi_3}]_M[[\alpha\, e_{13}]_J\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b^2\, c^4 y_1y_2y_3\, b_2\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b\, c\, y_1y_2\, b_2\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[c^3 y_1y_3\, b_2\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_1 b_1\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b\, c^2\, y_2y_3\, b_2\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[y_2\, b_2\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[c\, y_3\, b_1\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0]_3$
15. $[[\overline{\psi_3}]_M[\alpha\, e_{14}]_J[\overline{\alpha}]_K[\beta_3]_1^5[\psi_3]_2^4[[\beta_3]_1[\psi_3]_2 b\, c^3 y_1y_2y_3\, b_2\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2\, y_1y_2\, b_2\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[[\psi_3]_2 c^2 y_1y_3\, b_2\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_1 b_1\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2 c\, y_2y_3\, b_2\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_2\, b_2\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2 y_3\, b_1\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0]_3$
16. $[[\overline{\psi_3}]_M[[\alpha\, e_{15}]_J\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b\, c^3 y_1y_2y_3\, b_3\, c_3\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_1y_2\, b_3\, c_3\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[c^2 y_1y_3\, b_2\, c_3\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_1 b_1\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[c\, y_2y_3\, b_3\, c_3\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_2\, b_2\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[y_3\, b_1\, c_3\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0]_3$

Graphically the working space is described by the following picture:



Now we divide membranes J and K, after performing an extra step that moves membrane J out of membrane K, in order to check if the index of b_i from the 0 membranes is lower than k, and if the index of c_i from the 0 membranes is greater or equal to l. In parallel, if there still exist membranes 0 containing objects b or c, then the subscript of b_i or c_i, respectively, is increased.

17. $[[\overline{\psi_3}]_M[\alpha e_{15}]_J[\varphi]_K[\beta_3]_1^7[\psi_3]_2^5[[\beta_3]_1[\psi_3]_2 c^2 y_1y_2y_3 b_3 c_3\overline{\beta_3}\,\overline{\psi_3}\beta_0]_0[y_1y_2 b_3 c_3\overline{\beta_3}\,\overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2 c\, y_1y_3\, b_2\, c_3\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_1 b_1\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[[\psi_3]_2 y_2y_3\, b_3\, c_3\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[y_2\, b_2\, c_1\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_3\, b_1\, c_3\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[b_0\, c_0\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0]_3$
18. $[[\overline{\psi_3}]_M[\gamma_1]_J^2[\varphi_1]_K^2[\beta_3]_1^8[\psi_3]_2^8[c^2 y_1y_2y_3\, b_4\, c_4\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_1y_2\, b_3\, c_3\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$
$[c\, y_1y_3\, b_2\, c_4\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_1 b_1\, c_2\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0[y_2y_3\, b_3\, c_4\, \overline{\beta_3}\, \overline{\psi_3}\beta_0]_0$

$[y_2\ b_2\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_3\ b_1\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$

19. $[[\overline{\psi_3}]_M[\gamma_2]^4_J[\varphi_2]^4_K[\beta_3]^8_1[\psi_3]^7_2[[\psi_3]_2 c\ y_1y_2y_3\ b_4\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_1y_2\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2y_1y_3\ b_2\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_1b_1\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_2y_3\ b_3\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[y_2\ b_2\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_3\ b_1\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$

20. $[[\overline{\psi_3}]_M[\beta_3]^8_J[\psi_3]^8_K[\beta_3]^8_1[\psi_3]^8_2[c\ y_1y_2y_3\ b_4\ c_5\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_1y_2\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[y_1y_3\ b_2\ c_5\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_1b_1\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_2y_3\ b_3\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[y_2\ b_2\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_3\ b_1\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$

The next steps are used to generate a *yes* object in membrane M. We present only the final configuration obtained after $4n+2max\{k,l\}+5 = 4*3+2*3+5 = 23$ steps.

24. $[[\beta_3]^5_J[\psi_3]^4_K[\overline{\beta_0}]^2_K[\beta_3]^8_1[\psi_3]^8_2[[c\ y_1y_2y_3\ c_5\ \overline{\psi_3}\beta_0]_0]_J[[y_1y_2\ c_3\ \overline{\psi_3}\beta_0]_0]_J$
$[yes[y_1y_3\ b_2\ c_5\ \overline{\beta_3}\ \overline{\psi_3}]_0\beta_0]_M[[y_1b_1\ \overline{\beta_3}\beta_0]_0]_K[[y_2y_3\ c_4\ \overline{\psi_3}\beta_0]_0]_J$
$[[y_2\ b_2\ \overline{\beta_3}\beta_0]_0]_K[y_3\ b_1\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[b_0\ \overline{\beta_3}\beta_0]_0]_K]_3$

Exercise 2.9. Solve the Knapsack problem using other classes of mobile membranes.

2.5.6 2-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.

Consider $A = \{a_1, \ldots, a_n\}$, and a system of mutual mobile membranes having the initial configuration

$$[[\overline{\alpha}]_M[\alpha\ e_0]_J[\overline{\alpha}]_K[d^{n-1}d_0]_1[d^{n-1}d_0]_2[a_1\ b_0\ c_0\ \beta_0]_0]_3$$

and working over the alphabet

$$V = \{yes, no, b, c, d, d_0, \alpha, \alpha_1, \overline{\alpha}, \beta, \overline{\beta}, \psi, \overline{\psi}\}$$
$$\cup\{a_i, x_i, y_i, z_i, t_i, \psi_i, \overline{\psi}_i \mid 1 \le i \le n\}$$
$$\cup\{\beta_i, \overline{\beta_i} \mid 0 \le i \le n\} \cup \{c_i, b_i \mid 0 \le i \le g(A)\} \cup \{e_i \mid 0 \le i \le 4n + g(a) + 1\}$$

In addition to mutual endocytosis and mutual exocytosis rules, elementary division rules are used to generate all the possible subsets. The system of mutual mobile membranes solving the 2-partition problem uses the rules:

(i) $[a_i]_0 \to [x_i a_{i+1}]_0[z_i a_{i+1}]_0$, for $1 \le i \le n-1$ (div)
$[a_n]_0 \to [x_n\overline{\beta}\ \overline{\psi}]_0[z_n\overline{\beta}\ \overline{\psi}]_0$ (div)
$[d]_j \to [\,]_j[\,]_j$, for $1 \le j \le 2$ (div)
$[d_0]_1 \to [\beta]_1[\beta]_1$ (div)
$[d_0]_2 \to [\psi]_2[\psi]_2$ (div)

The first two rules generate 2^n membranes labelled by 0 containing all the possible subsets over variables $\{x_1, \ldots, x_n\}$. The complementary subset is formed over variables $\{z_1, \ldots, z_n\}$, in a such a manner that x_i and z_i do not exist at the same time inside a membrane, and the total number of x_i and z_j from each membrane after the generation stage is n. The objects x_i and z_i are used to introduce objects

that represent the weight of the objects a_i. In each membrane are placed also two symbols $\overline{\beta}$ and $\overline{\psi}$. The next two rules generate 2^n membranes labelled by 1 each containing an object β and 2^n membranes labelled by 2 each containing an object ψ. The symbols β, $\overline{\beta}$, ψ and $\overline{\psi}$ are used in the mobility of membranes 1 and 2.

(ii) $[\beta]_1[\overline{\beta}]_0 \rightarrow [[\beta_1]_1\overline{\beta_1}]_0$ (mendo)
$[\beta_i]_1[\overline{\beta_i}]_0 \rightarrow [[\beta_{i+1}]_1\overline{\beta_{i+1}}]_0$, for $1 \leq i \leq n-1$ (mendo)
$[[\beta_i]_1 x_i \overline{\beta_i}]_0 \rightarrow [\beta_i]_1[y_i\, b^{g(a_i)}\overline{\beta_i}]_0$, for $1 \leq i \leq n$ (mexo)
$[[\beta_i]_1\overline{\beta_i}]_0 \rightarrow [\beta_i]_1[\overline{\beta_i}]_0$, for $1 \leq i \leq n$ (mexo)
These rules are used to replace x_i by $y_i\, b^{g(a_i)}$, for $1 \leq i \leq n$. If x_i is not present in a membrane 0, then the indexes of β and $\overline{\beta}$ are incremented. After applying these rules, each membrane 0 contains a number of objects b equal to the weight of one partition of A, and also objects y_i used to remember which objects from the first partition are contained in this membrane.

(iii) $[\psi]_2[\overline{\psi}]_0 \rightarrow [[\psi_1]_2\overline{\psi_1}]_0$ (mendo)
$[\psi_i]_2[\overline{\psi_i}]_0 \rightarrow [[\psi_{i+1}]_2\overline{\psi_{i+1}}]_0$, for $1 \leq i \leq n-1$ (mendo)
$[[\psi_i]_2 z_i \overline{\psi_i}]_0 \rightarrow [\psi_i]_2[t_i c^{g(a_i)}\overline{\psi_i}]_0$, for $1 \leq i \leq n$ (mexo)
$[[\psi_i]_2\overline{\psi_i}]_0 \rightarrow [\psi_i]_2[\overline{\psi_i}]_0$, for $1 \leq i \leq n$ (mexo)
In parallel with (ii), these rules are used to replace z_i by $t_i\, c^{g(a_i)}$, for $1 \leq i \leq n$. If z_i is not present in a membrane 0, then the indexes of ψ and $\overline{\psi}$ are incremented. After applying these rules, each membrane 0 contains a number of objects c equal to the weight of the other partition of A, and also objects t_i used to remember which objects from the other partition are contained in this membrane.

(iv) $[\beta_n]_1[b\,\overline{\beta_n}]_0 \rightarrow [[\beta_n]_1\overline{\beta_n}]_0$ (mendo)
$[[\beta_n]_1 b_i\,\overline{\beta_n}]_0 \rightarrow [\beta_n]_1[b_{i+1}\,\overline{\beta_n}]_0$, for $0 \leq i$ (mexo)
These rules are used to calculate the weights of the first partition of A, by using the objects b_0 that appear in all 0 membranes. If an object b is present, then the index of b_i, placed in the same membrane as b, is incremented.

(v) $[\psi_n]_2[c\,\overline{\psi_n}]_0 \rightarrow [[\psi_n]_2\overline{\psi_n}]_0$ (mendo)
$[[\psi_n]_2 c_i\,\overline{\psi_n}]_0 \rightarrow [\psi_n]_2[c_{i+1}\,\overline{\psi_n}]_0$, for $0 \leq i$ (mexo)
In parallel with the rules (iv), these rules are used to calculate the weights of the other partition of A, by using the objects c_0 that appear in all 0 membranes. If an object c is present, then the index of c_i, placed in the same membrane as c, is incremented.

(vi) $[\alpha\, e_i]_J[\overline{\alpha}]_K \rightarrow [[\alpha\, e_{i+1}]_J\overline{\alpha}]_K$ (mendo)
$[[\alpha\, e_i]_J\overline{\alpha}]_K \rightarrow [\alpha\, e_{i+1}]_J[\overline{\alpha}]_K$, for $0 \leq i < 3n+g(A)-1$ (mexo)
$[[\alpha\, e_i]_J\overline{\alpha}]_K \rightarrow [\alpha\, e_{i+1}]_J[\alpha\, e_{i+1}]_K$, for $i = 3n+g(A)-1$ (mexo)
We use these rules in parallel to calculate the number of steps performed. The weight of the set A is given by $g(A) = \sum_{a\in A} g(a)$. The counting stops after $3n+g(A)+1$ steps, a number determined by: generating space (n steps), replacing x_i and z_i by corresponding weights ($2n$ steps) and calculating weights ($g(A)$ steps).

(vii) $[[\alpha\, e_{3n+g(A)}]_J\overline{\alpha}]_K \rightarrow [\alpha\, e_{3n+g(A)}]_J[\alpha\, e_{3n+g(A)}]_K$ (mexo)
$[\alpha\, e_{3n+g(A)}]_J \rightarrow [\gamma_1]_J[\gamma_1]_J$ (div)
$[\gamma_i]_J \rightarrow [\gamma_{i+1}]_J[\gamma_{i+1}]_J$, for $1 \leq i \leq n-2$ (div)
$[\gamma_{n-1}]_J \rightarrow [\beta_n]_J[\beta_n]_J$ (div)

$[\beta_n]_J[\overline{\beta_n}\,\overline{\psi_n}b_ic_j]_0 \to [[\,]_J]_0$, for $i \neq j$ (mendo)
One extra step, to prepare the membrane J for division, is performed if needed. We generate 2^n membranes J to check which membranes respect the weight condition. If membrane J contains an object $e_{3n+g(A)}$, and is placed near membrane K, then 2^n membranes are generated (n steps) in order to check in 2 steps which membranes contain c_i and b_i.

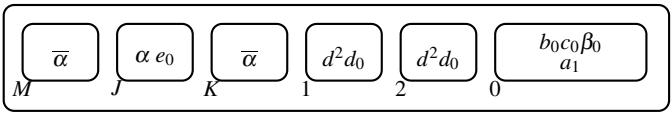
(viii) $[\alpha\, e_i]_K[\overline{\alpha}]_M \to [[\alpha\, e_{i+1}]_K\overline{\alpha}]_M$ for $3n+g(A) \leq i \leq 4n+g(A)$ (mendo)
$[[\alpha\, e_i]_K\overline{\alpha}]_M \to [\underline{\alpha}\, e_{i+1}]_K[\overline{\alpha}]_M$, for $3n+g(A) \leq i < 4n+g(A)$ (mexo)
$[[\underline{\alpha}\, e_i]_K\overline{\alpha}]_M \to [\overline{\beta_0}\beta_0]_K[\alpha_1]_M$, for $i = 4n+g(A)$ or $i = 4n+g(A)+1$ (mexo)
$[\overline{\beta_0}]_K[\beta_0]_0 \to [\underline{yes}[\,]_0]_K$ (mendo)
$[\alpha_1]_M \to [\overline{\beta_0}]_M[\overline{\beta_0}]_M$ (div)
$[\beta_0]_K[\overline{\beta_0}]_M \to [[no]_K]_M$ (mendo)
The above rules are used in parallel to calculate the number of steps performed. The number $4n+g(A)+1$ is determined by: generating space (n steps), replacing x_i and z_i by corresponding weights ($2n$ steps) and calculating weights ($g(A)$ steps), generating J membranes ($n-1$ steps), preparing division of J (1 step), blocking all membranes 0 that do not satisfy conditions (1 step). If there still exists a membrane 0 that is not inside a membrane J, then the object *yes* is created inside membrane K. Otherwise, after one more step, the *no* object is created inside membrane K. The computation stops after maximum $4n+g(A)+3$ steps, with the answer placed inside membrane K.

The number of membranes in the initial configuration is 6, and the number of objects is $2n+5$. The size of the working alphabet is $13n+3g(A)+15$. The number of rules in the above system is: $n+4$ rules of type (i), $3n$ rules of type (ii), $3n$ rules of type (iii), $2g(A)$ rules of type (iv), $2g(A)$ rules of type (v), $3n+g(A)$ rules of type (vi), $n+2$ rules of type (vii) and $2n+5$ rules of type $(viii)$. Hence, the size of the constructed system of mutual mobile membranes is $max\{\mathcal{O}(n), \mathcal{O}(g(A))\}$.

Example 2.7. Consider the 2-partition problem with $A = \{a_1, a_2, a_3\}$, $g(a_1) = 1$, $g(a_2) = 3$ and $g(a_3) = 2$. In this case, $n = 3$, $g(A) = 6$, and the initial configuration of the system of mutual mobile membranes

$$[[\overline{\alpha}]_M[\alpha\, e_0]_J[\overline{\alpha}]_K[d^2d_0]_1[d^2d_0]_2[a_1b_0\, c_0\, \beta_0]_0]_3$$

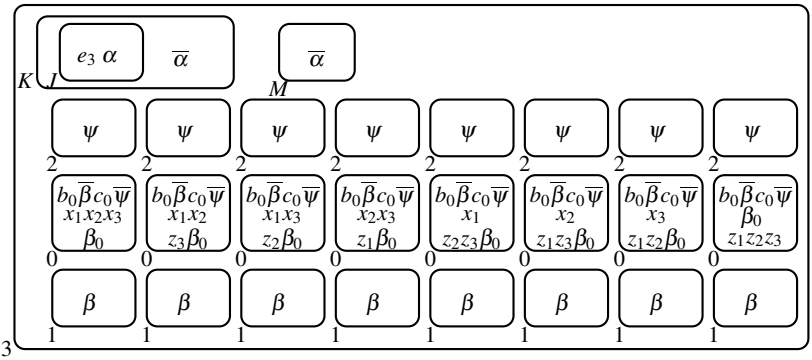
Graphically this is illustrated as:



The evolution of the system is described by the following steps. The working space is generated in $n = 3$ steps leading from the initial configuration 1 to configuration 4.

1. $[[\overline{\alpha}]_M[\alpha\, e_0]_J[\overline{\alpha}]_K[d^2d_0]_1[d^2d_0]_2[a_1b_0\, c_0\, \beta_0]_0]_3$
2. $[[\overline{\alpha}]_M[[\alpha\, e_1]_J\overline{\alpha}]_K[d\, d_0]_1^2[d\, d_0]_2^2[x_1a_2\, b_0\, c_0\, \beta_0]_0[z_1a_2\, b_0\, c_0\, \beta_0]_0]_3$
3. $[[\overline{\alpha}]_M[\alpha\, e_2]_J[\overline{\alpha}]_K[d_0]_1^4[d_0]_2^4[x_1x_2a_3\, b_0\, c_0\, \beta_0]_0[x_1z_2a_3\, b_0\, c_0\, \beta_0]_0$
$[z_1x_2a_3\, b_0\, c_0\, \beta_0]_0[z_1z_2a_3\, b_0\, c_0\, \beta_0]_0]_3$

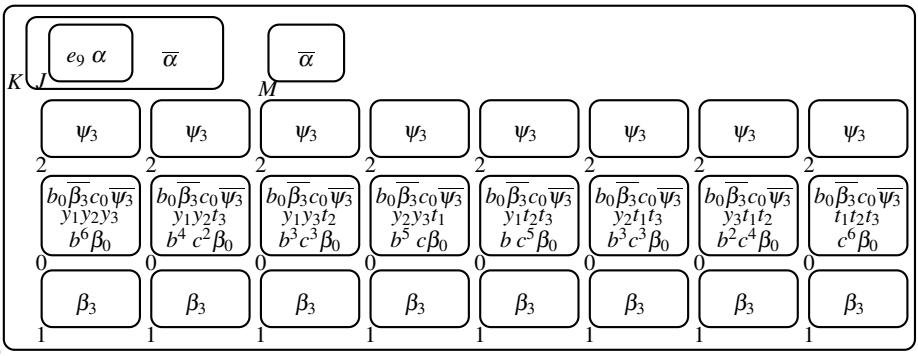
4. $[[\overline{\alpha}]_M[[\alpha\ e_3]_J\overline{\alpha}]_K[\beta]_1^8[\psi]_2^8[x_1x_2x_3\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\ \beta_0]_0[x_1x_2z_3\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\ \beta_0]_0$
$[x_1z_2x_3\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\ \beta_0]_0[x_1z_2z_3\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\ \beta_0]_0[z_1x_2x_3\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\ \beta_0]_0$
$[z_1x_2z_3\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\ \beta_0]_0[z_1z_2x_3\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\ \beta_0]_0[z_1z_2z_3\ b_0\ c_0\ \overline{\beta}\ \overline{\psi}\ \beta_0]_0]_3$
Graphically the working space is described by the following picture:



We notice from this image that each partition appears twice, but this does not increase the number of steps performed. In the next steps we replace x_i by y_i and $b^{g(a_i)}$. We use y_i to mark the fact that in the membrane 0 containing it there is a subset containing a_i. The multiset of objects $b^{g(a_i)}$ is used to denote the weight of object a_i. In parallel we replace z_j by t_j. We use t_j to mark the fact that in the membrane 0 containing it there is a subset containing a_j.

5. $[[\overline{\alpha}]_M[\alpha\ e_4]_J[\overline{\alpha}]_K[[\beta_1]_1[\underline{\psi_1}]_2x_1x_2x_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0$
$[[\beta_1]_1[\psi_1]_2x_1x_2z_3\ b_0\ c_0\ \underline{\overline{\beta_1}}\ \overline{\psi_1\beta_0}]_0[[\beta_1]_1[\psi_1]_2x_1z_2x_3\ b_0\ c_0\ \underline{\overline{\beta_1}}\ \overline{\psi_1\beta_0}]_0$
$[[\beta_1]_1[\psi_1]_2x_1z_2z_3\ b_0\ c_0\ \underline{\overline{\beta_1}}\ \psi_1\beta_0]_0[[\beta_1]_1[\psi_1]_2z_1x_2x_3\ b_0\ c_0\ \underline{\overline{\beta_1}}\ \overline{\psi_1}\beta_0]_0$
$[[\beta_1]_1[\psi_1]_2z_1x_2z_3\ b_0\ c_0\ \underline{\overline{\beta_1}}\ \overline{\psi_1}\beta_0]_0[[\beta_1]_1[\psi_1]_2z_1z_2x_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0$
$[[\beta_1]_1[\psi_1]_2z_1z_2z_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0]_3$
6. $[[\overline{\alpha}]_M[[\alpha\ e_5]_J\overline{\alpha}]_K[\beta_1]_1^8[\psi_1]_2^8[b\ y_1x_2x_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0[b\ y_1x_2z_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0$
$[by_1z_2x_3\ b_0\ c_0\ \underline{\overline{\beta_1}}\ \overline{\psi_1}\beta_0]_0[b\ y_1z_2z_3\ b_0\ c_0\ \underline{\overline{\beta_1}}\ \overline{\psi_1}\beta_0]_0[c\ t_1x_2x_3\ b_0\ c_0\ \underline{\overline{\beta_1}}\ \overline{\psi_1}\beta_0]_0$
$[c\ t_1x_2z_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0[c\ t_1z_2x_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0[c\ t_1z_2z_3\ b_0\ c_0\ \overline{\beta_1}\ \overline{\psi_1}\beta_0]_0]_3$
7. $[[\overline{\alpha}]_M[\alpha\ e_6]_J[\overline{\alpha}]_K[[\beta_2]_1[\underline{\psi_2}]_2b\ y_1x_2x_3\ b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}\beta_0]_0$
$[[\beta_2]_1[\psi_2]_2b\ y_1x_2z_3\ b_0\ c_0\ \underline{\overline{\beta_2}}\ \overline{\psi_2}\beta_0]_0[[\beta_2]_1[\psi_2]_2by_1z_2x_3\ b_0\ c_0\ \underline{\overline{\beta_2}}\ \overline{\psi_2}\beta_0]_0$
$[[\beta_2]_1[\psi_2]_2b\ y_1z_2z_3\ b_0\ c_0\ \underline{\overline{\beta_2}}\ \overline{\psi_2}\beta_0]_0[[\beta_2]_1[\psi_2]_2c\ t_1x_2x_3\ b_0\ c_0\ \underline{\overline{\beta_2}}\ \overline{\psi_2}\beta_0]_0$
$[[\beta_2]_1[\psi_2]_2c\ t_1x_2z_3\ b_0\ c_0\ \underline{\overline{\beta_2}}\ \overline{\psi_2}\beta_0]_0[[\beta_2]_1[\psi_2]_2c\ t_1z_2x_3\ b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}\beta_0]_0$
$[[\beta_2]_1[\psi_2]_2c\ t_1z_2z_3\ b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}\beta_0]_0]_3$
8. $[[\overline{\alpha}]_M[[\alpha\ e_7]_J\overline{\alpha}]_K[\beta_2]_1^8[\psi_2]_2^8[b^4\ y_1y_2x_3\ b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}\beta_0]_0[b^4\ y_1y_2z_3\ b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}\beta_0]_0$
$[b\ c^3y_1t_2x_3\ b_0\ c_0\ \underline{\overline{\beta_2}}\ \overline{\psi_2}\beta_0]_0[b\ c^3y_1t_2z_3\ b_0\ c_0\ \underline{\overline{\beta_2}}\ \overline{\psi_2}\beta_0]_0[b^3c\ t_1y_2x_3\ b_0\ c_0\ \underline{\overline{\beta_2}}\ \overline{\psi_2}\beta_0]_0$
$[b^3c\ t_1y_2z_3\ b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}\beta_0]_0[c^4\ t_1t_2x_3\ b_0\ c_0\ \underline{\overline{\beta_2}}\ \overline{\psi_2}\beta_0]_0[c^4\ t_1t_2z_3\ b_0\ c_0\ \overline{\beta_2}\ \overline{\psi_2}\beta_0]_0]_3$
9. $[[\overline{\alpha}]_M[\alpha\ e_8]_J[\overline{\alpha}]_K[[\beta_3]_1[\underline{\psi_3}]_2b^4\ y_1y_2x_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2b^4\ y_1y_2z_3\ b_0\ c_0\ \underline{\overline{\beta_3}}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2b\ c^3y_1t_2x_3\ b_0\ c_0\ \underline{\overline{\beta_3}}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2b\ c^3y_1t_2z_3\ b_0\ c_0\ \underline{\overline{\beta_3}}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2b^3c\ t_1y_2x_3\ b_0\ c_0\ \underline{\overline{\beta_3}}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2b^3c\ t_1y_2z_3\ b_0\ c_0\ \underline{\overline{\beta_3}}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2c^4\ t_1t_2x_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2c^4\ t_1t_2z_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$

10. $[[\overline{\alpha}]_M[[\alpha\ e_9]_J\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b^6\ y_1y_2y_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b^4c^2\ y_1y_2t_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[b^3\ c^3y_1t_2y_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b\ c^5y_1t_2t_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b^5c\ t_1y_2y_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[b^3c^3\ t_1y_2t_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b^2c^4\ t_1t_2y_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c^6\ t_1t_2t_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$
Graphically the working space is described by the following picture:



In what follows we calculate in $g(A) = 1+3+2 = 6$ steps, the weights of the subsets (over $\{y_1,y_2,y_3\}$) and complementary subsets (over $\{t_1,t_2,t_3\}$) from each membrane 0 by using the objects b_0 and c_0.

11. $[[\overline{\alpha}]_M[\alpha\ e_{10}]_J[\overline{\alpha}]_K[\beta_3]_1[\psi_3]_2[[\beta_3]_1b^5\ y_1y_2y_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2b^3c\ y_1y_2t_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2b^2\ c^2y_1t_2y_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2c^4y_1t_2t_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2b^4t_1y_2y_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2b^2c^2\ t_1y_2t_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2b\ c^3\ t_1t_2y_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2c^5\ t_1t_2t_3\ b_0\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$

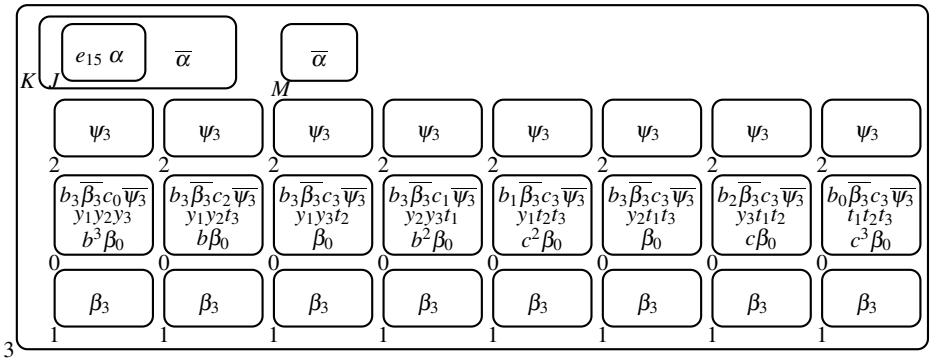
12. $[[\overline{\alpha}]_M[[\alpha\ e_{11}]_J\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b^5\ y_1y_2y_3\ b_1\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b^3c\ y_1y_2t_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[b^2\ c^2y_1t_2y_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c^4y_1t_2t_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b^4t_1y_2y_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[b^2c^2\ t_1y_2t_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b\ c^3\ t_1t_2y_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c^5\ t_1t_2t_3\ b_0\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$

13. $[[\overline{\alpha}]_M[\alpha\ e_{12}]_J[\overline{\alpha}]_K[\beta_3]_1^2[\psi_3]_2^2[[\beta_3]_1b^4\ y_1y_2y_3\ b_1\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2b^2y_1y_2t_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2b\ c\ y_1t_2y_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2c^3y_1t_2t_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1b^3t_1y_2y_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2b\ c\ t_1y_2t_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2c^2\ t_1t_2y_3\ b_1\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2c^4\ t_1t_2t_3\ b_0\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$

14. $[[\overline{\alpha}]_M[[\alpha\ e_{13}]_J\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b^4\ y_1y_2y_3\ b_2\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b^2y_1y_2t_3\ b_2\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[b\ c\ y_1t_2y_3\ b_2\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c^3y_1t_2t_3\ b_1\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b^3t_1y_2y_3\ b_2\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[b\ c\ t_1y_2t_3\ b_2\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c^2\ t_1t_2y_3\ b_2\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c^4\ t_1t_2t_3\ b_0\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$

15. $[[\overline{\alpha}]_M[\alpha\ e_{14}]_J[\overline{\alpha}]_K[\beta_3]_1^3[\psi_3]_2^3[[\beta_3]_1b^3\ y_1y_2y_3\ b_2\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1b\ y_1y_2t_3\ b_2\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1[\psi_3]_2y_1t_2y_3\ b_2\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2c^2y_1t_2t_3\ b_1\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\beta_3]_1b^2t_1y_2y_3\ b_2\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1[\psi_3]_2t_1y_2t_3\ b_2\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\psi_3]_2c\ t_1t_2y_3\ b_2\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2c^3\ t_1t_2t_3\ b_0\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$

16. $[[\overline{\alpha}]_M[[\alpha\ e_{15}]_J\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b^3\ y_1y_2y_3\ b_3\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b\ y_1y_2t_3\ b_3\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[y_1t_2y_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c^2y_1t_2t_3\ b_1\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b^2t_1y_2y_3\ b_3\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[t_1y_2t_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c\ t_1t_2y_3\ b_2\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c^3\ t_1t_2t_3\ b_0\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$
Graphically the working space is described by the following picture:



Now we divide membrane J, after performing an extra step that moves membrane J out of membrane K, in order to check whether the indexes of b_i and c_j from the 0 membranes are equal or not. In parallel, if there still exist membranes 0 containing objects b or c, then the subscript of b_i or c_i, respectively, is increased.

17. $[[\overline{\alpha}]_M[\alpha\ e_{15}]_J[\alpha\ e_{15}]_K[\beta_3]_1^5[\psi_3]_2^5[[\beta_3]_1 b^2\ y_1y_2y_3\ b_3\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1 y_1y_2t_3\ b_3\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_1t_2y_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\psi_3]_2 c\ y_1t_2t_3\ b_1\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1 b\ t_1y_2y_3\ b_3\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[t_1y_2t_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\psi_3]_2 t_1t_2y_3\ b_2\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2 c^2\ t_1t_2t_3\ b_0\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$
18. $[[\overline{\alpha}[\alpha\ e_{16}]_K]_M[\gamma_1]_J^2[\beta_3]_1^8[\psi_3]_2^8[b^2\ y_1y_2y_3\ b_4\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_1y_2t_3\ b_4\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[y_1t_2y_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c\ y_1t_2t_3\ b_1\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[b\ t_1y_2y_3\ b_4\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[t_1y_2t_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[t_1t_2y_3\ b_2\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c^2\ t_1t_2t_3\ b_0\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$
19. $[[\overline{\alpha}]_M[\alpha\ e_{17}]_K[\gamma_2]_J^4[\overline{\alpha}]_K[\beta_3]_1^6[\psi_3]_2^6[[\beta_3]_1 b\ y_1y_2y_3\ b_4\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[y_1y_2t_3\ b_4\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_1t_2y_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[[\psi_3]_2 y_1t_2t_3\ b_1\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\beta_3]_1 t_1y_2y_3\ b_4\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[t_1y_2t_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[t_1t_2y_3\ b_2\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[[\psi_3]_2 c\ t_1t_2t_3\ b_0\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$
20. $[[\overline{\alpha}[\alpha\ e_{18}]_K]_M[\beta_3]_J^8[\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b\ y_1y_2y_3\ b_5\ c_0\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_1y_2t_3\ b_4\ c_2\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[y_1t_2y_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_1t_2t_3\ b_1\ c_5\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[t_1y_2y_3\ b_5\ c_1\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[t_1y_2t_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[t_1t_2y_3\ b_2\ c_4\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[c\ t_1t_2t_3\ b_0\ c_5\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0]_3$
21. $[[\alpha_1]_M[\beta_0\overline{\beta_0}]_K[\beta_3]_J^8[\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b\ y_1y_2y_3\beta_0[\ ]_J]_0[y_1y_2t_3\beta_0[\ ]_J]_0$
$[y_1t_2y_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0[y_1t_2t_3\ \beta_0[\ ]_J]_0[t_1y_2y_3\ \beta_0[\ ]_J]_0[t_1y_2t_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[t_1t_2y_3\ \beta_0[\ ]_J]_0[c\ t_1t_2t_3\ \beta_0[\ ]_J]_0]_3$
The next step is used to generate a *yes* object in membrane M. The computation stops after $4n+g(A)+2=4*3+6+2=20$ steps.
22. $[[\overline{\beta_0}]_M^2[\beta_3]_J^8[\overline{\alpha}]_K[\beta_3]_1^8[\psi_3]_2^8[b\ y_1y_2y_3\beta_0[\ ]_J]_0[y_1y_2t_3\beta_0[\ ]_J]_0[y_1t_2y_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}\beta_0]_0$
$[y_1t_2t_3\ \beta_0[\ ]_J]_0[t_1y_2y_3\ \beta_0[\ ]_J]_0[yes\ \beta_0[t_1y_2t_3\ b_3\ c_3\ \overline{\beta_3}\ \overline{\psi_3}]_0]_K$
$[t_1t_2y_3\ \beta_0[\ ]_J]_0[c\ t_1t_2t_3\ \beta_0[\ ]_J]_0]_3$

Exercise 2.10. Solve the 2-Partition problem using other classes of mobile membranes.

2.6 Decidability Results

In [11] we investigate the problem of reaching a configuration from another configuration in a special class of systems of mobile membranes. We prove that reachability can be decided by reducing it to the reachability problem of a version of pure and public ambient calculus without the capability open. The relationship between mobile ambients and mobile membranes is presented in Section 3.3.

Reachability is the problem of deciding whether a system may reach a given configuration during its execution. This is one of the most critical properties in the verification of systems; most of the safety properties of computing systems can be reduced to the problem of checking whether a system may reach an "unintended state".

In what follows we investigate the problem of reaching a certain configuration in systems of mobile membranes starting from a given configuration. We prove that reachability in systems of mobile membranes can be decided by reducing it to the reachability problem of a version of pure and public ambient calculus from which the open capability has been removed. It is proven in [28] that reachability for this fragment of ambient calculus is decidable by reducing it to marking reachability for Petri nets, which is proven to be decidable in [115]. The reachability problem is investigated in [82] for other classes of P systems, namely for extensions of PB systems with volatile membranes.

When working with Petri nets, reachability is a property of general interest. Given a net with initial marking ω_0, we say that the marking ω is reachable if there exists a sequence of firings $\omega_0 \rightarrow \omega_1 \rightarrow \ldots \omega_n = \omega$ of the net. The reachability problems is decidable in Petri nets, even if they tend to have a very large complexity in practice. A good survey of the known decidability issues for Petri nets is given in [83].

2.6.1 Mobile Membranes with Replication

Since we use reduction to mobile ambients, we construct a class of systems of mobile membranes in which replication from mobile ambients is expressed explicitly by duplicating objects or membranes in systems of mobile membranes.

Definition 2.13. A system of n mobile membranes with replication rules is a structure

$$\prod = (V \cup \overline{V}, H \cup \overline{H}, \mu, w_1, \ldots, w_n, R), \text{ where:}$$

1. $n \geq 1$ represents the initial degree of the system;
2. $V \cup \overline{V}$ is an alphabet (its elements are called objects), where $V \cap \overline{V} = \emptyset$;
3. $H \cup \overline{H}$ is a finite set of labels for membranes, where $H \cap \overline{H} = \emptyset$;
4. $\mu \subseteq H \times H$ describes the membrane structure, such that $(i, j) \in \mu$ denotes that the membrane labelled by j is contained in the membrane labelled by i; we distinguish the external membrane (usually called the "skin" membrane) and several

internal membranes; a membrane without any other membrane inside it is said to be elementary;

5. $w_1, w_2, \ldots, w_n$ are multisets of objects from $V \cup \overline{V}$ placed in the n membranes of the system;
6. R is a finite set of developmental rules, of the following forms:

   a. $[\overline{a}\downarrow \rightarrow \overline{a}\downarrow\, a\downarrow]_h$, for $h \in H$, $a\downarrow \in V$, $\overline{a}\downarrow \in \overline{V}$; replication rule
   The objects $\overline{a}\downarrow$ are used to create new objects $a\downarrow$ without being consumed.
   b. $[\overline{a}\downarrow\, a\downarrow \rightarrow \overline{a}\downarrow]_h$, for $h \in H$, $a\downarrow \in V$, $\overline{a}\downarrow \in \overline{V}$; consumption rule
   The objects $a\downarrow$ are consumed.
   c. $[\overline{a}\uparrow \rightarrow \overline{a}\uparrow\, a\uparrow]_h$, for $h \in H$, $a\uparrow \in V$, $\overline{a}\uparrow \in \overline{V}$; replication rule
   The objects $\overline{a}\uparrow$ are used to create new objects $a\uparrow$ without being consumed.
   d. $[\overline{a}\uparrow\, a\uparrow \rightarrow \overline{a}\uparrow]_h$, for $h \in H$, $a\uparrow \in V$, $\overline{a}\uparrow \in \overline{V}$; consumption rule
   The objects $a\uparrow$ are consumed.
   e. $[\, a\downarrow\,]_h\, [\,]_a \rightarrow [\, [\,]_h\,]_a$, for $a, h \in H, a\downarrow \in V$; endocytosis
   An elementary membrane labelled h (containing an object $a\downarrow$) enters the adjacent membrane labelled a. The labels h and a remain unchanged during this process; however the object $a\downarrow$ is consumed during the operation. Membrane a is not necessarily elementary.
   f. $[\, [\, a\uparrow\,]_h\,]_a \rightarrow [\,]_h\, [\,]_a$, for $a, h \in H, a\uparrow \in V$; exocytosis
   An elementary membrane labelled h (containing an object $a\uparrow$) is sent out of a membrane labelled a. The labels of the two membranes remain unchanged; the object $a\uparrow$ of membrane h is consumed during the operation. Membrane a is not necessarily elementary.
   g. $[\,]_{\overline{h}} \rightarrow [\,]_{\overline{h}} [\,]_h$ for $h \in H$, $\overline{h} \in \overline{H}$ division rules
   An elementary membrane labelled $\overline{h}$ is divided into two membranes labelled by $\overline{h}$ and h and having the same objects.

$V \cap \overline{V} = \emptyset$ states that objects from $\overline{V}$ can participate only in the rules of type $(a) - (d)$. Similarly, $H \cap \overline{H} = \emptyset$ states that membranes having labels from the set $\overline{H}$ can participate only in rules of type (g).

The rules are applied using the following principles:

1. In biological systems molecules are divided into classes of different types. We make the same decision here and split the objects into four classes: $a \downarrow$ - objects which control the *endocytosis*, $a \uparrow$ - objects which control the *exocytosis*, and $\overline{a} \downarrow$, $\overline{a} \uparrow$ - objects which produce new objects of the first two classes without being consumed.
2. All the rules are applied in parallel, non-deterministically choosing the rules, the membranes, and the objects in such a way that the parallelism is maximal; this means that in each step we apply a set of rules such that no further rule, no further membranes and objects can evolve at the same time.
3. A membrane a from each rule of type (e) and (f) is said to be *passive*, while membrane h is said to be *active*. In any step of a computation, any object and any active membrane can be involved in at most one rule, but the passive membranes

are not considered involved in the use of rules (hence they can be used by several rules at the same time).
4. When a membrane is moved across another membrane, by endocytosis or exocytosis, its whole content (its objects) is moved.
5. If a membrane is divided, then its content is replicated in the two new copies.
6. The skin membrane can never be divided.

According to these rules, we get transitions among the configurations of the system. For two systems of mobile membranes M and N, we say that M reduces to N if there is a sequence of rules applicable in the system of mobile membranes M in order to obtain the system of mobile membranes N.

In what follows we prove that the problem of reaching a configuration starting from a certain configuration is decidable for the systems of mobile membranes from Definition 2.13.

Theorem 2.21. *For two arbitrary systems of mobile membranes with replication rules M_1 and M_2, it is decidable whether M_1 reduces to M_2.*

The main steps of the proof are as follows:

1. systems of mobile membranes are reduced to pure and public mobile ambients without the capability *open*;
2. the reachability problem for two arbitrary systems of mobile membranes is expressed as the reachability problem for the corresponding mobile ambients.
3. the reachability problem is decidable for a fragment of pure and public mobile ambients without the capability *open*.

The rest of this section is devoted to the proof of Theorem 2.21.

2.6.2 From Mobile Membranes to Mobile Ambients

We use the following translation steps:

1. any object $a\downarrow$ is translated into a capability *in* a;
2. any object $a\uparrow$ is translated into a capability *out* a;
3. any object $\overline{a}\downarrow$ is translated into a replication !*in* a
4. any object $\overline{a}\uparrow$ is translated into a replication !*out* a
5. a membrane h is translated into an ambient h
6. an elementary membrane $\overline{h}$ is translated into a replication $!h[\,]$ while all the objects inside membrane h are translated into capabilities in ambient h using the above steps.

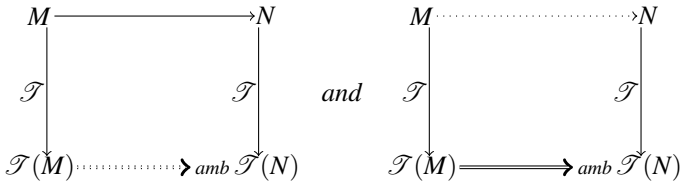
A correspondence exists between the rules of the systems of mobile membranes and the reduction rules of the mobile ambients as follows:

- rule (c) corresponds to rule **(In)**;

- rule (d) corresponds to rule **(Out)**;
- rules $(a),(b),(e)$ correspond to instances of rule **(Repl)**.

The rule **(Repl)** from mobile ambients has the form $A \Rightarrow_{amb} !A \mid A$. If we start with a system of mobile membranes M, we denote by $\mathscr{T}(M)$ the mobile ambient obtained using the above translation steps. For example, starting from the system of mobile membranes $M = [m\downarrow\ m\uparrow]_n[\ ]_m$ we obtain $\mathscr{T}(M) = n[in\ m \mid out\ m] \mid m[\ ]$.

Proposition 2.2. *For two systems of mobile membranes M and N, M reduces to N by applying one rule if and only if $\mathscr{T}(M)$ reduces to $\mathscr{T}(N)$ by applying only one reduction rule.*



Proof (Sketch). Since M reduces to N by applying one rule, then one of the rules of type $(a),\ldots,(e)$ is applied. We treat only the case when a rule of type (a) is applied, the others being treated in a similar manner.

If a rule $\overline{a}\downarrow \rightarrow \overline{a}\downarrow\ a\downarrow$ is applied, only one object from the system of mobile membranes M is used (namely $\overline{a}\downarrow$) to create a new object $a\downarrow$, thus obtaining the system of mobile membranes N. By translating the system of mobile membranes M into $\mathscr{T}(M)$, we have that $\overline{a}\downarrow$ is translated into $!in\ a$. By applying the reduction rule corresponding to (a) (namely the rule **(Repl)**) to $!in\ a$, we have that $!in\ a \Rightarrow_{amb} in\ a \mid !in\ a$, and so a new capability $in\ a$ is created. We observe that $\mathscr{T}(\overline{a}\downarrow\ a\downarrow) = !in\ a \mid in\ a$, which means that the obtained mobile ambient is $\mathscr{T}(N)$ (in fact it is structurally congruent to $\mathscr{T}(N)$). □

According to Proposition 2.2 the reachability problem for systems of mobile membranes can be reduced to a similar problem for mobile ambients.

2.6.3 From Mobile Ambients to Petri Nets

After translating the systems of mobile membranes into a fragment of mobile ambients, we present the algorithm used in [28] to translate this fragment of mobile ambients into a fragment of Petri nets which is known to be decidable from [115]. The fragment of mobile ambients used here is a subset of the fragment of mobile ambients used in [28] and the difference is provided by the extra rule $!A \Rightarrow_{amb} !A \mid !A$ used in [28].

We observe that applying a reduction rule over a process either increases the number of ambients or leaves it unchanged. The only reduction rule which increases the number of ambients when applied is the rule **(Repl)**, while the other reduction

rules leave the number of ambients unchanged. If we reach process B starting from process A, then the number of ambients of process B is known. Therefore, we can use this information to know how many times the reduction rule **(Repl)** is applied to replicate ambients. A similar argument does not hold for capabilities as they can be consumed by the reduction rules **(In)** and **(Out)**.

An ambient context $\mathscr{C}$ is a process in which some holes may occur (denoted by $\Box$). Using the ambient contexts, we split a process into two parts: one is a context containing ambients, whereas the other is a process without ambients. In order to uniquely identify all the occurrences of replication, ambient, capability or hole $\Box$ within an ambient context or a process, we introduce a labelling system. Using a countable set of labels, we say that a process A or an ambient context $\mathscr{C}$ is well-labelled if any label occurs at most once in A or $\mathscr{C}$. We denote by $Amb(\mathscr{C})$ the multiset of ambients occurring in an ambient context $\mathscr{C}$. We say that two processes are label-free-equivalent if after removing all the labels from the two processes, they are structurally congruent.

2.6.3.1 I) Labelled Transition System.

For the reachability problem $A \Rightarrow^* B$, we denote by $\mathscr{C}_A$ a well-labelled ambient context, and by θ_A a mapping from the set of holes in $\mathscr{C}_A$ to some labelled processes without replicable ambients such that $\theta_A(\mathscr{C}_A)$ is well-labelled, and $\theta_A(\mathscr{C}_A) = A$ where labels are ignored.

A labelled transition system $L_{A,B}$ describes all possible reductions for a context $\mathscr{C}_A$: this includes reductions of replications and capabilities contained in $\mathscr{C}_A$ and in the processes associated with the holes of the context. The states of the labelled transition system $L_{A,B}$ are associative-commutative equivalent classes of ambient contexts and, for simplicity, we often identify a state as one of the representatives of its class.

We define a mapping $\theta_{L_{A,B}}$ which extends the mapping θ_A. Initially, $L_{A,B}$ contains (the equivalence class of) $\mathscr{C}_A$ as a unique state, and we have $\theta_{L_{A,B}} = \theta_A$. We present in what follows the construction steps of $\theta_{L_{A,B}}$, where cap stands for *in* or *out*:

1. For any ambient context $\mathscr{C}$ from $L_{A,B}$ and for any labelled capability $cap^w n$ in $\mathscr{C}$, if this capability can be executed using one of the rules **(In)** or **(Out)** leading to some ambient context $\mathscr{C}'$, then a state $\mathscr{C}'$ and a transition from $\mathscr{C}$ to $\mathscr{C}'$ labelled by $cap^w n$ are added to $L_{A,B}$.
2. For any ambient context $\mathscr{C}$ from $L_{A,B}$ and for any labelled replication $!^w$ in $\mathscr{C}$ such that the reduction rule **(Repl)** is applied, we define the ambient context $\mathscr{C}'$ as follows: $\mathscr{C}'$ is identical to $\mathscr{C}$ except that the subcontext $!^w\mathscr{C}_a$ in $\mathscr{C}$ is replaced by $!^w\mathscr{C}_a \mid \gamma(\mathscr{C}_a)$ in $\mathscr{C}'$; the mapping γ relabels $\mathscr{C}_a$ with fresh labels, such that $\mathscr{C}'$ is well-labelled. If $Amb(\mathscr{C}') \subseteq Amb(B)$, then state $\mathscr{C}'$ and a transition from $\mathscr{C}$ to $\mathscr{C}'$ labelled by $!^w$ is added to $L_{A,B}$. Additionally, we define $\theta'_{L_{A,B}}$ as an extension of $\theta_{L_{A,B}}$ such that for all $\Box^{w'}$ in $\mathscr{C}_a$ we have:

   (i) $\theta'_{L_{A,B}}(\gamma(\Box^{w'}))$ and $\theta_{L_{A,B}}(\Box^{w'})$ are label-free-equivalent,

(ii) labels in $\theta'_{L_{A,B}}(\gamma(\square^{w'}))$ are fresh in the currently built transition system $L_{A,B}$,
(iii) $\theta'_{L_{A,B}}(\gamma(\square^{w'}))$ is well-labelled.

Finally, we set $\theta_{L_{A,B}}$ to be $\theta'_{L_{A,B}}$.

3. For any ambient context $\mathscr{C}$ from $L_{A,B}$, for any labelled hole $\square^w$ in $\mathscr{C}$ and for any capability $\mathsf{cap}^w\mathsf{n}$ in the process $\theta_{L_{A,B}}(\square^w)$, we consider the ambient context $\mathscr{C}_m$ identical to $\mathscr{C}$ except that $\square^w$ in $\mathscr{C}$ has been replaced by $\square^w \mid cap^w n$ in $\mathscr{C}_m$. If the capability $cap^w n$ can be consumed in $\mathscr{C}_m$ using one of the rules **(In)** or **(Out)** leading to an ambient context $\mathscr{C}'$, then state $\mathscr{C}'$ and a transition from $\mathscr{C}$ to $\mathscr{C}'$ labelled by $cap^w n$ are added to transition system $L_{A,B}$.
4. For any ambient context $\mathscr{C}$ from $L_{A,B}$ and for any labelled hole $\square^w$ in $\mathscr{C}$ associated by $\theta_{L_{A,B}}$ with a process of the form $!^{w'}A'$, if a replication $!^{w'}$ can be reduced in process $\theta_{L_{A,B}}(\mathscr{C})$ using rule **(Repl)**, then a transition from $\mathscr{C}$ to itself labelled by $!^{w''}$ is added to $L_{A,B}$ for any replication $!^{w''}$ in $\theta_{L_{A,B}}(\square^w)$.

In the second step, the reduction of a replication contained in the ambient context by means of the rule **(Repl)** is done only when the number of ambients in the resulting process is smaller than the number of ambients in the target process B, namely $Amb(\mathscr{C}') \subseteq Amb(B)$. This requirement is crucial as it implies that the transition system $L_{A,B}$ has only finitely many states.

As an example, we give in Figure 2.6 the labelled transition system associated with the process $n[!^1 in\ m.!^2 out\ m] \mid m[\,]$ (we omit in this process unnecessary labels). We use the labelled replications $!^1$ and $!^2$ to distinguish between different replication operators which appear in this process.

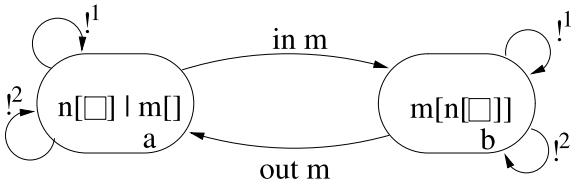


Fig. 2.6 A Labelled Transition System for the Process $n[!^1 in\ m.!^2 out\ m] \mid m[\,]$

We observe that the labelled transitions in $L_{A,B}$ for replications and capabilities of an ambient context correspond to the reductions performed over processes. As shown in steps 3 and 4, the transitions applied for any capabilities or replications associated with the holes are independent of the fact that they are effectively available to perform a transition (at this point).

2.6.3.2 II) From Processes Without Ambients to Petri Nets.

In what follows we show how to build a Petri net from a labelled process without ambients. We denote by $\mathscr{E}(E)$ the set of all multisets which can be built with elements from the set E.

We recall that a Petri net is given by a 5-tuple $(\mathscr{P}, \mathscr{P}_i, \mathscr{T}, Pre, Post)$, where

- $\mathscr{P}$ is a finite set of *places*;
- $\mathscr{P} \subseteq \mathscr{P}_i$ is a set of initial places;
- $\mathscr{T}$ is a finite set of transitions;
- $Pre, Post : \mathscr{T} \to \mathscr{E}(\mathscr{P})$ are mappings from transitions to multisets of places.

We say that an ambient-free process is *rooted* if it is of the form $cap^w n.A'$ or of the form $!^w A'$. We define the Petri net $PN_{A'}$ associated with a rooted process A' as follows: the places of $PN_{A'}$ are precisely the rooted subprocesses of A', and A' itself is the unique initial place; the transitions are defined as the set of all capabilities $in^w n$, $out^{w'} n$ and replications $!^w$ occurring in A'. Finally, *Pre* and *Post* are defined for all transitions as follows:

- $Pre(cap^w n) = \{cap^w n\}$ and $Post(cap^w n) = \emptyset$ if $cap^w n$ is a place in $PN_{A'}$.
- $Pre(cap^w n) = \{cap^w n.(A_1 \mid \ldots \mid A_k)\}$ and $Post(cap^w n) = \{A_1 \mid \ldots \mid A_k\}$ if $cap^w n.(A_1 \mid \ldots \mid A_k)$ is a place in $PN_{A'}$ ($A_1 \mid \ldots \mid A_k$ being rooted processes).
- $Pre(!^w) = \{!^w A'\}$ and $Post(!^w) = \{!^w A', A'\}$ if $!^w A'$ is a place in $PN_{A'}$.

For $!^1 in\ m.!^2 out\ m$, we obtain the Petri net given in Figure 2.7.

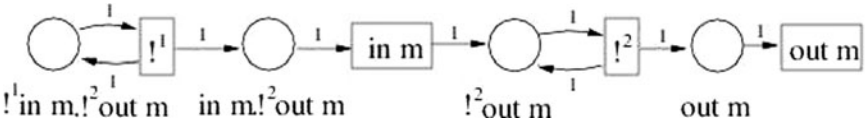


Fig. 2.7 A Petri Net for the Process $!^1 in\ m.!^2 out\ m$

We denote by $PN_{\Box^w}$ the Petri net $PN(\theta_{L_{A,B}}(\Box^w))$, that is, the Petri net corresponding to the rooted ambient-free process associated with $\Box^w$ by $\theta_{L_{A,B}}$. In what follows we show how to combine the transition system $L_{A,B}$ and the Petri nets $PN_{\Box^w}$ into one single Petri net.

2.6.3.3 III) Combining the Transition Systems and Petri Nets.

We first turn the labelled transition system $L_{A,B}$ into a Petri net
$PN_L = (\mathscr{P}_L, \mathscr{P}_L^i, \mathscr{T}_L, Pre_L, Post_L)$, where

- $\mathscr{P}_L$ is a set of states of $L_{A,B}$;
- $\mathscr{P}_L^i$ is a singleton set containing the state corresponding to the ambient context $\mathscr{C}_A$ of A;
- $\mathscr{T}_L$ is the set of transitions of the form (s, l, s'), with
  - s and s' states from $L_{A,B}$,
  - a transition l from s to s' in $L_{A,B}$;

- $Pre(t) = s$ and $Post(t) = \{s'\}$ for all transitions $t = (s, l, s')$.

We define a Petri net $PN_{A,B} = (\mathscr{P}_{A,B}, \mathscr{P}^i_{A,B}, \mathscr{T}_{A,B}, Pre_{A,B}, Post_{A,B})$ by

- places (initial places) of $PN_{A,B}$ are the union of places (initial places) of PN_L and of each of the Petri nets $PN_{\Box^w}$ (for $\Box^w$ occurring in one of the states of $L_{A,B}$);
- transitions of $PN_{A,B}$ are precisely the transitions of PN_L;
- the mappings $Pre_{A,B}$ and $Post_{A,B}$ are defined for all transitions $t = (a, f, b)$ as:

  (i) $Pre_{A,B}(t) = \{a\}$ and $Post_{A,B}(t) = \{b\}$ if f does not occur as a transition in any $PN_{\Box^w}$ (for $\Box^w$ occurring in one of the states of $L_{A,B}$),

  (ii) if f is a transition of $PN_{\Box^w}$, then $Pre_{A,B}(t) = \{a\} \cup Pre_{\Box^w}(f)$ and $Post_{A,B}(t) = \{b\} \cup Post_{\Box^w}(f)$, where $Pre_{\Box^w}$ and $Post_{\Box^w}$ are the mappings Pre and $Post$ of $PN_{\Box^w}$, respectively.

2.6.4 Deciding Reachability

We recall that for a Petri net $PN = (\mathscr{P}, \mathscr{P}^i, \mathscr{T}, Pre, Post)$, a marking m is a multiset from $\mathscr{E}(P)$. A transition t is enabled by a marking m if $Pre(t) \subseteq m$. Executing an enabled transition t for a marking m gives a marking m' defined as $m' = (m \setminus Pre(t)) \cup Post(t)$ (where $\setminus$ stands for the multiset difference). A marking m' is reachable from m if there exists a sequence $m_0, \ldots, m_k$ of markings such that $m_0 = m$, $m_k = m'$ and for each m_i, m_{i+1}, there exists an enabled transition for m_i whose execution gives m_{i+1}.

Theorem 2.22 ([115]). *For all Petri nets P, for all markings m, m' of P, one can decide whether m' is reachable from m.*

For the reachability problem $A \Rightarrow^* B$ over ambients, we consider the Petri net $PN_{A,B}$ and the initial marking m_A defined as $m_A = \mathscr{P}^i_{A,B}$. Figure 2.8 depicts the initial marking for process $n[!^1 in\ m.!^2 out\ m] \mid m[\,]$ as a combination of the labelled transition system of Figure 2.6 and the Petri net of Figure 2.7.

It should be noticed that for any marking m reachable from m_A, m contains exactly one occurrence of a place from $\mathscr{P}_L$. Roughly speaking, to any reachable marking corresponds exactly one ambient context. Moreover, the execution of one transition in the Petri net $PN_{A,B}$ simulates a reduction from $\Rightarrow_{amb}$.

We define now the set $\mathscr{M}_B$ of markings of $PN_{A,B}$ corresponding to B. Intuitively, a marking m belongs to $\mathscr{M}_B$ if m contains exactly one occurrence $\mathscr{C}$ of a place from $\mathscr{P}_L$ (that is, representing some ambient context) and in the context $\mathscr{C}$, the holes can be replaced with processes without ambients to obtain B. Each of the processes without replication must correspond to a marking of the sub-Petri net associated with the hole it fills up. $\mathscr{M}_B$ is defined as the set of markings m for $PN_{A,B}$ satisfying:

(i) there exists exactly one ambient context $\mathscr{C}_m$ in m;

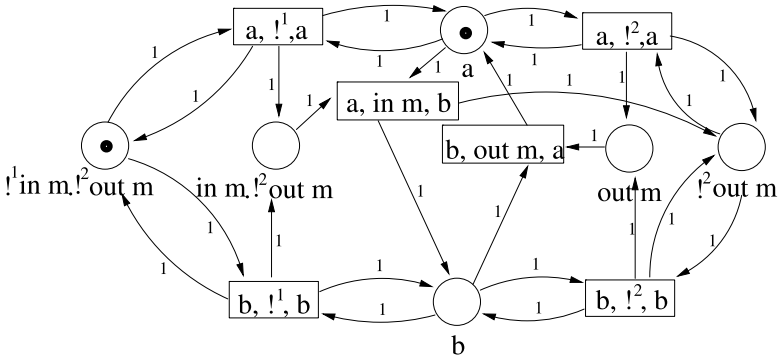


Fig. 2.8 The Petri Net for the Labelled Process $n[!^1 in\ m.!^2 out\ m] \mid m[\,]$

(ii) $\sigma_m(\mathscr{C}_m)$ and B are label-free-equivalent, for any substitution σ_m from holes $\Box^w$ occurring in $\mathscr{C}_m$ to processes without ambients defined as $\sigma_m(\Box_m) = P_1 \mid \ldots \mid P_k$ for $\{P_1, \ldots, P_k\}$ the multiset corresponding to the restriction of m to the places of $PN_{\Box^w}$;

(iii) for all holes $\Box^w$ occurring in a state of the transition system $L_{A,B}$ but not in $\mathscr{C}_m$, the restriction of m to places of $PN_{\Box^w}$ is precisely the set of initial places of $PN_{\Box^w}$.

We adapt the results presented in [28] to our restricted fragment of mobile ambients.

Proposition 2.3. *For a Petri net $PN_{A,B}$, there are only finitely many markings corresponding to a process B, and the set $\mathscr{M}_B$ can be computed.*

The translation correctness is ensured by the following result.

Proposition 2.4. *For all processes A, B we have that $A \Rightarrow_{amb} B$ if and only if there exists a marking from $\mathscr{M}_B$ such that m_B is reachable from m_A in $PN_{A,B}$.*

Using Proposition 2.4 and Theorem 2.22, we can decide whether an ambient A can be reduced to an ambient B.

Theorem 2.23. *For two arbitrary ambients A and B from our restricted fragment, it is decidable whether A reduces to B.*

http://www.springer.com/978-3-642-24866-5

Mobility in Process Calculi and Natural Computing
Aman, B.; Ciobanu, G.
2011, XIV, 210 p., Hardcover
ISBN: 978-3-642-24866-5