General Topologies and P Systems

Erzsébet Csuhaj-Varjú[1], Marian Gheorghe[2,3], and Mike Stannett[2]

[1] Department of Algorithms and Their Applications
Faculty of Informatics, Eötvös Loránd University,
Pázmány Péter st. 1/c, Budapest, 1117, Hungary
csuhaj@inf.elte.hu
[2] Department of Computer Science, The University of Sheffield
Regent Court, 211 Portobello, Sheffield S1 4DP, United Kingdom
{M.Stannett, M.Gheorghe}@dcs.shef.ac.uk
[3] Department of Computer Science, University of Piteşti
Str Targu din Vale, Piteşti, Romania

Summary. In this paper we investigate the use of general topological spaces as control mechanisms for membrane systems. For simplicity, we illustrate our approach by showing how arbitrary topologies can be used to study the behaviour of membrane systems with rewrite and communication rules.

1 Introduction

Membrane computing has emerged in the last more than ten years as a vigorous research field as part of natural computing or unconventional computing. It is a nature-inspired computational paradigm including a large variety of models, called *membrane systems*, well-investigated from a computational perspective, especially with respect to their computational power and complexity aspects [9]. A number of promising applications, mainly in biology, but also in distributed computing, linguistics and graphics [1], have been identified and described.

The key features of a membrane system are a set of *compartments* (called *regions*) delimited by *membranes*, *multisets of objects* contained in these regions, *transformation and communication rules* describing interactions between objects, and a *strategy* for evolving the system. This basic model is inspired by standard models of the structure and functions of a typical eukaryotic cell, comprising multiple compartments containing localised biochemical materials and reactions: various chemical entities with different levels of complexity react under specified circumstances to produce new biochemicals supporting the cell's life and metabolism, and these may or may not be transported to other compartments depending on context. Many variants of membrane system have been considered, some using different types of biochemical agent and interaction, others using various types of structural organisation for the compartments and their connections [9].

Membrane systems introduce in a very natural way a specific topology on the system described, in which membranes delimit compartments containing local objects and interaction rules, together with specific links between compartments. These links describe communication channels allowing adjacent compartments to exchange chemicals. Although this topology is flexible enough to cope with the challenge of modelling various natural or engineering systems, there are cases when a finer grain topological structure is required. In a series of papers, J.-L. Giavitto and his collaborators have investigated the use of topological transformations applied to various data structures, where algebraic topology helps in defining the appropriate data sets selected to be transformed [2]. The use of this approach to model various elements and transformations occurring in membrane computing has been investigated in [4], while concepts related to a spatial computing programming paradigm, which permit the definition and handling of a sort of geometry, have been described in the context of the unconventional programming language, MGS [7].

In this paper we investigate the use of topological spaces as *control mechanisms* for membrane systems. While the algebraic topological approach shows how the membrane structure and its basic operations with multisets can be represented, here we use a topological space as a framework to control the evolution of the system with respect to a family of open sets that is associated with each compartment. This approach produces a fine grain description of local operations occurring in each compartment by restricting the interactions between objects to those from a given neighbourhood. This initial study shows the influence of an arbitrary topology on the way basic membrane systems compute. In future work (cf. Sect. 5) we aim to investigate the role of more specific topologies, their impact on other types of membrane system, and their applications in solving/approaching various problems.

2 Basic notations and definitions

We briefly recall basic notions concerning P systems. For more details on these systems and on P systems in general, we refer to [8, 9]. A basic evolution-communication P system (P system for short) of degree n is a construct

$$\Pi = (O, \mu, w_1, \ldots, w_n, R_1, \ldots, R_n, i_0)$$

where

1. O is a finite alphabet of symbols called objects;
2. μ is a membrane structure consisting of n membranes that are labelled (in a one-one manner) with elements from a given alphabet A; these membranes are organised in a hierarchical way, like a tree, with the top membrane (root) called the *skin membrane*, and the bottom ones (leaves) called *elementary membranes*;

3. for each $1 \leq i \leq n$, $w_i \in O^*$ is a multiset of objects associated with the region i (this is the region delimited by membrane i, but not including the subregions delimited by i's children);
4. for each $1 \leq i \leq n$, R_i is a finite set of rules associated with the region i, of the form $u \to (v_1, tar_1) \ldots (v_m, tar_m)$, where $u \in O^+$, $v_j \in O$ and $tar_j \in \{in, out, here\}$ $(1 \leq j \leq m)$; when tar_j is *here*, we write simply v_j in place of (v_j, tar_j);
5. i_0 is the label of an elementary membrane of μ that identifies the corresponding output region.

A P system is interpreted dynamically as a computational device comprising a set of n hierarchically nested membranes that identify n distinct regions (the membrane structure μ), where each region $i = 1, \ldots, n$ contains a multiset of *objects* (w_i) and a finite set of *evolution rules* (R_i) of the form $u \to (v_1, tar_1) \ldots (v_m, tar_m)$. This rule removes multiset u from region i, and then adds each multiset v_j $(1 \leq j \leq m)$ to the multiset of objects in the corresponding target region tar_j.

- If tar_j does not appear in the notation (by convention this occurs when the target is *here*), then v_j remains in membrane i;
- If tar_j is *out*, then v_j is sent to the parent membrane of i; if i is the skin membrane then v_j is sent out of the system;
- If tar_j is *in*, then v_j is sent to one of the inner membranes of i (if there is more than one child, the target is chosen non-deterministically);
- The *in* target can be replaced by a precisely defined destination region. If region k is a child of i and tar_j is k, then v_j is sent to k.

A computation of the system is obtained by applying the available rewrite rules in a non-deterministic maximally parallel manner[4], where each region i initially contains the corresponding finite multiset w_i.

A computation is considered successful when it starts from the initial configuration and reaches a configuration where no further rules can be applied. Its result, a natural number, is obtained by counting the objects present in region i_0 on completion (other ways of interpreting the result of a P system computation are also considered in the literature [9]). Given the non-deterministic nature of P system computation, different runs of a given system may generate different results. For a given P system Π the set of numbers that can be computed is denoted $N(\Pi)$.

Recall that rewrite rules are of the form $u \to (v_1, tar_1) \ldots (v_m, tar_m)$, where u is a multiset. If in each of the rules in Π the multiset u contains only a single object, then Π is called a *P system with non-cooperative rules*; otherwise it is a *P system with cooperative rules*. When $tar_j = in$ the rule is said to have *arbitrary target*, and when $tar_j = in_k$ for a specific region k, it has a *selected* target.

[4] A simultaneous application of rewrite rules is *non-deterministic maximally parallel* provided the applied rules are chosen non-deterministically (possibly with repetition), and there are insufficient resources to trigger the simultaneous application of any additional rule.

2.1 Topological conventions

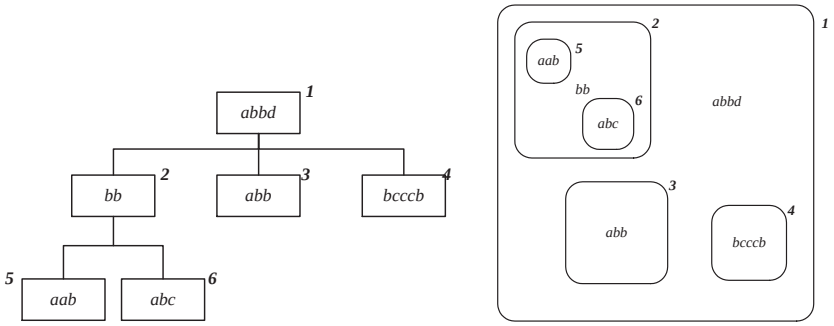
Our notation will generally follow that of [10]. Given any non-empty set X, its power set will be denoted $\wp X$. We write $\varnothing$ for the empty set. A *topology* on X is any subset of $\wp X$ containing both $\varnothing$ and X, which is closed under arbitrary unions and finite intersections; the members of $\mathcal{T}$ are *open* (or $\mathcal{T}$*-open* where ambiguity might otherwise arise). The topology $\{\varnothing, X\}$ is the *indiscrete* topology on X; the topology in which every singleton $\{x\} \in \wp X$ is open is the *discrete* topology. An *open cover* of $A \subseteq X$ is a subset of $\mathcal{T}$ whose union contains A.

The complement of an open set is *closed*. The *closure* $\overline{A} = \mathrm{Cls}_X(A)$ of a set $A \subseteq X$ is the intersection of all closed sets containing A; it is the smallest such set. The *interior* $A^\circ = \mathrm{Int}_X(A)$ of A is the union of all open sets contained in A; it is the largest such set. The difference between the closure and interior of a set is its *boundary*, $\partial A = \overline{A} \setminus A^\circ$.

Any topology $\mathcal{T}$ can also be regarded as a partially ordered set (*poset*) ordered by set inclusion. If $(Y, \leq)$ is a poset, an *order embedding* of Y in $\mathcal{T}$ is an injection $\imath\colon Y \to \mathcal{T}$ such that $y_1 \leq y_2$ if and only if $\imath(y_1) \leq \imath(y_2)$.

3 Control structures

For the purposes of this paper, a P system can be regarded structurally as a tree whose nodes are the *membranes*, together with a function mapping each node p in Π to a corresponding multiset over A. This multiset tells us how many copies of each object lie in the region situated between the membrane and its internal sub-membranes; see Fig. 1.



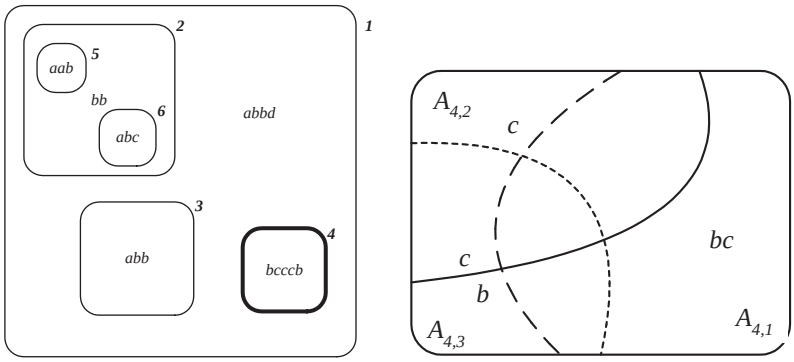
(a) tree (b) nested membranes

Fig. 1. A generic P system structure represented as (a) a tree; (b) a set of nested membranes.

In each membrane and in any computation step it is assumed that all the objects present in the corresponding region can freely interact according to the

set of rules available in that region. Maximal parallelism also implies that all the objects that might take part in various interactions must interact (each object takes part in at most one interaction). While this scheme is easy to implement, it distorts to some extent the biological intuition that interactions are *local*. It is not enough that two chemicals are present in a cell, they must also be located close to one another, but the regions of a P system are not inherently associated with any notion of separation distance. We will therefore order-embed the membranes of the P system as open sets within an essentially arbitrary topology, and use (finite) open covers to provide an indication of the distance between two objects. We then consider how the choice of topology affects the computations that can be implemented.

In general the members of an open cover need not be disjoint. Region 4 of Fig. 1 contains the multiset *bcccb*. Figure 2 illustrates a covering of this region by three open sets: $A_{4,1}$, $A_{4,2}$ and $A_{4,3}$. The open set $A_{4,2}$ contains *cc*, and each of the others contains *bc*. Initially we only consider open covers of *regions*; subregions of the enclosing membrane will be equipped with covers in their own right.

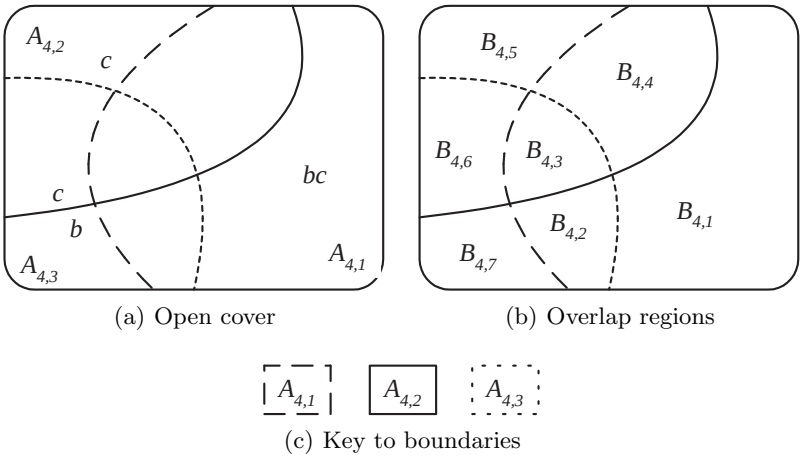


(a) Nested membranes (region 4 highlighted) (b) Open covering of region 4

Fig. 2. Covering of region 4 by open sets.

The *topologically controlled computation* that takes place with respect to these open sets is defined as follows: *rules associated with membrane i are enabled if and only if there is a member of the open cover which contains all of the participating objects*. If any target of an enabled rule is *here*, the associated products should then be placed back into the same open set (if the initial objects lie in more than one member of the cover, we choose one at random and place the associated results there; they need not be injected back into the intersection). Otherwise if the target is *tar* (where *tar* is assumed to carry its own open cover), the output will be placed in an arbitrary member of *tar*'s cover.

Despite the intrinsically local nature of controlled computation, the locus of computation can migrate from one compartment to another one via non-empty overlap regions, as the following example illustrates. Figure 3 shows the disjoint parts, $B_{4,1}$ – $B_{4,7}$, of region 4's cover. These are all of the form $U \setminus V$ where U and V are open; for example $B_{4,3} = (A_{4,1} \cap A_{4,2} \cap A_{4,3}) \setminus \varnothing$, and $B_{4,1} = A_{4,1} \setminus (A_{4,2} \cup A_{4,3})$.



(a) Open cover (b) Overlap regions

(c) Key to boundaries

Fig. 3. The finite covering of region 4 and its disjoint overlap regions.

Suppose, then, that region 4 has the following rules associated with it:

$$r_1 : bc \to b; \quad r_2 : bcc \to c; \quad r_3 : cc \to c.$$

If we consider the system as a P system with no topological control in place, the following computations can take place:

1. $bc\,c\,cb \overset{r_1,r_1}{\Longrightarrow} bcb \overset{r_1}{\Longrightarrow} bb$
2. $bc\,cc\,b \overset{r_1,r_3}{\Longrightarrow} bcb \overset{r_1}{\Longrightarrow} bb$
3. $bc\,ccb \overset{r_1,r_2}{\Longrightarrow} bc \overset{r_1}{\Longrightarrow} b$

But when the open sets are in place computation path 3 is blocked, because none of the open sets ever contains bcc, whence r_2 cannot be triggered. We have the following two cases instead:

1' r_1 is applied in both $A_{4,1}$ and $A_{4,3}$ resulting in a copy of b in each of these open sets; if $b \in A_{4,3}$ is not in $A_{4,2} \cap A_{4,3}$ the computation stops here with bbc scattered across different open sets. If, on the other hand, $b \in A_{4,2} \cap A_{4,3}$ then the computation can continue; after applying r_1 in $A_{4,2}$ a copy of b is obtained in each of $A_{4,1}$ and $A_{4,2}$. In this case the result is the same as that obtained in (1);

2'. r_3, r_1 result in copies of $b \in A_{4,1}$ and $c \in A_{4,2}$; as in (1') this c can reside either in the intersection or outside it; in the first case r_1 can be applied again and b is computed (so that the result from (2) is obtained). Otherwise *bbc* will remain in the membrane unchanged.

Consider in particular the second case of the first step of (1'). After r_1 is applied in $A_{4,3}$ the result b can be considered to lie in $A_{4,2}$, whence (as suggested above) the locus of computation can migrate from $A_{4,3}$ to $A_{4,2}$ via their intersection. A similar situation occurs in (2') as well.

More generally, suppose that region i has a rule whose target is region j. We will allow the rule to be triggered only when the two regions are sufficiently close to one another (their boundaries must intersect: $\partial i \cap \partial j \neq \varnothing$). In this case, and provided all of the required components are available within a single member of i's cover, the rule can fire with the resulting multiset v_j being injected into an arbitrarily selected member of j's cover. In future work we plan to investigate what happens when this restriction is weakened, so that interactions can occur between non-neighbouring regions.

Definition 1 (Output of a controlled computation). *For a P system Π and associated topology $\mathcal{T}$ the set of numbers computed by Π when controlled by $\mathcal{T}$ will be denoted $N_{\mathcal{T}}(\Pi)$.* □

Having now defined controlled computation, we will address the following problems. In Sect. 4 we discuss the role of a control mechanism based on an associated topology and show how a general topology influences the computation for a basic class of P systems. In Sect. 5 we summarise our findings and discuss future research topics related to various topologies associated with classes of P systems.

4 Basic Results

We will first consider P systems with non-cooperative rules.

Lemma 1. *For any P system with non-cooperative rules and either arbitrary targets or selected targets, Π, and any associated topology $\mathcal{T}$, $N(\Pi) = N_{\mathcal{T}}(\Pi)$.*

Proof. In a P system with non-cooperative rules the left hand side of any rule has only one single object, hence no interactions are involved. In this case it is obvious the the topology $\mathcal{T}$ does not influence the computation for either P systems with arbitrary targets or selected targets, hence the result stated holds. □

For P systems with cooperative rules the situation is totally different and the topologies associated with them may lead to different computations and distinct results.

Lemma 2. *There is a P system with cooperative rules and either arbitrary or selected targets, Π, such that for any associated topology $\mathcal{T}$ where for at least one region not all the objects belong to the same open set, it follows that $N(\Pi) \neq N_{\mathcal{T}}(\Pi)$.*

Proof. Let us consider $\Pi = (O, \mu, w_1, w_2, R_1, R_2, i_0)$, where $O = \{a, b, c\}$, $\mu = [[]_2]_1$, $w_1 = ab$, $w_2 = \lambda$, $R_1 = \{ab \to c, c \to (c, in)\}$, $R_2 = \varnothing$, $i_0 = 2$. This system uses an arbitrary target, which, in this case, is the same as selected target, in_2. This P system computes c in two steps in the output region, 2. Any topology, $\mathcal{T}$, associated with Π that provides a cover for region 1 with more than an open set, must have an open set for a and another one for b and their intersection does not contain any of these two objects; otherwise, a and b will stay in the same open set. In this case the rule $ab \to c$ can not be applied and consequently c is never obtained in the output region, hence $N(\Pi) \neq N_{\mathcal{T}}(\Pi)$. □

Theorem 1. *For any P system with either arbitrary or selected targets, the computation and the topologically controlled computation are the same when non-cooperative rules are used and are not in general the same for cooperative rules.*

Proof. The proof is an immediate consequence of Lemmas 1 and 2. □

There are P systems with cooperative rules where the content of the regions can be matched against the open sets in such a way that the computation is equivalent to the computation of the original system. Indeed let us consider the problem of checking that a positive integer m is divided by another positive integer k. We propose a P system below which is an adaptation of the P system presented in [9].

Example 1. Let us consider $\Pi = (O, \mu, w_1, w_2, R_1, R_2, i_0)$, where $O = \{a, b, c, y, n\}$, $\mu = [[]_2]_1$, $w_1 = a^m b^k$, $w_2 = y$, $R_1 = \{r_1 : ab \to c, r_2 : ac \to b, r_3 : bc \to (n, in)\}$, $R_2 = \{yn \to n\}$, $i_0 = 2$.

In the first step at most k objects ab are replaced by the same number of objects c (using r_1 at most k times) and then objects ac are replaced by objects b (using r_2). If k divides m then the process will stop after h steps, where $m = kh$, and in membrane 2 will remain y; otherwise in membrane 1 the process of alternatively applying rules r_1 and r_2 will stop with some objects b and objects c and the rule r_3 can be used. In this case n is sent into region 2 and finally n is obtained in this region.

Now, if we aim to obtain the same results in region 2, i.e., y, when k divides m, or n otherwise, then we have to build the topology, $\mathcal{T}$, associated with Π in a certain way which is subsequently described. Region 2 is covered by only one single open set and region 1 will have an arbitrary number of open sets, $q > 1$, associated with. *Any two such open sets are disjoint.* The objects will be distributed as follows: the k b's will be randomly distributed in $q - 1$ of the q open sets, $b^{k_1}, \ldots, b^{k_{q-1}}$, $k_i \geq 0$ and $k_1 + \cdots + k_{q-1} = k$. If $m = kh + r$, then in each of the $q - 1$ open sets containing k_i b's, the number of a's is hk_i a's. If $r > 0$ then one more a will be consider in one of the $q - 1$ open sets with b's and the rest will be associated with the q^{th} open set. Clearly, in each of the $q - 1$ open sets the computation will go for h steps. In $q - 2$ of them it will be obtained either only b's or only c's; the open set with an additional a in it will end up after one more step with a mixture of b's and c's and the rule r_3 will push an n into membrane 2 and finally will get n in this membrane. Objects a's occurring in the q^{th} open set will remain there forever. It follows that $N(\Pi) = N_{\mathcal{T}}(\Pi)$. □

The question of whether the control structure introduced by a topology can be ignored, perhaps by using a more complex P system, is answered by the following result. This takes into account the interpretation of the outcome of the computation as being the number of objects, given by the size of the multiset, present in the output region.

Theorem 2. *For any P system, Π, and any associated topology, $\mathcal{T}$, there is a P system, Π', of the same degree with Π, such that $N_{\mathcal{T}}(\Pi) = N(\Pi')$.*

Proof. The idea of the proof is to construct a new P system such that objects belonging to a region adequately refer to objects of the open sets in the corresponding regions of the initial P system.

Let Π be a P system of degree n, $\Pi = (O, \mu, w_1, \ldots, w_n, R_1, \ldots, R_n, i_0)$, and $\mathcal{T}$ a topology associated with it. In order to build a new P system, Π', of degree n, a few preliminary notations are made. First, please observe that for each region i, $1 \leq i \leq n$, there exists a family of open sets $A_{i,1}, \ldots, A_{i,k_i}$ covering it. In general these open sets are not disjoint and we describe the finest disjoint parts of the cover by considering either some intersections of open sets or the complement of an open set with respect to the rest of the open sets; it follows that there exists a finite set, denoted B_i, containing the sets $B_{i,1}, \ldots, B_{i,m_i}$, such that $B_{i,j}$ denotes either $A_{i,l_1} \cap \cdots \cap A_{i,l_j}$, $1 \leq l_j \leq k_i$ or $A_{i,j} \setminus (A_{i,1} \cup \cdots \cup A_{i,j-1} \cup A_{i,j+1} \cup \cdots \cup A_{i,k_i})$. The set of indexes of the above sets $B_{i,j}$ is denoted by C_i, i.e., $C_i = \{(i,j) \mid B_{i,j} \in B_i\}$. Each object, $a \in O$, of the multiset from region i belongs to a certain $B_{i,j}$. For each a from $B_{i,j}$, the following objects are considered, $a^{\alpha}, \alpha \in C_i$.

The P system Π', of degree n, is built as follows:

$$\Pi' = (O', \mu, w'_1, \ldots, w'_n, R'_1, \ldots, R'_n i_0) \ ,$$

where:

1. $O' = \{a^{\alpha} | a \in O, \alpha \in C_i, 1 \leq i \leq n\}$;
2. μ is the membrane structure of Π;
3. $w'_i = a_{i,1}^{(i,r_1)} \ldots a_{i,p_i}^{(i,r_{p_i})}$, where $a_{i,j} \in B_{i,r_j}$, $1 \leq j \leq p_i$, for $w_i = a_{i,1} \ldots a_{i,p_i}$, initial multiset of Π;
4. for each rule $a_{i,1} \ldots a_{i,q_i} \rightarrow b_{i,1} \ldots b_{i,s_i} \in R_i$, R'_i contains $a_{i,1}^{(i,r_1)} \ldots a_{i,q_i}^{(i,r_{q_i})} \rightarrow b_{i,1}^{(i,s_1)} \ldots b_{i,p_i}^{(i,s_{p_i})}$, $(i,r_j) \in C_i$, $1 \leq j \leq q_i$, $(i,s_j) \in C_i$, $1 \leq j \leq p_i$ when a target, t, appears on the right hand side of the rule from R_i, associated with an object $b_{i,j}$, then the target will point to any of the open sets $A_{t,j}$ of the target region t;
5. Π and Π' have the same output membrane, i_0.

The codification provided by Π' allocates, in a unique way, in every region, i, each object, a, to a specific open set, by "stamping" it with the corresponding index, $(i,j) \in C_i$, of the set $B_{i,j}$. Whenever a rule is applied, the resulted multiset is also composed of objects uniquely associated with certain open sets, either from the current region or from the target ones.

More precisely, when in region i of Π the current multiset is

$$u = a_1 \dots a_{q_1} a_{q_1+1} \dots a_{q_2} \dots a_{q_{e-1}} a_{q_{e-1}+1} \dots a_{q_e} z$$

and there are rules $\rho_1, \dots, \rho_e \in R'_i$, where $\rho_j : a_{q_{j-1}+1} \dots a_{q_j} \to b_{p_{j-1}+1} \dots b_{p_j}$, $q_0 = 0$, then $\rho_1, \dots, \rho_e$ are applied in a computation step, according to maximal parallelism semantics, to u with respect to topology $\mathcal{T}$.

If $u \Rightarrow_{\rho_1,\dots\rho_e} v$, with $v = b_1 \dots b_{p_1} \dots b_{p_{e-1}} \dots b_{p_e} z$ then each a_h, $q_{j-1} + 1 \leq h \leq q_j$, belongs to a certain B_{i,r_h} included in the same open set $A_{i,j}$ where ρ_j is applied. Each of the objects b_h, $p_{j-1} + 1 \leq h \leq p_j$, belongs to some B_{i,s_h} included in the same $A_{i,j}$ set.

In the P system Π', in region i, there is

$$u' = a_1^{(i,r_1)} \dots a_{q_1}^{(i,r_{q_1})} a_{q_1+1}^{(i,r_{q_1+1})} \dots a_{q_2}^{(i,r_{q_2})} \dots a_{q_{e-1}}^{(i,r_{q_{e-1}})} a_{q_{e-1}+1}^{(i,r_{q_{e-1}+1})} \dots a_{q_e}^{(i,r_{q_e})} z',$$

where $(i,j) \in C_i$, $j \in \{r_1, \dots, r_{q_e}\}$. The multiset z' consists of objects $(a^\alpha)^c$ for a^c occurring in z and $\alpha \in C_i$. There are rules $\rho'_1, \dots, \rho'_e$, where
$\rho'_j : a_{q_{j-1}+1}^{(i,r_{q_{j-1}+1})} \dots a_{q_j}^{(i,r_{q_j})} \to b_{p_{j-1}+1}^{(i,s_{p_{j-1}+1})} \dots b_{p_j}^{(i,s_{p_j})}$, which are applied in a maximal parallel manner to u'. If $u' \Rightarrow_{\rho'_1,\dots\rho'_e} v'$, then

$$v' = b_1^{(i,s_1)} \dots b_{p_1}^{(i,s_{p_1})} \dots b_{p_{e-1}}^{(i,s_{p_{e-1}})} \dots b_{p_e}^{(i,s_{p_e})} z',$$

where $(i,j) \in C_i$, $j \in \{s_1, \dots, s_{p_e}\}$.

The above construction proves that $u \Rightarrow_{\rho_1,\dots\rho_e} v$ in Π if and only if $u' \Rightarrow_{\rho'_1,\dots\rho'_e} v'$ in Π'. This shows that the same number of symbols are engaged in any computation step in Π and Π', hence these P systems compute the same number of symbols in i_0. □

From the above proof it is clear that the numbers of objects and rules used by the P system Π' are both significant compared to those of Π. The next result provides lower and upper bound limits for these two parameters. We need a few more notations to describe the result.

For a finite set X, let us denote by $card(X)$, the number of elements of X. With respect to the proof of Theorem 2, the following notations are introduced: K is the number of elements of the set O, n is the degree of the two P systems, Π and Π'; given that for each region i, $1 \leq i \leq n$, the number of sets $B_{i,j}$ is m_i, let us denote $m = min\{m_i \mid 1 \leq i \leq n\}$, $M = max\{m_i \mid 1 \leq i \leq n\}$, $p = min\{|x|, |y| \mid \text{all } x \to y \in R_i, 1 \leq i \leq n\}$ and $P = max\{|x|, |y| \mid \text{all } x \to y \in R_i, 1 \leq i \leq n\}$; if g_i is the maximum number of neighbours that appear in the rules of R_i, then $g = min\{g_i \mid 1 \leq i \leq n\}$; finally we have $Q = card(R_1 \cup \dots \cup R_n)$. With these notations we can formulate the following result.

Corollary 1. *For any P system Π and any associated topology $\mathcal{T}$, define Π' and the associated notation as above. Then*

(i) $Kmn \leq card(O') \leq KMn$;

(ii) $Qm^p(min\{m,g\})^p \leq card(R'_1 \cup \cdots \cup R'_n) \leq QM^P(M+n-1)^P$.

Proof. Part (i) follows from the fact that for each object $a \in O$, distinct instances are created for each of the n membranes and in every region i $(1 \leq i \leq n)$, and each set $B_{i,j}$ $(1 \leq j \leq m_i)$. Hence, $card(O')$ is bounded between Kmn and KMn.

To prove (ii), we observe that for each rule $x \to y \in R_i$, the following rules are added to R'_i, $x^\alpha \to y^\beta$, $\alpha \in C_i$, $\beta \in C_i \cup C_{j_1} \cup \cdots \cup C_{j_i}$, where $j_1, \ldots, j_i$ are neighbours of i where objects of y^β can go to. The left hand side, x^α, will have elements from any of the m_i sets, $B_{i,j}$, hence the lower and upper bounds are m^p and M^P, respectively. Each of the right hand side elements of y^β should belong to either one of the $B_{i,j}$ sets or to one of the neighbours of i, maximum $n-1$, so the lower and upper bounds are $(min\{m,g\})^p$ and $(M+n-1)^P$, respectively. We can then get the two boundaries of $card(R'_1 \cup \cdots \cup R'_n)$. □

5 Summary and Conclusions

In this paper we have investigated the use of general topological spaces to control local interactions in basic membrane systems. This approach produces a fine grain description of local operations occurring in each compartment by restricting the interactions between objects to those from a certain vicinity. In our future work we aim to investigate the role of more specific topologies, their impact on other types of membrane systems and their applications to various problems. In particular:

1. By construction, P systems have a tree-like nested membrane structure. Given the topological embeddings used in this paper, it is no longer clear whether this structure is relevant; the same proofs appear to work for different underlying graph structures with some adjustments.
2. It would be interesting to study the robustness of P systems with respect to different topologies. How much we can change the topology while still obtaining the same or almost the same computed set of numbers? To what extent can locality be refined starting from a given topology and changing it?
3. If we restrict attention to classes of control space (Tychonov spaces, compact Hausdorff spaces, metric spaces, etc) for which a wide range of topological results are available, can these results be applied to produce associated characterisations of controlled computability?

Acknowledgements. MPS and MG were partially supported under the Royal Society International Exchanges Scheme (ref. IE110369); MG was also partially supported by the project MuVet, Romanian National Authority for Scientific Research (CNCS – UEFISCDI) grant number PN-II-ID-PCE-2011-3-0688.

References

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

2. J.-L. Giavitto: Topological collections, transformations and their applications to the modeling and simulation of dynamical systems. *The 14-th International Conference on Rewriting Techniques and Applications*, LNCS 2706, Springer, 2003, 208–233.
3. J.-L. Giavitto, O. Michel: MGS: A rule-based programming language for complex objects and collections. *Electronic Notes in Theoretical Computer Science*, 59 (2001), 286–304.
4. J.-L. Giavitto, O. Michel: The topological structures of membrane computing. *Fundamenta Informaticae*, 49 (2002), 123–145.
5. J.-L. Giavitto, H. Klaudel, F. Pommereau: Qualitative modelling and analysis of regulations in multi-cellular systems using Petri nets and topological collections. Proceedings Fourth Workshop on Membrane Computing and Biologically Inspired Process Calculi (MeCBIC) 2010 (G. Ciobanu, M. Koutny eds.), EPTCS, 40 (2010).
6. J.-L. Giavitto, H. Klaudel, F. Pommereau: The topological ttructures of membrane computing. *Theoretical Computer Science*, submitted.
7. J.-L. Giavitto, A. Spicher: Topological rewriting and the geometrization of programming. *Physica D*, 237 (2008), 1302–1314.
8. Gh. Păun: Computing with membranes. *Journal of Computer and System Science*, 61 (2000), 108–143.
9. Gh. Păun, G. Rozenberg, A. Salomaa, eds.: *The Oxford Handbook of Membrane Computing*. Oxford Univ. Press, 2010.
10. S. Willard: *General Topology*. Dover Publications Inc., Mineola, NY, 2004.