A NEW WAY TO OBTAIN HOMOLOGY GROUPS IN BINARY 2D IMAGES USING MEMBRANE COMPUTING

DANIEL DÍAZ-PERNIL[1], MIGUEL A. GUTIÉRREZ-NARANJO[2], PEDRO REAL[1], AND VANESA SÁNCHEZ-CANALES[1]

Abstract. Membrane Computing is a computational model inspired in the structure and function of living cells and tissues. In this paper we use Membrane Computing techniques to solve the *Homology Groups of Binary 2D Image (HGB2I) Problem.* This is a classical problem in *Homology Theory* which tries to calculate the number of connected components and the representative curves of the holes of these components of a given binary 2D image. To this aim, we use a family of P systems which solves all the instances of the problem in the framework of *Tissue-like P systems with catalysts.*

Introduction

Natural Computing studies new computational paradigms inspired from Nature. It abstracts the way in which Nature *acts*, conceiving new computing models from natural phenomena in physics, chemistry and biology[1]. The field is growing rapidly and currently there are many open research lines based on Nature, offering solutions to many classical computational problems from a new perspective. From this new paradigm, we can cite areas such as *Cellular Automata* [29], *Genetic algorithms* [15], *Neural Networks* [21], *DNA-based* molecular computing [1], *Swarm Intelligence* [11] or *Membrane Computing* [24, 25] among others. All these computational paradigms have in common the use of an alternative way of encoding the information, adapted to the bio-inspired substrate and the use of intrinsic parallelism of natural processes.

In this paper we use techniques from *Membrane Computing*[2]. This new research area was born from the assumption that the processes taking place within the compartmental structure of a living cell can be interpreted as computations [25]. In particular, it focuses on membranes, which are involved in many reactions taking place inside various compartments of a cell. Biological membranes are much more than mere barriers that define compartments. They act as selective channels of communication between different compartments as well as between the cell and its environment [2].

The computational devices in Membrane Computing are called *P systems*. Roughly speaking, a P system consists of a membrane structure, in the compartments of which one places multisets of objects which evolve according to given rules. In the most extended model, the rules are applied in a synchronous non-deterministic maximally parallel manner, but some other semantics are being explored.

Membranes divide the Euclidian space into regions, which contains objects. Objects evolve according to local reaction rules. Such objects can be described by symbols or by

[1]An introduction on Natural Computing can be found in [16].

[2]We refer to [25] for basic information in this area, to [27] for a comprehensive presentation and the web site [30] for the up-to-date information.

strings of symbols from a given alphabet. These objects can also pass through membranes, under the control of specific rules.

We will focus here on a P system model called (because of their membrane structure) *Tissue-like P systems*. This P system model has two biological inspirations (see [20]): intercellular communication and cooperation between neurons. The common mathematical model of these two mechanisms is a net of processors dealing with symbols and communicating these symbols along channels specified in advance. The communication among cells is based on symport/antiport rules[3]. Symport rules move objects across a membrane together in one direction, whereas antiport rules move objects across a membrane in opposite directions.

Homology theory is a branch of algebraic topology that attempts to distinguish between spaces by constructing algebraic invariants that reflect the connectivity properties of the space. The field has it origins in the work of Poincaré. Homology groups (related to the *different* n-dimensional holes, connected components, tunnels, cavities, etc.) are invariants from Algebraic Topology which are frequently used in Digital Image Analysis and Structural Pattern Recognition. In some sense, they reflects the topological nature of the object in terms of the number and characteristics of its holes.

This is not the first time in which life-based methods are applied to Algebraic Topology. In 1996, J. Chao and J. Nakayama connected Natural Computing and Algebraic Topology using Neural Networks [7] (extended Kohonen mapping). Some years after, K.G. Subramanian *et al.* presented in [6, 5] two works where Digital Image and Natural Computing were linked.

In 2009, Christinal *et al.* presented in [9] a new way to obtain the Betti numbers of a binary 2D and 3D digital image. In this paper, we present a new technique to get not only the Betti numbers, we obtain the representative curves of the homology group H_1 of a binary 2D digital image too.

The necessary time to calculate the homology groups of 2D digital images with basic P systems defined in Section 2 is logarithmic with respect to the input data ($\mathcal{O}(n)$). This involves an improvement with respect to the algorithms development by S. Peltier *et al.* in [28], where they use irregular graphs pyramids with a time complexity of $\mathcal{O}(n^{5/3})$.

The paper is organized as follows: firstly, we present our bio-inspired formal framework. Next, we briefly recall the *Homology Groups of Binary 2D Image (HGB2I) Problem*. In the next Section, we present our solution in the framework of tissue-like P systems with catalysts. Finally, some conclusions are presented.

1. Formal Framework

In tissue-like P systems the membrane structure is a general undirected graph. The edges of such graph are not given explicitly, but they are deduced from the set of rules. From the seminal definition of tissue P systems, several research lines have been developed and other variants have arisen (see, for example, [3, 4, 14, 18, 26]).

In this paper, we endow tissue-like P systems with catalysts. Catalytic P systems were introduced in [24]. The main feature of these P systems is the presence of objects in membranes such that they are not consumed by the application of the rule, but their presence

[3]This way of communication for P systems was introduced in [22].

in the membrane is necessary for the triggering (see, e.g., [12, 13, 19, 17]. Next we provide the definition of tissue-like P systems with catalysts:

Definition 1.1. A *tissue-like P system with catalysts* of degree $q \geq 1$ is a tuple of the form

$$\Pi = (\Gamma, \mathcal{E}, w_1, \ldots, w_q, \mathcal{R}, i_0),$$

where:

(1) Γ is a finite alphabet, whose symbols will be called objects.
(2) $\mathcal{E} \subseteq \Gamma$ is a finite alphabet representing the set of the objects in the environment available in an arbitrary large amount of copies.
(3) $w_1, \ldots, w_q$ are strings over Γ representing the multisets of objects associated with the cells in the initial configuration.
(4) $\mathcal{R}$ is a finite set of *catalytic rules* of the following form: $(cat \,|\, i, u/v, j)$ for $i, j \in \{0, 1, 2, \ldots, q\}, i \neq j$ and $cat, u, v \in \Gamma^*$. The length of a catalytic rule is defined as $|u| + |v|$. The catalyst cat is modified by the application of the rules and cat and v can be empty.
(5) $i_0 \in \{0, 1, 2, \ldots, q\}$ denotes the output region, which can be the environment ($i_0 = 0$) or the region inside a cell ($1 \leq i_0 \leq q$).

Informally, a tissue-like P system with catalysts of degree $q \geq 1$ can be seen as a set of q cells (each one consisting of a single membrane) labeled by $1, 2, \ldots, q$. The cells are the nodes of a virtual graph, where the edges connecting the cells are determined by the communication rules of the system, i.e., as usual in tissue-like P systems, the edges linking calls are not provided explicitly: If a rule $(cat \,|\, i, u/v, j)$ is given, then cells i and j are considered linked.

The application of a catalytic rule $(cat \,|\, i, u/v, j)$ consists on the trade of the multiset u (initially in the cell i) against the multiset v (initially in j). The trade can also be between one cell and the environment, labelled by 0. The rule is applied if in the cell with label i the objects of the set cat are present (catalyst). If the catalyst is empty, then the rule is called a *communication rule*.

In our definition, all objects in the alphabet can act as catalysts, depending on the applied rule. Rules are used as usual in the framework of membrane computing, that is, in a maximally parallel way (a universal clock is considered). In one step, each object in a membrane can only be used for one rule (non-deterministically chosen when there are several possibilities), but any object which can participate in a rule of any form must do it, i.e., in each step we apply a maximal multiset of rules.

A *configuration* is an instantaneous description of the P system and it is represented as a tuple $(w_1, \ldots, w_q)$. Given a configuration, we can perform a computation step and obtain a new configuration by applying the rules in a parallel manner as it is shown above. A sequence of computation steps is called a *computation*. A configuration is *halting* when no rules can be applied to it. The output of a computation is collected from its halting configuration by reading the objects contained in the output cell.

Example 1.2. Let us consider the following *tissue-like P system with catalyst* of degree 3, $\Pi = (\Gamma, \mathcal{E}, w_1, w_2, w_3, \mathcal{R}, i_0)$ where $\Gamma = \{a, b, c, d, e\}$, $\mathcal{E} = \{d\}$, $w_1 = a$, $w_2 = bd$ and $w_3 = ce$.

The set R has four rules:

$$R_1 \equiv (c\,|\,2, b/a, 1) \quad R_2 \equiv (e\,|\,3, c/d, 2) \quad R_3 \equiv (3, d/a, 0) \quad R_4 \equiv (3, d/b^2, 0)$$

The output region is $i_0 = 3$. According to the set of rules, cell 2 is linked to cells 1 and 3 and cell 3 is linked to the environment. The computation starts from the initial configuration $C_0 = (w_0, w_1, w_2)$. Rule R_1 cannot be applied in this initial configuration, since the catalyst c does not appear in cell 2. Rules R_3 and R_4 cannot be applied, since object d is not placed on cell 3. In this initial configuration we only can apply rule 2, since objects e and c are placed in membrane 3 and d is placed in membrane 2. After applying this rule we obtain a new configuration. $C_1 = (w_0', w_1', w_2')$ with $w_1' = a$, $w_2' = cb$ and $w_3' = de$. Now, rule 2 can be applied by interchanging objects a and b, because the catalyst c is placed on cell 2. The communication rules 3 and 4 can be applied, since d appears in cell 3 and the environment contains objects a and b, but only one of them can be applied, and it is non-deterministically chosen. In we choose rule 3, then the new configuration is $C_2 = (w_0'', w_1'', w_2'')$ with $w_1'' = b$, $w_2'' = ac$ and $w_3'' = ae$. No more rules can be applied and the computation finishes. The output of the computation is the multiset in cell 3 in the halting configuration $w_3'' = ae$. If we choose rule 4, the obtained configuration is the same, but $w_2'' = b^2e$.

2. Calculating Homology Groups

In a binary 2D image, the computation of homology groups can be reduced to a process of black and white connected components labeling. The different black connected components are the generators of the 0-dimensional homology group of the *black* part of the image whereas the closed *black* curves surrounding the different white connected components of the image are the generators of its 1-dimensional homology group.

In order to formalize this problem we will consider a ordered[4] set of n^2 pixels $P = \{(i,j) : 1 \leq i,j \leq n\}$. An image on P with colors in the finite set $\mathcal{C}$ is a mapping $I : P \to \mathcal{C}$. As usual, such image can be written as a set of pairs $((i,j),x)$ where $i,j \in \{1,\ldots,n\}$ and $a = I(i,j)$. To simplify we denote a_{ij}. As we consider in this paper to work with binary images, we take $\mathcal{C} = \{b,w\}$ where b codifies black and w white.

The new technique for P systems presented in this paper consist to assign to each object codifying a pixel a label. So, we our objects pass to be of the form $(a_{ij},(i,j))$. We will see below how to use these labels to solve our problem with P systems.

From a formal point of view we can formulate our problem as follows,

Homology Groups of Binary 2D Image (HGB2I) Problem: Given a binary 2D digital image, calculate the number of black connected components and the representative curves of the holes of these components.

We have decide to consider in this paper the 4-adjacency for black pixels and the 8-adjacency for white pixels. The only one reason to do this choice is the computational complexity from a membrane models point of view. It is easier for this type of models works with 8-adjacency for blacks and 4 for whites. Moreover, in this last case the systems are very similar to the systems show below. Then, we have decided to present the membrane solution of HGB2I problem for the 4-adjacency for black pixels and the 8-adjacency for white pixels.

[4]We consider the lexicographic order for $\mathbb{N}^2$ in P.

2.1. **A membrane solution of HGB2I Problem.** In order to provide a logarithmic-time uniform solution to the our problem, we design a family of tissue-like P systems with catalyst, Π. Given an image I of size n^2, we take the system of the family $\Pi(n)$ to work with I. The input data (image I) is codified by a set of objects A_{ij} with $A = B \vee W$ and $1 \leq i, j \leq n$.

The family of P systems is defined as follows:

$$\Pi(n) = (\Gamma, \Sigma, \mathcal{E}, \omega_1, \omega_2, \omega_3, \omega_4, \mathcal{R}_1, \ldots, \mathcal{R}_{19}, \{\mathcal{R}_{15}, \mathcal{R}_{17}\} > \mathcal{R}_{10}, i_{in}, i_0)$$

where:

- $\Gamma = \{z_i : 1 \leq i \leq n+6\} \cup$
  $\{B_{ij}, b_{ij}, b'_{ij}, \bar{b}_{ij}, W_{ij}, w_{ij}, w'_{ij}, (b_{ij}, (k,l)), (w_{ij}, (k,l)) : 1 \leq i, j, k, l \leq n\} \cup$
  $\{(p_{ij}, (0,0)), (p_{ji}, (0,0)) : i = 0, n+1, 0 \leq j \leq n+1\} \cup$
  $\{A_{ijkl}, Z_{ijkl} : (1,1) \leq (i,j) < (k,l) \leq (n,n)\}$.
- $\Sigma = \{b_{ij}, w_{ij} : 1 \leq i, j \leq n\}$.
- $\mathcal{E} = \Gamma - \Sigma$.
- $\omega_1 = \emptyset$.
- $\omega_2 = \{z_1\}$.
- $\omega_3 = \{z_1, (p_{ij}, (0,0)), (p_{ji}, (0,0)) : i = 0, n+1, 0 \leq j \leq n+1\}$.
- $\omega_4 = \emptyset$.
- The sets of rules are:
  - $R_1 \equiv (1, A_{ij}/a_{ij}a'_{ij}, 0)$ for $1 \leq i, j \leq n$.
    For each pixel of our input image we generate two copies with these rules.
  - $R_2 \equiv (1, a_{ij}/\lambda, 2)$ for $1 \leq i, j \leq n$.
  - $R_3 \equiv (1, a'_{ij}/\lambda, 3)$ for $1 \leq i, j \leq n$.
    With the last two types of rules, the first copies of our pixels are sent to cell 2 and the second copies of our pixels are sent to cell 3.
    From this point, we have two parallel processes. A process associated to cell 2 (to calculate H_0) and the second one associated to cell 3 (to calculate H_1):
  - Rules associated to the H_0 process:
    - $R_4 \equiv (2, z_i/z_{i+1}, 0)$ for $1 \leq i \leq n+1$.
      These rules generate a counter that will be used in the output of the system.
    - $R_5 \equiv (2, b_{ij}/(b_{ij}, (i,j)), 0)$ for $1 \leq i, j \leq n$.
      These rules add labels to black pixels in order to work with them.
    - $R_6 \equiv (2, (b_{ij}, (k,l))(b_{i'j'}, (k',l'))/(b_{ij}, (k,l))(b_{i'j'}, (k,l))A_{klk'l'}, 0)$ for $(1,1) \leq (k,l) < (k',l') \leq (n,n)$, $1 \leq i, j, i', j' \leq n$ and (i,j), (i',j') adjacent pixels.
    - $R_7 \equiv (2, (b_{ij}, (k,l))(b_{i'j'}, (k',l'))/(b_{ij}, (k',l'))(b_{i'j'}, (k',l'))A_{k'l'kl}, 0)$ for $(1,1) \leq (k',l') < (k,l) \leq (n,n)$, $1 \leq i, j, i', j' \leq n$ and (i,j), (i',j') adjacent pixels.
      The two last types of rules change the labels of adjacent pixels, we need all the adjacent black pixels to have the same label, so we will know that they are all in the same connected component.
    - $R_8 \equiv (A_{ijkl}|2, (b_{i'j'}, (k,l))/(b_{i'j'}, (i,j)), 0)$ for $1 \leq i, j, k, l, i', j' \leq n$.
      In these rules we introduce catalysts, and process becomes faster. The

catalyst has been created when the pixel labeled by (k,l) traded its label for (i,j), so (i,j) and (k,l) are adjacent pixels and other pixels with these labels can be changed.

- $R_9 \equiv (z_{n+2}|2,(b_{ij},(i,j))/\lambda,4)$.
  With these rules we send one pixel for each connected component to the cell 2.

– Rules associated to H_1 process:

- $R_{10} \equiv (3, z_i/z_{i+1}, 0)$ for $1 \leq i \leq n+5$.
  This rule counts the number of steps of the process. We will use this to start the *Deleting Stage* after $n+2$ steps, and the *Segmenting Stage* after $n+4$ steps.
- $R_{11} \equiv (3, w_{ij}/(w_{ij},(i,j)), 0)$ for $1 \leq i,j \leq n$.
  These are the only rules used in the *Label Allocation Stage*. These rules add labels to white pixels in order to work with them.
- $R_{12} \equiv (3, (w_{ij},(k,l))(w_{i'j'},(k',l'))/(w_{ij},(k,l))(w_{i'j'},(k,l))Z_{klk'l'}, 0)$ for $(1,1) \leq (k,l) < (k',l') \leq (n,n)$, $1 \leq i,j,i',j' \leq n$ and $(i,j),(i,'j')$ adjacent pixels.
- $R_{13} \equiv (3, (w_{ij},(k,l))(w_{i'j'},(k',l'))/(w_{ij},(k',l'))(w_{i'j'},(k',l'))Z_{k'l'kl}, 0)$ for $(1,1) \leq (k',l') < (k,l) \leq (n,n)$, $1 \leq i,j,i',j' \leq n$ and $(i,j),(i',j')$ adjacent pixels.
  These two set of rules are used in *Label Conversion Stage* to compare two adjacent white pixels, and change the label of one of them. We need all the adjacent white pixels to have the same label.
- $R_{14} \equiv (Z_{ijkl}|3, (w_{i'j'},(k,l))/(w_{i'j'},(i,j)), 0)$ for $1 \leq i,j,k,l,i',j' \leq n$.
  The catalyst Z_{ijkl} acts to become the process faster. It has been created when the pixel labeled by (k,l) traded its label for (i,j), so (i,j) and (k,l) are adjacent pixels and other pixels with these labels can be changed.
- $R_{15} \equiv (z_{n+3}|3, (p_{ij},(0,0))(w_{kl},(k',l'))/(p_{ij},(0,0))(p_{kl},(0,0))Z_{00kl}, 0)$ for $(i,j),(k,l)$ 8-adjacent pixels, $0 \leq i,j \leq n+1$, $1 \leq k,l,k',l' \leq n$.
  These rules are used in *Deleting Stage* to delete white pixels which are out of the connected black component. By using 8-adjacency, we become outer white pixels into pink pixels, in order to differentiate them from the interior white pixels (holes). We will refer to the objects p_{ij} as *pink* pixels.
- $R_{16} \equiv ((Z_{00ij}|3, (w_{i'j'},(i,j))/(p_{i'j'},(0,0)), 0)$ for , $1 \leq i,j,i',j' \leq n$.
  A new catalyst acts in the same way, trading white exterior pixel for pink pixels. In this way, the *Deleting Stage* takes only 2 step.
- $R_{17} \equiv (z_{n+5}|3, (w_{ij},(i',j'))b_{kl}/(w_{ij},(i',j'))\bar{b}_{kl}, 0)$ for $1 \leq i',j',i,j,k,l \leq n$ and $(i,j),(k,l)$ 8-adjacent pixels.

  In the *Segmenting Stage* a black pixel is marked if it and a white pixel are 8-adjacent pixels. It starts after $n+2$ steps.
- $R_{18} \equiv (3, \bar{b}_{ij}/\lambda, 4)$ for $1 \leq i,j \leq n$.
  At the end, in the *Answer Stage*, black marked pixels are sent to membrane number 4, so we obtain which black pixels are containing the holes.

- $R_{19} \equiv (z_{n+6}|3, (w_{ij}, (i,j))/\lambda, 4)$ for $1 \leq i, j \leq n$.
  We want to obtain the number of holes too, so these rules send one white pixel for each hole to membrane number 4.

- $i_{in} = 1$ is the input cell.
- $i_0 = 4$ is the output cell.

We will also use priorities among rules. Rules from sets $\mathcal{R}_{15}$ and $\mathcal{R}_{17}$ are applied before rules from the set $\mathcal{R}_{10}$.

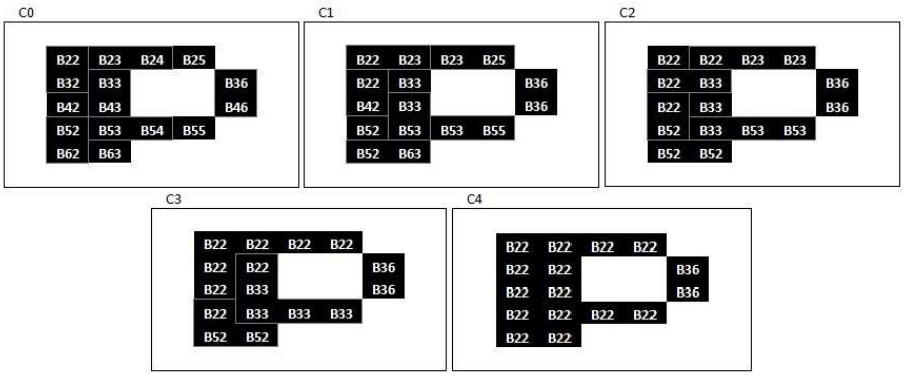

FIGURE 1. A simple example of the process to obtain H_0

Each system of the family implements the following stages:

(1) *Input Stage*: When the objects A_{ij} (with $A = B \vee W$) arrive to cell 1 the first type of rules are applied. We change these objects by objects a_{ij} and a'_{ij} (with $a = b \vee w$). In the next step, we send objects a_{ij} to cell 2 and objects a'_{ij} to cell 3. From this point we have two parallel processes: The first process occur in cell 2, and it is dedicated to obtain the number of black connected components (H_0). The second one is located in cell 3 and is dedicated to generate the representative curves of the holes of the black connected components. Moreover, system will give us the number of these holes (H_1).

(2) H_0 *Process*: It begins when objects a_{ij} arrive to cell 2.
   (a) *Label Allocation Stage*: Cell 3 trades objects b_{ij} against others with the form $(b_{ij}, (i,j))$ with the environment. The white objects are not transformed.
   (b) *Label Conversion Stage*: We can compare the black adjacent pixels by using catalyst, and we trade the label of the greatest pixel against the label of the other pixel; i.e. $(i, (b_{ij}, (i',j'))(b_{kl}, (k',l'))/(b_{ij}, (i',j'))(b_{kl}, (i',j'))A_{i'j'k'l'}, j)$, where (i,j) and (k,l) are adjacent pixels. Moreover, we can see a new object arriving to cell i. It is a catalyst and it is used to codify if two labels must be compared. Later, they are connected, and one of them can be changed by the other one, as we can see in the Figure 1.
   (c) *Answer Stage*: In the step $n+2$, the object z_{n+2} arrives to the cell 1 due to the counter. It is used by the system as a catalyst, and the objects with the form $(b_{ij}, (i,j))$ are sent to the output cell representing each one to a black connected

component. The P system have used $n+2$ steps to obtain the number of black connected components of an n^2 image.

Figure 1 shows a sequence of configurations of the process to obtain H_1 for input data given by the configuration $C0$ of the picture.

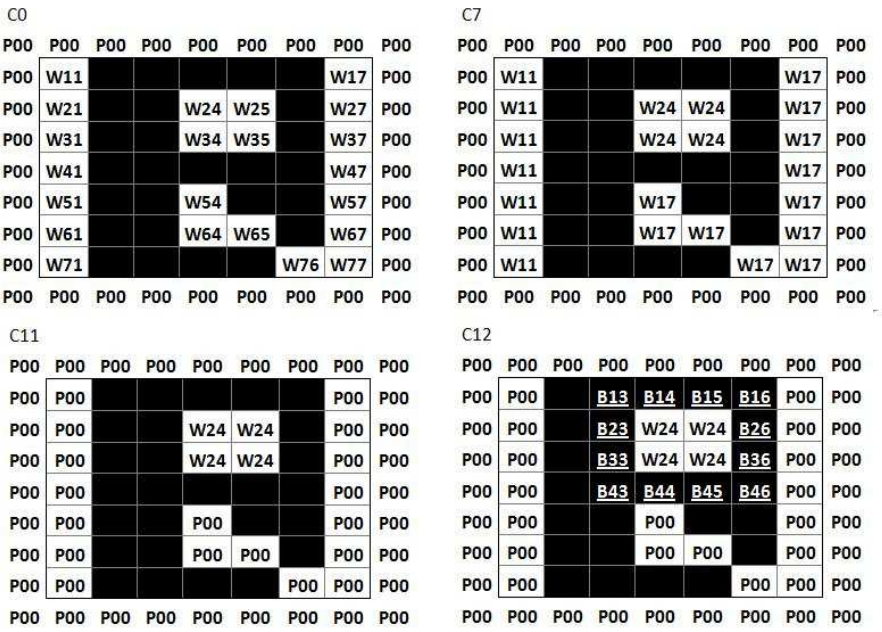

FIGURE 2. Representative configurations of a simple example to obtain H_1

(3) H_1 *process*: It begins when objects a'_{ij} arrive to cell 3.
   (a) *Label Allocation Stage*: Cell 3 trades objects w'_{ij} against others with the form $(w_{ij}, (i, j))$ with the environment.
   (b) *Label Conversion Stage*: We compare the label of two white adjacent pixels, and we trade the label of the greatest pixel against the label of the other pixel; i.e., we use rules with the form $(i, (w_{ij}, (i', j'))(w_{kl}, (k', l'))/(w_{ij}, (i', j'))(w_{kl}, (i', j'))Z_{i'j'k'l'}, j)$, where (i, j) and (k, l) are adjacent pixels. Moreover, we can see a new object arriving to cell i, $Z_{i'j'k'l'}$. It is a catalyst and is used to codify when two labels must be compared. Then, the labels are connected, and one of them can be changed by the other one, as we can see in $C7$ in the Figure 2.
   (c) *Deleting Stage*: Initially, system keeps in cell 1 a set of objects codifying the frame of the input image $(p_{0i}, p_{n+1i}, p_{i0}, p_{in+1}$ for $i = 0, \ldots, n+1)$ with the label $(0, 0)$ associated. When the input data is introduced in the system, the white pixels not contained inside of black connected components are sent to the environment to trade against of objects with the form of the frame. We need a linear number of steps with respect to n to eliminate all the possible white pixels. We can see the result in $C11$ in the Figure 2.

(d) *Segmenting Stage*: This part begins when deleting stage finishes due to the counter z_i (rules R_1). If there are white pixels in cell 1 in this step are in a hole. The P system takes pairs of adjacent pixels, one black and the other white, adding a mark to the black pixels of these pairs. Then, we have marked the black pixels adjacent to a hole. We need a constant number of steps to segment an image with P systems. Figure 2 shows in $C12$ how the holes of the image are codified.

(e) *Answer Stage*: We send the marked black pixels to output cell in the following step to be marked. So, we obtain, the representative curves of the holes in the image I. We also send white pixels which keep their labels, there is only one pixel for each connected white component, ie, for each hole in the image. We only need one step more with respect to the segmenting stage.

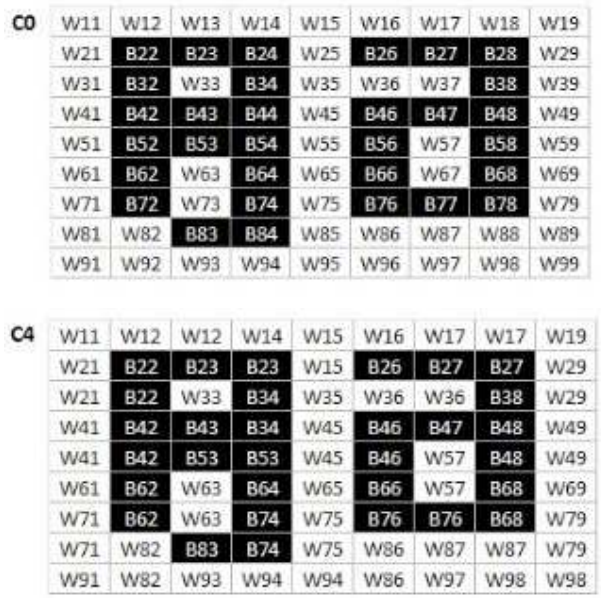

FIGURE 3. Initial and fourth configuration

2.2. **Example.** In this section we will show with a simple example our technique to obtain the homology groups H_0, and H_1 of a $2D$ image by using membrane computing. Let us consider the first image in Figure 3 which have 9×9 pixels. The black pixels represent two connected components, the white pixels represent the exterior of the components and the holes, one in each connected component. In the initial configuration we have 4 cells. There is one object A_{ij} (with $A = B \vee W$) in cell 1 codifying each pixel. At the beginning of the process the system duplicates these elements and sends one copy to the membrane 2 on order to calculate H_0, and the other copy to the membrane 3 in order to calculate H_1. These two processes are working at the same time, so we obtain a parallel system.

The process to calculate H_0 is the following one: The cell 1 has the object z_1 working as a counter, it allow us start a different stage after the necessary number of the steps. In the

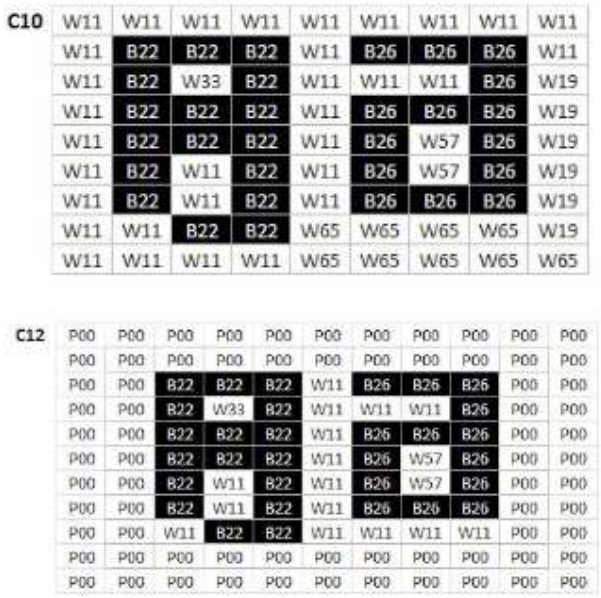

FIGURE 4. Configuraciones C10 y C12

third step labels (i, j) are added to objects a_{ij}, we need these labels to change them within change the original position of the pixels. In the fourth step (configuration C4 in Figure 3) the *label conversion stage* lets start. The aim is to have all the pixels in the same connected component with the same label. Rules R_6, R_7 changed the label of two adjacent pixels, to become these in the same label. These rules created catalysts to become the process faster. A catalyst represents that two labels are located in adjacent pixels, and it acts in subsequent steps changing, at the same time, the labels of several pixels. In our example, we need seven steps from the beginning of the stage until have black pixels with the correct label. We can see in configuration C10 in Figure 4 all the black pixels of each connected component with the same label. When the counter reaches the appropriate subindex, a representative object for each connected component is sent to cell 4. We obtain two objects $(b_{22}, (2, 2))$, $(b_{26}, (2, 6))$, so our image has two connected components.

The process to calculate H_1 is the following one: The beginning of this process is the same of the previous one. In the cell 3, white pixels are changing its labels until all the pixels in the exterior of the picture have the same label, and all the pixels in the holes of the components have also the same label (see configuration C12). In the 12th step the *deleting stage* lets start. We need to know how many holes there are, so we have to "delete" white exterior pixels. We add a pink frame around the picture, pink pixels acts in white exterior pixels becoming them into pink pixels too. By using catalysts we only need two step to color pixels in pink. The configuration C14 (see Figure 5) shows us the result. With the rules R_{17} we can also mark black pixels adjacent to white pixels, in order to obtain information not only about how many holes there are, but about the shape of these holes.

FIGURE 5. Configuration C14

2.3. **Complexity and Necessary Resources.** Bearing in mind the size of the input data is $O(n^2)$, the amount of necessary resources for defining the systems of our two families and the complexity of our problems can be observed in the following table:

HGB2I Problem	
Complexity	
Number of steps of a computation	$n+9$
Necessary Resources	
Size of the alphabet	$O(n^4)$
Initial number of cells	4
Initial number of objects	$O(n)$
Number of rules	$O(n^6)$
Upper bound for the length of rules of the systems	5

3. Conclusions and Future Works

Digital Imagery, treated by techniques of Algebraic Topology, can be suitable for Membrane Computing techniques. The starting point is that such problems can be treated locally by a set of processors, the information can be split into little pieces and expressed as (multi)sets of objects and the computation steps can be processed by re-writing-type rules.

In many cases, the same sequential algorithm must be applied in different regions of the image which are independent. All these features lead us to consider parallel bio-inspired computational models to deal with Digital Imagery.

This paper can be seen as a first attempt of formalizing the bridges between Membrane Computing and Algebraic Topology presented recently by Cristinal *et al.* [8, 9, 10]. Many research lines are open. In this paper, we have showed that the homological groups to 2D images can be obtained by using tissue-like P systems with catalyst, but a deeper study is necessary. The research lines related to the most suitable P system model for Homology Theory problems or which are the most relevant features of P systems which can represent the nature of the problems are open. From the Algebraic Topology point of view, the

question is to find new representations and new problems which can be expressed and dealt with Membrane Computing techniques.

In future we wish to to use P systems to obtain more homological information: homology groups, spanning trees, homology gradient vector field, etc. Until now, this homological information is typically calculated using sequential algorithms or, in the best case, partially parallel algorithms. Then, we can use P systems in some research fields where the homological information is important: 2D, 3D and 4D Image, Robotics, etc.

Acknowledgement

DDP and MAGN acknowledge the support of the projects TIN2008-04487-E and TIN-2009-13192 of the Ministerio de Ciencia e Innovación of Spain and the support of the Project of Excellence with *Investigador de Reconocida Valía* of the Junta de Andalucía, grant P08-TIC-04200. PR and VSC acknowledge the support of the project MTM2009-12716 of the Ministerio español de Educación y Ciencia, the project PO6-TIC-02268 of Excellence of Junta de Andalucía, and the "Computational Topology and Applied Mathematics" PAICYT research group FQM-296.

References

[1] Leonard M. Adleman, *Molecular computation of solutions to combinatorial problems*, Science **266** (1994), 1021–1024.

[2] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter, *Molecular biology of the cell*, fourth ed., Garland Science, London, UK, 2002.

[3] Artiom Alhazov, Rudolf Freund, and Marion Oswald, *Tissue P systems with antiport rules and small numbers of symbols and cells*, Developments in Language Theory (Clelia de Felice and Antonio Restivo, eds.), Lecture Notes in Computer Science, vol. 3572, Springer, 2005, pp. 100–111.

[4] Francesco Bernardini and Marian Gheorghe, *Cell communication in tissue P systems: universality results*, Soft Computing **9** (2005), no. 9, 640–649.

[5] Rodica Ceterchi, Radu Gramatovici, Natasa Jonoska, and K. G. Subramanian, *Tissue-like P systems with active membranes for picture generation*, Fundamenta Informaticae **56** (2003), no. 4, 311–328.

[6] Rodica Ceterchi, Madhu Mutyam, Gheorghe Păun, and K. G. Subramanian, *Array-rewriting P systems*, Natural Computing **2** (2003), no. 3, 229–249.

[7] J. Chao and J. Nakayama, *Cubical singular simplex model for 3d objects and fast computation of homology groups*, 13th International Conference on Pattern Recognition (ICPR'96) (Los Alamitos, CA, USA), vol. IV, IEEE Computer Society, 1996, pp. 190–194.

[8] Hepzibah A. Christinal, Daniel Díaz-Pernil, and Pedro Real, *Segmentation in 2d and 3d image using tissue-like P system*, CIARP (Eduardo Bayro-Corrochano and Jan-Olof Eklundh, eds.), Lecture Notes in Computer Science, vol. 5856, Springer, 2009, pp. 169–176.

[9] Hepzibah A. Christinal, Daniel Díaz-Pernil, and Pedro Real, *Using membrane computing for obtaining homology groups of binary 2d digital images*, IWCIA (Petra Wiederhold and Reneta P. Barneva, eds.), Lecture Notes in Computer Science, vol. 5852, Springer, 2009, pp. 383–396.

[10] Hepzibah A. Christinal, Daniel Díaz-Pernil, and Pedro Real, *P systems and computational algebraic topology*, Journal of Mathematical and Computer Modelling (2010), In press.

[11] A. Engelbrecht, *Fundamentals of computational swarm intelligence*, 2005.

[12] Rudolf Freund and Oscar H. Ibarra, *Catalytic P systems*, ch. 4, pp. 83 – 117, in Păun et al. [27], 2010.

[13] Rudolf Freund, Lila Kari, Marion Oswald, and Petr Sosík, *Computationally universal P systems without priorities: two catalysts are sufficient*, Theoretical Computer Science **330** (2005), no. 2, 251–266.

[14] Rudolf Freund, Gheorghe Păun, and Mario J. Pérez-Jiménez, *Tissue P systems with channel states*, Theoretical Computer Science **330** (2005), no. 1, 101–116.

[15] John H. Holland, *Adaptation in natural and artificial systems*, MIT Press, Cambridge, MA, USA, 1992.

[16] Lila Kari and Grzegorz Rozenberg, *The many facets of natural computing*, Communications of the ACM **51** (2008), no. 10, 72–83.
[17] Shankara Narayanan Krishna, *On pure catalytic P systems*, UC (Cristian S. Calude, Michael J. Dinneen, Gheorghe ăun, Grzegorz Rozenberg, and Susan Stepney, eds.), Lecture Notes in Computer Science, vol. 4135, Springer, 2006, pp. 152–165.
[18] Shankara Narayanan Krishna, Kuppuswamy Lakshmanan, and Raghavan Rama, *Tissue P systems with contextual and rewriting rules*, WMC-CdeA (Gheorghe Păun, Grzegorz Rozenberg, Arto Salomaa, and Claudio Zandron, eds.), Lecture Notes in Computer Science, vol. 2597, Springer, 2002, pp. 339–351.
[19] Shankara Narayanan Krishna and Andrei Păun, *Results on catalytic and evolution-communication P systems*, New Generation Computing **22** (2004), no. 4.
[20] Carlos Martín-Vide, Gheorghe Păun, Juan Pazos, and Alfonso Rodríguez-Patón, *Tissue P systems*, Theoretical Computer Science **296** (2003), no. 2, 295–326.
[21] Warren S. McCulloch and Walter Pitts, *A logical calculus of the ideas immanent in nervous activity*, Neurocomputing: foundations of research (1988), 15–27.
[22] Andrei Păun and Gheorghe Păun, *The power of communication: P systems with symport/antiport*, New Generation Computing **20** (2002), no. 3, 295–306.
[23] Gheorghe Păun, *Computing with membranes*, Tech. Report 208, Turku Centre for Computer Science, Turku, Finland, November 1998.
[24] Gheorghe Păun, *Computing with membranes*, Journal of Computer and System Sciences **61** (2000), no. 1, 108–143, See also [23].
[25] Gheorghe Păun, *Membrane computing. An introduction*, Springer-Verlag, Berlin, Germany, 2002.
[26] Gheorghe Păun, Mario J. Pérez-Jiménez, and Agustín Riscos-Núñez, *Tissue P systems with cell division*, International Journal of Computers, Communication and Control **3** (2008), no. 3, 295–303.
[27] Gheorghe Păun, Grzegorz Rozenberg, and Arto Salomaa (eds.), *The Oxford handbook of membrane computing*, Oxford University Press, 2010.
[28] Samuel Peltier, Adrian Ion, Yll Haxhimusa, Walter G. Kropatsch, and Guillaume Damiand, *Computing homology group generators of images using irregular graph pyramids*, GbRPR (Francisco Escolano and Mario Vento, eds.), Lecture Notes in Computer Science, vol. 4538, Springer, 2007, pp. 283–294.
[29] John von Neumann, *Theory of self-reproducing automata*, Univ. of Illinois Press, Urbana, IL, 1966.
[30] *P system web page. http://ppage.psystems.eu/.*

[1] Research Group on Computational Topology and Applied Mathematics, Universidad de Sevilla, Avda. Reina Mercedes s/n, 41012, Sevilla, Spain.
Email address: {sbdani,real,vscanales}@us.es

[2] Research Group on Natural Computing, Universidad de Sevilla, Avda. Reina Mercedes s/n, 41012, Sevilla, Spain.
Email address: magutier@us.es