Obtaining Homology Groups in Binary 2D Images Using P Systems

Hepzibah A. Christinal #,*1, Daniel Díaz-Pernil #2, Pedro Real #3

# *Research Group on Computational Topology and Applied Mathematics, Universidad de Sevilla*
Avda. Reina Mercedes s/n, 41012, Sevilla, Spain
2 sbdani@us.es
3 real@us.es

* *Karunya University*
Coimbatore, Tamilnadu, India
1 hepzi@us.es

Abstract—**Membrane Computing is a new paradigms inspired from cellular communication. We use in this paper the computational devices called P systems to calculate in a general maximally parallel manner the homology groups of binary 2D images. So, the computational time to calculate this homology information only depends on the thickness of them.**

I. Introduction

Natural Computing studies new computational paradigm inspired from Nature. It abstracts the way in which Nature "computes", conceiving new computing models. There are several fields in Natural Computing that are now well established. Among them, Genetic Algorithms introduced by J. Holland[13] which is inspired by natural evolution and selection in order to find an optimal solution in a large set of feasible candidate solutions; Neural Networks introduced by W.S. McCulloch and W. Pitts[14] which is based on the interconnections of neurons in the brain; or DNA-based molecular computing, that was initiated when L. Adleman[1] published a solution to an instance of the Hamiltonian path problem by manipulating DNA strands in a lab.

Membrane Computing[19] is a new field in Natural Computing that is originated from the assumption that the processes taking place within the compartmental structure of a living cell can be interpreted as computations. In particular, we focus 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 place multisets of objects which evolve according to given rules in a synchronous non-deterministic maximally parallel manner. Since the seminal paper [18], different models of P systems have been studied. According to their architecture, these models can be split into two sets: cell-like P systems and tissue-like P systems[23], [4]. In cell-like P systems, membranes are hierarchically arranged in a tree-like structure. The inspiration for such architecture is the set of vesicles inside the cell. All of them perform their biological processes in parallel and life is the consequence of the harmonious conjunction of such processes.

We work here with *basic P system*. This election is doubt these models appeared before to tissue-like P systems, and it is not necessary to use a dynamical membrane structure.

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 Poincar´e. Homology groups (related to the "different" n-dimensional holes -connected component, tunnels, cavities,...- a geometric object has) are invariants from Algebraic Topology which are frequently used in Digital Image Analysis and Structural Pattern Recognition. In some sort, they reflects the topological nature of the object in terms of the number and characteristics of its holes. 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.

J. Chao and J. Nakayama connected Natural Computing and Algebraic Topology using Neural Networks[3] (extended Kohonen mapping). We begin in this paper a new research line. We use, by first time, the power and efficiency of P systems[11], [21], [10] to calculate the homology groups to binary 2D images. The parallelism is massive in this model, so the time used to obtain the homology groups not depends of the number of connected components or the number of the holes, but only depends of the thickness of them.

The paper is structured as follows: in the next section we present the definition of basic P systems with input. In Section III, we design two systems to calculate H_0 and H_1 of any 2D image ($n \times n$) and present an specific 8×8 image to show how these systems work. Finally, some conclusions and future work are given in the last section.

II. Formal Framework

We will present a description, detailed but informal, of a model of *Cellular Computation with Membranes* introduced by Gh. Păun in [18]: *basic P systems*.

A P system has, basically, a set of membrane, organized by a membrane structure than can be viewed as a rooted tree. There exists a *skin membrane* (root) containing all the others membranes inside it and separating the system of the environment. Moreover, the membranes that not contain others membranes inside it are called *elemental membranes* (leaves of the tree). The *regions* delimited by the membranes (i.e., the space between a membrane and inside membranes to it, if there exist someone) can contain some *objects* that can appear repeatedly. By means of application of specific *evolution rules* associated to membranes, these objects can be transformed in others, and these can go of one regions to other adjacent one crossing the membrane between them.

Following this idea we begin to define a membrane structure, μ, as a element of MS set defined by recursion as follows:

1) $[\,] \in MS$.
2) If $\mu_1, \ldots, \mu_k \in MS$ (con $k \geq 1$), then $[\mu_1 \ldots \mu_k] \in MS$.

Each pair of coincident brackets appearing in μ is called *membrane*. The pair of external brackets is called *skin membrane*. We call *elemental membranes* to the pair of brackets without others brackets inside it. The *grade* of μ is the number of membranes containing is μ.

Let two membranes be, m_1 and m_2 of μ, such as the second one is contained in the first ($m_2 \subset m_1$). If there not exists any membrane m' contained in m_1 and containing to m_2 ($m_2 \subset m' \subset m_1$) then these membranes are adjacent. In this case, we can say m_1 is the *father membrane* of m_2, and m_2 is the *daughter membrane* of m_1.

In the following, we informally present the *sintaxis* of the basic model of cellular computation with membranes.

A *basic* P *system* of grade $q \geq 1$ is a tuple

$$\Pi = (\Gamma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, (R_1, \rho_1), \ldots, (R_q, \rho_q), o_{Pi})$$

where:

- Γ is a finite alphabet, whose elements are called objects.
- μ is a membrane structure. The membranes are labeled using natural numbers from 1 to q (it means μ contain exactly q membranes identify by its label).
- $\mathcal{M}_i$ is a finite multiset over Γ associate to membrane of the system with label $i \in \mathbb{N}$, for each $i = 1, \ldots, q$.
- R_i is a finite set of evolution rules associate to the membrane of the system labeled by i, for each $i = 1, \ldots, q$. An evolution rule is a pair (u, v), normally represented $u \to v$, where u is a string over Γ and $v = v'$ or $v = v'\delta$, being v' a string over $\Gamma \times (\{here, out\} \cup \{in_i : i = 1, \ldots, q\})$. The length of the string is called *radio* of this rule.
- ρ_i is a strict partial order over R_i, for each $i = 1, \ldots, q$, establishing a priority relation between rules of R_i.
- *omemb* is a natural number between 1 and q indicating the output membrane of the system.

A *basic* P *system with input* of grade $q \geq 1$ is a tuple (Π, Σ, i_{Pi}), where:

- Π is a basic P system with working alphabet Γ, with q membranes labeled by $1, \ldots, q$ and initial associated multisets $\mathcal{M}_1, \ldots, \mathcal{M}_q$, respectively.
- Σ is an (input) alphabet strictly contained in Γ.
- $\mathcal{M}_i$ is a multiset over alphabet $\Gamma \setminus \Sigma$, for each $i = 1, \ldots, q$.
- i_{Pi} is the label of a distinguished membrane representing the input membrane.

To describe the *semantic* of the model it is previously introduce the concept of configuration of the system, and the notion of computation of the same.

A *configuration* of a basic P system $\Pi = (\Gamma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, (R_1, \rho_1), \ldots, (R_q, \rho_q), m)$ is a tuple $(\mu', M_{i_1}, \ldots, M_{i_t})$ such as:

- μ' is the membrane structure obtained from μ when some membranes different to those labeled by $i_1, \ldots, i_t$ are eliminated. Skin membrane can not be eliminated.
- M_{i_j} is a finite multiset over Γ, for each $j = 1, \ldots, t$.

The *initial configuration* of the basic P system, $\Pi = (\Gamma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, (R_1, \rho_1), \ldots, (R_q, \rho_q), m)$, is the tuple $(\mu, \mathcal{M}_1, \ldots, \mathcal{M}_q)$.

Let (Π, Σ, i_{Pi}) be a basic P system with input. Let Γ, μ and $\mathcal{M}_1, \ldots, \mathcal{M}_q$ be the working alphabet, the membrane structure and the initial multisets of Π. Let m be a multiset over the input alphabet Σ. Then the initial configuration of the system with input m is $(\mu, \mathcal{M}_1, \ldots, \mathcal{M}_{i_\Pi} + m, \ldots, \mathcal{M}_q)$.

The application of a rule $u \to v$ associated to a membrane with label i presented in a configuration is realized as follows: the objects in u are eliminated of the membrane with label i (then, this membrane must contain suffer objects in each membrane to apply this rule); then, for each pair $(s, out) \in v$ an object $s \in \Gamma$ is included in the father membrane of i (or it leaves the system if i is the skin); for each $(s, here) \in v$ an object $s \in \Gamma$ is added to the membrane i; for each pair $(s, in_j) \in v$ an object $s \in \Gamma$ is introduced in the membrane j (if the membrane j is not a daughter membrane of i, then the rule can not be applied); at the end, if $\delta \in v$, then the membrane i is dissolved; i.e., it is eliminated of the membrane structure, passing its objects to the first ascendent membrane not dissolve (the skin can not be dissolved).

In the other way, we interpret the priority relation between rules in a *strong sense*: this relation forbid the application of a rule if it can be applied other with bigger priority.

Given two configurations, C y C', of Π, we call that C' is obtained from C in a transition step, $C \Rightarrow_\Pi C'$, if the second is the result by applying the first, in parallel (simultaneously), and for all membranes at the same time, a maximal subset of evolution rules associated to the membranes that they appear in the membrane structure of C. In this multiset is indicated the number of times that each rule is applied. This subset must be maximal, in the sense, when the application of rules is ended any object that can evolve and activate a rule not appear in any membrane of the system. Bearing in mind, for a configuration, usually there exist more than one multiset of applicable rules, the transition step are realized in a non deterministic manner; i.e., a configuration of the system can have more than one following configuration.

A computation $\mathcal{C}$ of Π is a succession (finite or infinite) of configurations, $\{C_i\}_{i<r}$, such as

- C_0 is the (or a) initial configuration of Π.
- $C_i \Rightarrow_\Pi C_{i+1}$, for all $i < r - 1$.
- Or $r \in \mathbb{N}^+$ (i.e., is a natural number distinct of zero) and there not exists any rule that it can apply in any membrane of C_{r-1} (in this case $\mathcal{C}$ is an stopped computation, and it has realized $|\mathcal{C}| = r - 1$ steps and C_{r-1} is its *stopped configuration*), or $r = \infty$ (in this case $\mathcal{C}$ is not an stopped computation).

We say a computation $\mathcal{C}$ is *successful* if it is a stopped computation and the output membrane o_Π appear in C_{r-1} as an elemental membrane.

Given a successful computation of Π, the output of this computation is codified by the content of the output membrane; so, for example, it could define the output of this computation as the size of the associated multiset to the output membrane of the system in its stopped configuration. Then, a basic P system is a machine that it generate the set of natural numbers, $N(\Pi)$, forming by the output of the all successful computations of this system.

III. Calculating Homology groups

Homology is a powerful topological invariant, which characterizes an object by its p-dimensional holes. Intuitively the 0-dimensional holes can be seen as connected components, 1-dimensional holes can be seen as tunnels and 2-dimensional holes as cavities. For example, the torus contains one 0-dimensional hole, two 1-dimensional holes (each of them are an edge cycle) and one 2-dimensional hole (the cavity enclosed by the entire surface of the torus).

In the following, we will try to calculate homology groups, H_0 and H_1, to a 2D digital images. So, we can codify the images with multiple pixels forming a network of points of $\mathbb{N}^2$. Moreover, we will suppose each pixel has associated one of the two possible colors, black and white. Then, the black or white pixel in the position (i, j) can be codified by the object B_{ij} or W_{ij}.

H_0 is given by the number of connected components formed by the black pixels and H_1 is given by the number of the holes created by the black pixels; i.e., the number of connected components of white pixels surrounded by black pixels. But, there exists a previous problem when we want to calculate the homology groups to a 2D image: the adjacency concept. There exists two natural definitions: 4-adjacency and 8-adjacency. In the first we consider a pixel K_{ij} (where $K = B \vee K = W$), the list of adjacent pixels to this is $\{K_{ij-1}, K_{ij+1}, K_{i-1j}, K_{i+1j}\}$ i.e.; the adjacent pixels to any pixel $K_{i,j}$ are just up, down, right and left of this, such we can observe in the following:

$$\begin{array}{ccc} & B & \\ B & K & W \\ & W & \end{array}$$

In the second we consider the pixel K_{ij} (where $K = B \vee K = W$), the list of adjacent pixels to this is $\{K_{i-1j-1}, K_{i-1j}, K_{i-1j+1}, K_{ij-1}, K_{ij+1}, \quad K_{i+1j-1}, K_{i+1j}, K_{i+1j+1}\}$ i.e.; the adjacent pixels to a any pixel $K_{i,j}$ are just up, down, right and left of this and, moreover, we consider the diagonal objects, such we can observe in the following:

We will consider to work in this paper with 4-adjacency, because from a computational point of view is easier to work with 8-adjacency.

Then, we can consider the *black graph associated to the image*, $G_B = (V, E)$, where the vertices are formed by the black pixels and there exists an edge between two black pixels if they are adjacent.

So, from this point we just can try to solve our problem: to obtain the homology groups of a 2D image.

A. A P system to obtain H_0

In this point, we want to know the number of connected components formed with the black pixels. So, for each image with n^2 pixels ($n \in \mathbb{N}$) we will construct a basic P system where we will have three types of elements: B_{ij} codifying the black pixels, W_{ij} codifying the white pixels and G_{ij}. For each G_{ij} that appear in the system is sent to the environment one object C. The number of objects C that appear in the environment when the system stop is the number of connected components.

Next, we present a family of basic P system where at the initial configuration has one membrane. We shall address the resolution via an parallel algorithm, which consists in the following:

By the rules of the system we work which each connected components at the same time. We will reduce each one to a only one object G_{ij}.

For each $n \in \mathbb{N}$ we will consider the basic P system

$$\Pi_0(n) = (\Gamma, \Sigma, \mu, \mathcal{M}_0, \mathcal{M}_1, (R_0, \rho_0), (R_1, \rho_1), i_\Pi, o_\Pi)$$

defined as follows

a) $\Gamma = \Sigma \cup \{G_{ij} : 1 \le i, j \le n\} \cup \{C\}$,
b) $\Sigma = \{B_{ij}, W_{ij} : 1 \le i, j \le n\}$,
c) $\mu = [[]_1]_0$,
d) $M_0 = \emptyset$, $M_1 = \{W_{ij} : (i = 0 \wedge 1 \le j \le n) \vee (i = n + 1 \wedge 1 \le j \le n) \vee (j = 0 \wedge 1 \le i \le n) \vee (j = n + 1 \wedge 1 \le i \le n)\}$,
e) $R_0 = \emptyset$, R_1 is the following set of rules:

1)

$$\begin{array}{ccc} & W & K \\ W & B & B \\ & W & B \end{array} \rightarrow \begin{array}{ccc} & W & K \\ W & W & B \\ & W & B \end{array}$$

where $K = B$ or $K = W$. There exists 8 rules of this type depending the position of the black

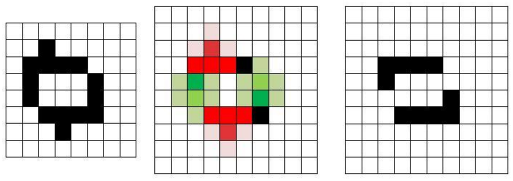

Fig. 1. Cutting branches of two black connected components

pixels (up, down, left and right) and the value of K. Formally, one of these rules is written as follows:

$$W_{i-1j}W_{ij-1}B_{ij}W_{ij+1}K_{i+1j-1}B_{i+1j}B_{i+1j+1} \rightarrow$$
$$W_{i-1j}W_{ij-1}W_{ij}W_{ij+1}K_{i+1j-1}B_{i+1j}B_{i+1j+1}$$

for $1 \leq i, j \leq n-1$.

2)

$$\begin{array}{ccccccc} & W & W & & & W & W \\ W & B & B & \rightarrow & W & W & B \\ & W & W & & & W & W \end{array}$$

There exists 2 rules of this type depending the position of the white pixel in the center line respect to two black pixels (left and right). Formally, one of these rules is written as follows:

$$W_{i-1j}W_{ij-1}B_{ij}W_{ij+1}W_{i+1j}B_{i+1j}W_{i+1j+1} \rightarrow$$
$$W_{i-1j}W_{ij-1}W_{ij}W_{ij+1}W_{i+1j}B_{i+1j}W_{i+1j+1}$$

for $1 \leq i \leq n-1$ and $1 \leq j \leq n$.

The two first types of rules eliminate the black pixels connected to other black pixel which is a ***cutting vertex*** of the G_B as we can see in the Figure 1.

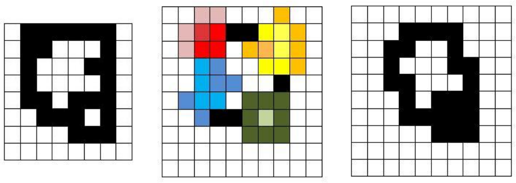

Fig. 2. Reducing black connected components with rules of types 5 to 8.

The rules of types from 3 to 7 are used to reduce the dimensions of the black connected components with white pixels inside them, as we can see in Figure 2 where we have colored pixels used by these types of rules with different colors: red, blue, yellow and green, respectively.

3)

$$\begin{array}{ccccccc} & W & & & & W & \\ W & B & B & \rightarrow & W & W & B \\ & B & B & & & B & B \end{array}$$

There exists 4 rules of this type depending the position of the black pixel (up, down, left and right). Formally, one of these rules is written as follows:

$$W_{i-1j}W_{ij-1}B_{ij}B_{ij+1}B_{i+1j}B_{i+1j+1} \rightarrow$$
$$W_{i-1j}W_{ij-1}W_{ij}B_{ij+1}B_{i+1j}B_{i+1j+1}$$

for $1 \leq i, j \leq n-1$.

4)

$$\begin{array}{ccccccccc} & W & & & & & W & & \\ W & B & B & B & \rightarrow & W & W & B & B \\ & B & W & W & & & B & B & W \\ & & W & & & & & W & \end{array}$$

There exists 4 rules of this type depending of the orientation of the black pixel. Formally, one of these rules is written as follows:

$$W_{i-1j}W_{ij-1}B_{ij}B_{ij+1}B_{i+1j}W_{i+1j+1}$$
$$W_{i+1j+2}B_{i+2j}W_{i+2j+1} \rightarrow$$
$$W_{i-1j}W_{ij-1}W_{ij}B_{ij+1}B_{i+1j}W_{i+1j+1}$$
$$W_{i+1j+2}B_{i+2j}W_{i+2j+1}$$

for $1 \leq i \leq n-2$ and $1 \leq j \leq n-1$.

5)

$$\begin{array}{ccccccccc} W & W & W & & & W & W & W & \\ B & B & B & W & \rightarrow & B & W & W & W \\ B & W & B & & & B & B & B & \\ & W & & & & & W & & \end{array}$$

There exists 8 rules of this type depending of the orientation of the black pixel and the position of the white pixel in the second line respect to the black pixels. Formally, one of these rules is written as follows:

$$W_{i-1j-1}B_{i-1j}B_{i-1j+1}W_{ij-1}B_{ij}W_{ij+1}W_{ij+2}$$
$$W_{i+1j-1}B_{i+1j}B_{i+1j+1}W_{i+2j} \rightarrow$$
$$W_{i-1j-1}B_{i-1j}B_{i-1j+1}W_{ij-1}W_{ij}B_{ij+1}$$
$$W_{ij+2}W_{i+1j-1}W_{i+1j}B_{i+1j+1}W_{i+2j}$$

for $1 \leq i, j \leq n-1$.

6)

$$\begin{array}{ccccccc} B & B & B & & B & B & B \\ B & W & B & \rightarrow & B & B & B \\ B & B & B & & B & B & B \end{array}$$

Formally, this rule is written as follows:

$$B_{i-1j-1}B_{i-1j}B_{i-1j+1}B_{ij-1}W_{ij}$$
$$B_{ij+1}B_{i+1j-1}B_{i+1j}B_{i+1j+1} \rightarrow$$
$$B_{i-1j-1}B_{i-1j}B_{i-1j+1}B_{ij-1}B_{ij}B_{ij+1}$$
$$B_{i+1j-1}B_{i+1j}B_{i+1j+1}$$

for $1 \leq i, j \leq n-1$.

7)

$$\begin{array}{ccccccccc} & W & W & & & & W & W & \\ W & B & B & W & \rightarrow & W & B & W & W \\ W & B & B & W & & W & W & W & W \\ & W & W & & & & W & W & \end{array}$$

Formally, this rule is written as follows:

$$W_{i-1j}W_{i-1j+1}W_{ij-1}B_{ij}B_{ij+1}$$
$$W_{ij+2}W_{i+1j-1}B_{i+1j}B_{i+1j+1}$$
$$W_{i+1j+2}W_{i+2j}W_{i+2j+1} \rightarrow$$
$$W_{i-1j}W_{i-1j+1}W_{ij-1}B_{ij}W_{ij+1}$$
$$W_{ij+2}W_{i+1j-1}W_{i+1j}W_{i+1j+1}$$
$$W_{i+1j+2}W_{i+2j}W_{i+2j+1}$$

for $1 \leq i, j \leq n-1$.

8)
$$\begin{array}{ccccccc} & W & & & & W & \\ W & B & W & \rightarrow & W & G & W \\ & W & & & & W & \end{array}$$
Formally, this rule is written as follows:
$$W_{i-1j}W_{ij-1}B_{ij}W_{ij+1}W_{i+1j} \rightarrow$$
$$W_{i-1j}W_{ij-1}G_{ij}W_{ij+1}W_{i+1j}$$
for $1 \leq i, j \leq n-1$.

9) $G_{ij} \rightarrow W_{ij}(C, out)$ for $1 \leq i, j \leq n$. This first type of rules sends an object C to the environment and removes the object G_{ij} and adds the object W_{ij} to the membrane of the system.

f) $\rho_0 = \rho_1 = \emptyset$,

g) $i_\Pi = 1$,

h) $o_\Pi = 0$.

Overview of the Computation:

Given an image as input data whose size is $O(n^2)$, it exists a system of this family working in a parallel manner: First, it eliminates branches of the black connected components that appear in the image using as much 4 steps. For this, system uses rules of type 1 and 2. Secondly, system reduces the size of the black connected components from four directions: up, down, left and right. System takes the rules of types 3 to 7 to realizes this task. For this, system needs a logarithmic number of steps respecting to the size of the biggest black connected component to reduce to a only one black pixel each component. In fact, system can use these types of rules in a parallel manner while the size of the black connected components is big (even, it could take more than one rule of each type in several computation step and it could be necessary to use rules of type 1 and 2 again). Finally, it uses the rules of type 8 to identify the connected component and rules of type 9 to send one object C for each component to the environment.

Complexity and Necessary Resources:

Taking account the size of the input data is $O(n^2)$, the amount of necessary resources for defining the systems of our family and the complexity of our problem can be observed in the following table:

H_0 Problem	
Complexity	
Number of steps of a computation	$O(n)$
Necessary Resources	
Size of the alphabet	$3n^2+1$
Initial number of cells	1
Initial number of objects	$4n-3$
Number of rules	$O(n^2)$
Upper bound for the length of the rules	24

B. A P system to obtain H_1

For each $n \in \mathbb{N}$ we will consider the basic P system
$\Pi_1(n) = (\Gamma, \Sigma, \mu, \mathcal{M}_0, \mathcal{M}_1, \mathcal{M}_2, (R_0, \rho_0), (R_1, \rho_1), (R_1, \rho_2), i_\Pi, o_\Pi)$
defined as follows

a) $\Gamma = \Sigma \cup \{G_{ij} : 1 \leq i, j \leq n\} \cup \{C\}$,

b) $\Sigma = \{B_{ij}, W_{ij} : 1 \leq i, j \leq n\}$,

c) $\mu = [[[]_2]_1]_0$,

d) $M_0 = M_1 = \emptyset$, $M_2 = \{a_1\} \cup \{P_{ij} : (i = 0 \wedge 1 \leq j \leq n) \vee (i = n+1 \wedge 1 \leq j \leq n) \vee (j = 0 \wedge 1 \leq i \leq n) \vee (j = n+1 \wedge 1 \leq i \leq n)\}$,

e) $R_0 = \emptyset$, R_1 is similar to the set of rules R_1 in the previous family of system but exchanging the white pixels by black pixels and the other way round

1)
$$\begin{array}{ccccccc} & B & K & & & B & K \\ B & W & W & \rightarrow & B & B & W \\ & B & W & & & B & W \end{array}$$
where $K = B$ or $K = W$.

2)
$$\begin{array}{ccccccc} & B & B & & & B & B \\ B & W & W & \rightarrow & B & B & W \\ & B & B & & & B & B \end{array}$$
The two first types of rules eliminate the black pixels connected to other black pixel which is a ***cutting vertex*** of the G_W.

3)
$$\begin{array}{ccccccc} & B & & & & B & \\ B & W & W & \rightarrow & B & B & W \\ & W & W & & & W & W \end{array}$$

4)
$$\begin{array}{ccccccccc} & B & & & & & B & & \\ B & W & W & W & \rightarrow & B & B & W & W \\ & W & B & B & & & W & W & B \\ & & B & & & & & B & \end{array}$$

5)
$$\begin{array}{ccccccccc} B & B & B & & & B & B & B & \\ W & W & W & B & \rightarrow & W & B & B & B \\ W & B & W & & & W & W & W & \\ & B & & & & & B & & \end{array}$$

6)
$$\begin{array}{ccccccc} W & W & W & & W & W & W \\ W & B & W & \rightarrow & W & W & W \\ W & W & W & & W & W & W \end{array}$$
The rules of types from 3 to 6 are used to reduce the dimensions of the white connected components with white pixels inside them

7)
$$\begin{array}{ccccccccc} & B & B & & & & B & B & \\ B & W & W & B & \rightarrow & B & W & B & B \\ B & W & W & B & & B & B & B & B \\ & B & B & & & & B & B & \end{array}$$
The rules of types from 3 to 7 are used to reduce the dimensions of the white connected components with white pixels inside them

8)
$$\begin{array}{ccccccc} & B & & & & B & \\ B & W & B & \rightarrow & B & G & B \\ & B & & & & B & \end{array}$$

9) $G_{ij} \rightarrow B_{ij}(C, out)$ for $1 \leq i, j \leq n$. This first type of rules sends an object C to the environment and removes the object G_{ij} and adds the object B_{ij} to the membrane of the system.

R_2 is the following set of rules:

1)
$$a_i \rightarrow a_{i+1}$$
for $i = 1 \dots 5$. It is a counter used to decide when the membrane 2 is dissolved.

2)
$$PW \rightarrow PP$$
There exists 4 rules of this type depending the white pixel would be up, down, right or left of the pink pixel. Formally, one of these rules is written as follows:
$$P_{ij}W_{ij+1} \rightarrow P_{ij}P_{ij+1}$$
for $1 \leq i, j \leq n$.

3)
$$a_6 \rightarrow \delta$$
When the object a_6 appear in the membrane 2 this dissolved the membrane and all the objects pass to the membrane 1.

f) $\rho_0 = \rho_1 = \rho_2 = \emptyset$,
g) $i_\Pi = 2$,
h) $o_\Pi = 0$.

Overview of a computation:

We use a P system to compute H_1 of a digital image in a similar way to compute H_0. There exists a system of this family working in a parallel manner: First, it takes the white pixels not contained in black connected components and transforms these pixels in pink (second type of rules). When the counter a_i bring the object a_6 to the membrane 2 it is dissolved and all the elements in this membrane pass to the membrane 1. In this membrane system eliminates the branches of the white connected components that appear in the image in 4 steps (types of rules 1 and 2). Then, system reduces the size of the white connected components from four directions: up, down, left and right (rules of types 3 to 7) using a logarithmic number of steps respect to the size of the digital image and proportionate to the size of the biggest white connected component and reduce each component to only one white pixel. The rules of type 8 to identify the connected components and rules of type 9 to send one object C for each component to the environment. Then, the complexity of the problem to obtain homology group H_1 of binary 2D digital image using tissue-like P systems is $O(n)$.

Complexity and Necessary Resources:

Taking account the size of the input data is $O(n^2)$, the amount of necessary resources for defining the systems of our family and the complexity of our problem can be observed in the following table:

H_1 Problem	
Complexity	
Number of steps of a computation	$O(n)$
Necessary Resources	
Size of the alphabet	$3n^2 + 1$
Initial number of cells	1
Initial number of objects	$4n - 3$
Number of rules	$O(n^2)$
Upper bound for the length of the rules	24

C. Examples

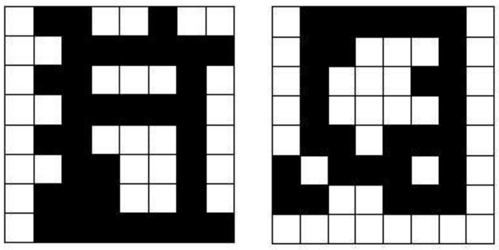
Fig. 3. Two input images to check

First, we are going to obtain the different connected components for the images given by Figure 3.

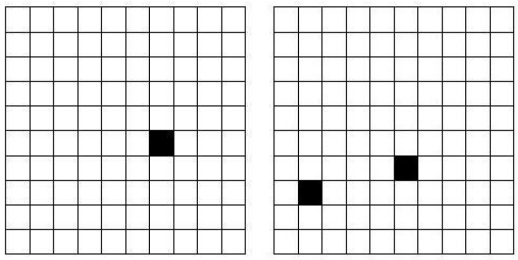
Fig. 4. Number of black connected components of the input images

After a logarithmic number of steps respect to the input data, the output data appears in the output cell of the system Π_1 (created to calculate the number of black connected components). This output is given using elements codifying the images are shown in the figure 4.

In the other way, using again a logarithmic number of steps respect to the input data, the number of white connected components inside of black connected components will appear in the output cell. It is shown in figure 5 the outputs codified by the elements that appear in the output cell for each one of images of figure 3.

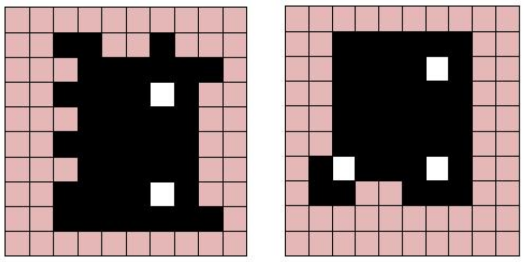

Fig. 5. Number of white holes of the input images

IV. Conclusions and Future Work

We have showed in this paper how we can efficiently obtain the homological groups to 2D images using P systems. The following steps to research are three: First, to use P systems to obtain more homological information: homology groups, spanning trees, homology gradient vector field... Until now, this homological information is 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 so important: 2D, 3D and 4D Image, Robotic,.... Second, to take software simulating P systems to check these systems and create new programs to obtain a better solution, because the actual simulators are learning software to help researchers to understand the models. Finally, other possible interesting research way from a computational point of view is to use some variants of P systems like Tissue P systems or Spiking Neural P systems to calculate homological information.

Acknowledgement

The first author acknowledge the support of the project "Computational Topology and Applied Mathematics" PAICYT research project FQM-296. The second author acknowledge the support of the project TIN2006-13425 of the Ministerio de Educación y Ciencia of Spain, cofinanced by FEDER funds, and the support of the Project of Excellence with *Investigador de Reconocida Valía* of the Junta de Andalucía, grant P08-TIC-04200. The third author acknowledge the support of the project MTM2006-03722 of the Ministerio español de Educación y Ciencia and the project PO6-TIC-02268 of Excellence of Junta de Andalucía.

References

[1] Adleman, L.M. Molecular computations of solutions to combinatorial problems. *Science*, **226** (1994), 1021–1024.

[2] Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K. and Walter, P. *Molecular Biology of The Cell*. Garland Science, 4th edition, (2002).

[3] Chao, J. and Nakayama, J. Cubical Singular Simples Model for 3D Objects and Fast Computation of Homology Groups. Proc. *ICPR'96, IEEE*, (1996), 190–194 .

[4] Díaz–Pernil D.:*Sistemas P de Tejido: Formalización y Eficiencia Computacional*. PhD Thesis, University of Seville (2008).

[5] González Díaz, R., Real, P.:Computation of Cohomology Operations of Finite Simplicial Complexes. *Homology Homotopy and Applications*, **2**, (2003), 83–93.

[6] Gonzalez-Diaz, R., Real, P.: Towards Digital Cohomology.*DGCI2003, LNCS*, **2886**, (2003), 92–101.

[7] Gonzalez-Diaz, R., Real, P.: On the cohomology of 3D digital images, *Discrete Applied Mathematics*, **147** (2005) 245-263.

[8] Gonzalez-Diaz, R., Medrano, B., Real, P., Sanchez-Pelaez, J.. Algebraic Topological Analysis of Time-sequence of Digital Images, LNCS, **3718**, (2005), 208-219.

[9] Gonzalez-Diaz, R., Medrano B., Real, P., Sánchez J.: Reusing Integer Homology Information of Binary Digital Images. *DGCI06, LNCS-4245*, (2006) 199–210.

[10] Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: A Fast P System for Finding a Balanced 2-Partition. *Soft Computing*, **9(9)**, (2005), 673–678.

[11] Gutiérrez–Naranjo, M.A., Pérez–Jiménez, M.J. and Romero–Campero, F.J. A linear solution for QSAT with Membrane Creation. *Lecture Notes in Computer Science*, **3850**, (2006), 241–252.

[12] A. Hatcher. Algebraic Topology. *Cambridge Univ. Press*, Cambridge, UK, (2001).

[13] Holland, J.H. Adaptation in Natural and Artificial Systems. Ann Arbor, *MI: University of Michigan Press*, (1975).

[14] McCulloch, W.S. and Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, **5**, (1943), 115–133.

[15] Martín–Vide, C., Pazos, J., Păun, Gh. and Rodríguez Patón, A. A New Class of Symbolic Abstract Neural Nets: Tissue P Systems. *Lecture Notes in Computer Science*, **2387**, (2002), 290–299.

[16] Martín–Vide, C., Pazos, J., Păun, Gh. and Rodríguez Patón, A. Tissue P systems. *Theoretical Computer Science*, **296**, (2003), 295–326.

[17] Molina-Abril, H., Real. P. Advanced Homological information on 3D Digital volumes, *SSPR 2008, LNCS*, **5342**, (2008), 361-371.

[18] Păun, Gh. Computing with membranes. *Journal of Computer and System Sciences*, **61**, 1, (2000), 108–143.

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

[20] Păun, A. and Păun, Gh. The power of communication: P systems with symport/antiport. *New Generation Computing*, **20**, 3, (2002), 295–305.

[21] Păun, Gh.: Computing with Membranes: Attacking **NP**–complete Problems. *In Unconventional Models of Computation, UMC'2K* (I. Antoniou, C. Calude, M.J. Dinneen, eds.), Springer–Verlag, (2000), 94–115.

[22] Păun, Gh. and Pérez–Jiménez, M.J. Recent computing models inspired from biology: DNA and membrane computing. *Theoria*, **18**, 46, (2003), 72–84.

[23] Păun, Gh., Pérez–Jiménez, M.J. and Riscos–Núñez, A. Tissue P System with cell division. *Second Brainstorming Week on Membrane Computing*, Sevilla, Report RGNC 01/2004, (2004), 380–386.

[24] Real, P. An Algorithm Computing Homotopy Groups. *Mathematics and Computers in Simulation*. **42**, n. 4-6, (1996), 461-465.

[25] Real, P. Homological Perturbation Theory and Associativity. *Homology Homotopy and Applications* (2000) 51-88.

[26] Real, P., Molina-Abril, H., Kropatsch, W. Homological tree-based strategies for image analysis. *Computer Analysis and Image Patterns, CAIP*, (2009).

[27] Sergeraert, F. The computability problem in algebraic topology. *Advances in Mathematics*, *104*, (1994), 1-29.

[28] The P Systems Website: `http://ppage.psystems.eu/`