Searching Partially Bounded Regions with P Systems

Article *in* Advances in Intelligent Systems and Computing · April 2014

DOI: 10.1007/978-81-322-1771-8_5

CITATIONS
2

READS
33

4 authors:

Hepzibah A. Christinal
Karunya University
26 PUBLICATIONS **163** CITATIONS
SEE PROFILE

Ainhoa Berciano
Universidad del País Vasco / Euskal Herriko Unibertsitatea
71 PUBLICATIONS **204** CITATIONS
SEE PROFILE

Daniel Díaz-Pernil
Universidad de Sevilla
59 PUBLICATIONS **737** CITATIONS
SEE PROFILE

Miguel A. Gutiérrez-Naranjo
Universidad de Sevilla
120 PUBLICATIONS **1,068** CITATIONS
SEE PROFILE

Searching Partially Bounded Regions with P Systems

Hepzibah A. Christinal[1,3], Ainhoa Berciano[1,2], Daniel Díaz-Pernil[1], and Miguel A. Gutiérrez-Naranjo[4]

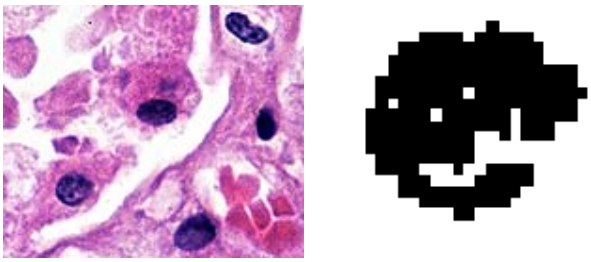
[1] CATAM Research Group - Dept. of Applied Mathematics I
University of Seville, Spain
[2] Departamento de Didáctica de la Matemática y de las Ciencias Experimentales
University of the Basque Country
[3] Karunya University, Coimbatore, Tamilnadu, India
[4] Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla, 41012, Spain
hepzi@yahoo.com, ainhoa.berciano@ehu.es, sbdani@us.es, magutier@us.es

Abstract. The problem of automatically marking the interior and exterior regions of a simple curve in a digital image becomes a hard task due to the noise and the intrinsic difficulties of the media where the image is taken. In this paper, we propose a definition of the *interior* of a partially bounded region and present a bio-inspired algorithm for finding it in the framework of Membrane Computing.
Keywords: Partially bounded region, Membrane Computing, Tissue P System

1 Introduction

In Mathematics, the apparently naive concepts of interior and exterior regions of a *simple closed curve* have a long list of approaches and further refinements. From the initial definition of a simple closed curve (or *Jordan curve*) in the plane $\mathbb{R}^2$ as the image of an injective continuous map of a circumference into the plane, $\phi : S_1 \to \mathbb{R}^2$ and the well-known *Jordan theorem* which claims that the complement of a Jordan curve in the plane has two connected components, many extensions, refinements and new proofs have been necessary [11]. Some of them were generalizations of higher dimensions, and some other due to *mathematical monsters* [5] as nowhere differentiable curves or Jordan curve of positive area.

In Digital Image Analysis, the problem of fixing the interior and exterior of a curve has new features. First of all, in Digital Image Analysis, the image is not infinitely divisible (as mathematical curves). The basic unit is the pixel and the concepts of connection of components, frontiers and bounds must be based on it. Such concepts have been widely studied in the literature. Nonetheless, the use of real-world images provides a new type of problem. The noise, the resolution of the picture or the intrinsic difficulties of the media where the image is taken



(a) (b)

Fig. 1. Images taken from a IgA nephropathy

are sources of misleading interpretations in the concepts of interior or exterior of a plain curve.

In this paper, we propose a definition and an algorithm for considering whether a black connected component (BCC) *surrounds*, in some sense, a portion of white region in a binary image. The practical application is immediate, since the BCC can be considered as an *imperfect closed curve*, and the surrounded region as an *imperfect topological hole* of the BCC. The definition of such *partially bounded regions* is a hard task even for human experts. A typical case study is a medical image where the expert must decide whether a set of pixels *enclose* a region or not. As an example, Figure 1 (a) shows the detail of an image associated with the IgA nephropathy, the most common glomerulonephritis throughout the world [1]. Figure 1 (b) shows an amplified region of the image after a binarization process and the algorithm presented in [4] for detecting connected components. Even for the human eye, it is difficult to decide whether the black pixels surrounds a white region. A computer assistant for such decision must consider that the hole can be *imperfectly surrounded* by 4-adjacent black pixels.

Next, a novel definition of *Partially Bounded Region* of an image and an algorithm for automatically finding them based on Membrane Computing[5] techniques are presented. As it will be shown below, such techniques are inspired in the flow of metabolites between cells of a living tissue or between the organelles in a eucaryotic cell. This flow of metabolites takes place in parallel in Nature and can be interpreted as a flow of information.

The paper is organized as follows: Firstly, we recall the computational bio-inspired used model used (Section 2). Next, our definition of *Partially Bounded Region* is presented, together with our Membrane Computing algorithm (Section 4). Finally, some conclusions are shown.

[5] We refer to [9] for basic information in this area, to [10] for a comprehensive presentation and the web site `http://ppage.psystems.eu` for the up-to-date information.

2 Membrane Computing

Membrane Computing [10] is a model of computation inspired by the structure and functioning of cells as living organisms able to process and generate information. The computational devices in Membrane Computing are called *P systems*. Roughly speaking, a P system consists of a membrane structure, in whose compartments one places multisets of objects which evolve according to given rules. These multisets encode the information and the rules deal with them performing the computation. Following a biological inspiration, the multisets of objects represent the chemicals placed in a vesicle of a living being. Such chemicals are sent to other vesicles or transformed according to biochemical reactions, represented here by computational rules. These rules are usually applied in a synchronous non-deterministic maximally parallel manner. In this paper, the so-called (because of their membrane structure) *tissue P Systems* [6] are considered.

2.1 Tissue P systems

The chosen P system model in this paper is the *tissue P systems model* which has been widely used to solve computational problems in other areas (see, e.g., [2,3]), but recently, they have been also used in the study of digital images (see, e.g., [4,7,8] and references therein).

Informally, a *tissue P system* of degree $q \geq 1$ can be seen as a set of q cells 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, i.e., as usual in tissue P systems, the edges linking cells are not provided explicitly: If a rule $(i, u/v, j)$ is given, then cells i and j are considered linked. The application of a rule $(i, u/v, j)$ consists of trading the multiset u (initially in the cell i) against the multiset v (initially in j). After the application of the rule, the multiset u disappears from the cell i and it appears in the cell j. Analogously, the multiset v disappears from the cell j and it appears in the cell i. In what follows we assume the reader is familiar with the basic notions and the terminology underlying P systems.

Formally, a *tissue P system* of degree $q \geq 1$ is a tuple of the form $\Pi = (\Gamma, \Sigma, \mathcal{E}, w_1, \ldots, w_q, \mathcal{R}, i_{in}, i_{out})$ where q is the number of cells and

1. Γ is a finite alphabet, whose symbols will be called objects. These objects can be placed in the cells or in the surrounding space (called the *environment*),
2. $\Sigma \subseteq \Gamma$ is the input alphabet. The *input* of the computation performed by the P system is encoded by using this alphabet,
3. $\mathcal{E} \subseteq \Gamma$ is a finite alphabet representing the set of the objects in the environment. Following a biological inspiration, the objects in the environment are available in an arbitrary large amount of copies;
4. $w_1, \ldots, w_q$ are strings over Γ representing the multisets of objects placed inside the cells at the starting of the computation;
5. $\mathcal{R}$ is a finite set of rules of type $(i, u/v, j)$ for $0 \leq i \neq j \leq q,\ u, v \in \Gamma^*$
6. $i_{in}, i_{out} \in \{1, 2, \ldots, q\}$ denotes the input cell and output cell, respectively.

The biological inspirations of this model are intercellular communication and cooperation between neurons. Rules are used as usual in the framework of membrane computing, that is, in a maximally parallel way (a universal clock is considered).

A *configuration* is an instantaneous description of the tissue 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 configuration is *halting* when no rules can be applied to it. A *computation* is a sequence of computation steps such that either it is infinite or it is finite and the last step yields a halting configuration (i.e., no rules can be applied to it). Then, a computation halts when the P system reaches a halting configuration. The output of a computation is the multiset placed in the output cell in a halting configuration collected from its halting configuration by reading the objects contained in the output cell.

2.2 Example

Let us consider the following tissue P system with cell division of degree 3, $\Pi = (\Gamma, \Sigma, \mathcal{E}, w_1, w_2, w_3, \mathcal{R}, i_\Pi, i_0)$, where $\Gamma = \{a_1, a_2, b, c, p, q, r, x\}$, $\Sigma = \{a_1, a_2\}$ and $\mathcal{E} = \{x\}$. The multisets in the initial configuration are $w_1 = b$, $w_2 = c$ and $w_3 = r$. The set of rules are $R_1 \equiv (1, a_1 b / x^3, 0)$, $R_2 \equiv (1, x/p, 2)$, $R_3 \equiv (1, x/q, 2)$, $R_4 \equiv (1, q/r, 3)$ and $R_5 \equiv [c]_2 \rightarrow [p]_2\,[q]_2$. Finally, the input cell is $i_\Pi = 1$ and the output cell is $i_0 = 3$. Notice that rules $R_1 \ldots, R_4$ determine a virtual graph with the cells as nodes. From R_2 and R_3 we can consider an edge between cell 1 and cell 2. Analogously, R_4 determines an edge between cell 1 and cell 3. We also know by rule R_1 that cell 1 can trade some objects with the environment.

Let us consider as input of our computation the multiset $a_1 a_2$ (one copy of a_1 and two copies of a_2) placed in the input cell 1. By considering the input, the initial configuration C_0 has three cells, labelled with 1,2,3 and with the multisets $w_1 = a_1 a_2^2 b$, $w_2 = c$ and $w_3 = r$. In the first step of computation, rules R_1 and R_5 are applied. Rule 1 interchanges the objects $a_1 b$ from cell 1 with three copies of x taken from the environment. Rule 5 divides the cell 2. Hence, the configuration C_1 has four cells: two of them labelled with 1 and 3 (respectively) and the other two cells have the label 2. The multiset in the cell labelled by 1 is $w_1 = x^3 a_2$, the cell labelled by 3 contains the multiset $w_3 = r$ and the cells labelled by 2 have, respectively, the multisets $w_2 = p$ and $w_2 = q$. In the following step of computation rules R_2 and R_3 are applied. These rules send one copy of the object x to the corresponding cell labelled by 2 against one copy of p and q (respectively). Therefore, the configuration C_2 has the same four cells as in the configuration C_1, but with the multisets $w_1 = x a_2 pq$, $w_3 = r$ and the cells labelled by 2 have the same multiset $w_2 = x$. Finally, in the third computation step the rule R_4 is applied and the object q in the cell 1 is interchanged with the object r in the cell 3. We get the final configuration C_3 with $w_1 = x a_2^2 pr$, $w_3 = q$ and $w_2 = x$ in both cells labelled by 2. As the output cell is cell 3, the multiset r placed in this cell in the last configuration is the output of the configuration.

3 Digital Imagery

A *point set* is simply a topological space consisting of a collection of objects called points and a topology which provides for such notions as *nearness* of two points, the *connectivity* of a subset of the point set, the *neighborhood* of a point, *boundary points*, and *curves* and *arcs*. For a point set X in Z, a *neighborhood function* from X in Z, is a function $N : X \to 2^Z$. For each point $x \in X$, $N(x) \subseteq Z$. The set $N(x)$ is called a *neighborhood* for x.

There are two neighborhood function on subsets of $\mathbf{Z}^2$ which are of particular importance in image processing, the *von Neumann* neighborhood and the *Moore* neighborhood. The first one $N : X \to 2^{\mathbf{Z}^2}$ is defined by $N(x) = \{y : y = (x_1 \pm j, x_2)$ *or* $y = (x_1, x_2 \pm k),\ j, k \in \{0, 1\}\}$, where $x = (x_1, x_2) \in X \subset \mathbf{Z}^2$. While the Moore neighborhood $M : X \to 2^{\mathbf{Z}^2}$ is defined by $M(x) = \{y : y = (x_1 \pm j, x_2 \pm k),\ j, k \in \{0, 1\}\}$, where $x = (x_1, x_2) \in X \subset \mathbf{Z}^2$. The von Neumann and Moore neighborhood are also called the *four neighborhood* (4-adjacency) and *eight neighborhood* (8-adjacency), respectively.

An *Z-valued image* on X is any element of Z^X. Given an Z-valued image $I \in Z^X$, i.e. $I : X \to Z$, then Z is called the set of possible range values of I and X the spatial domain of I. The graph of an image is also referred to as the *data structure representation* of the image. Given the data structure representation $I = \{(x, I(x)) : x \in X\}$, then an element $(x, I(x))$ is called a *picture element* or *pixel*. The first coordinate x of a pixel is called the *pixel location* or *image point*, and the second coordinate $I(x)$ is called the *pixel value* of I at location x.

For example, X could be a subset of $\mathbf{Z}^2$ where $x = (i, j)$ denotes spatial location, and Z could be a subset of $\mathbf{N}$ or $\mathbf{N}^3$, etc. So, given an image $I \in Z^{\mathbf{Z}^2}$, a pixel of I is the form $((i, j), I(x))$, which be denoted by $I(x)_{ij}$. We call the *set of colors* or *alphabet of colors of* I, $\mathcal{C}_I \subseteq Z$, to the image set of the function I with domain X and the image point of each pixel is called *associated color*. We can consider an order in this set. Usually, we consider in digital image a predefined alphabet of colors $\mathcal{C} \subseteq Z$. We define $h = |\mathcal{C}|$ as the size (number of colors) of $\mathcal{C}$. In this paper, we work with images in grey scale, then $\mathcal{C} = \{0, \ldots, 255\}$, where 0 codify the black color and 255 the white color.

A *region* could be defined by a subset of the domain of I whose points are all mapped to the same (or similar) pixel value by I. So, we can consider the region R_i as the set $\{x \in X : I(x) = i\}$ but we prefer to consider a region r as a maximal connected subset of a set like R_i. We say two regions r_1, r_2 are adjacent when at less a pair of pixel $x_1 \in r_1$ and $x_2 \in r_2$ are adjacent. We say x_1 and x_2 are *border pixels*. If $I(x_1) < I(x_2)$ we say x_1 is an *edge pixel*. The set of connected edge pixels with the same pixel value is called a *boundary* between two regions.

4 Partially Bounded Regions

Given a binary 2D digital image, the HGB2I problem consists on calculating the number of connected components and the representative curves of the holes

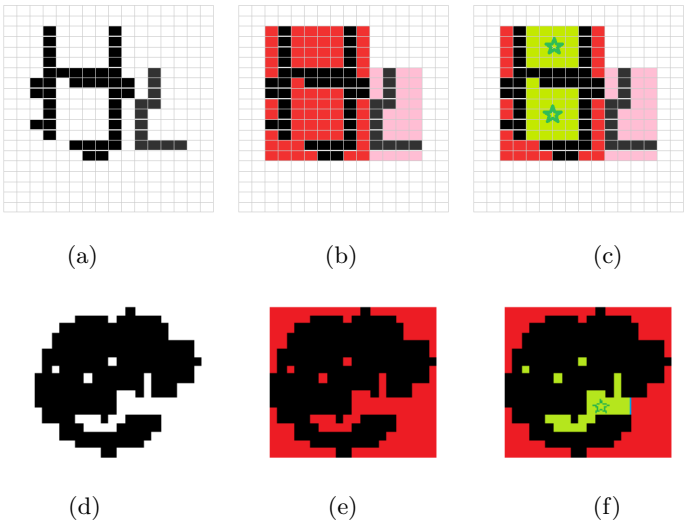


Fig. 2. (a) and (d) Input images; (b) and (e) Minimum regions containing a BCC. We can see two labels (colors), one of each connected component (c) and (f) Candidate regions to PBRs (marked with an star).

of these components. We can divide this problem in two sub-problems: the H_0 *problem* consists on calculating the number of connected components and the H_1 *problem*, which consists on calculating the number of holes. In [4], the HGB2I problem was solved by using tissue P systems. In this paper, we extend the design of such P systems in order to find *Partially Bounded Regions* (PBR).

Definition 1. *Let I be a black and white image and C_1 a connected component of black pixels of I. Let P be a white pixel of the image I*

- *We will say that C_1 bounds the pixel P on the north (resp. south, east and west) if there is at least one black pixel of the image I on the north (resp. south, east and west) of P and the first of such black pixels belongs to C_1.*
- *We will say that the region of white pixels C_2 is partially bound by C_1 if it is connected, all its pixels are bounded by C_1 at least on three sides (from north, south, east or west) and it is maximal, i.e., it is not a proper subset of a region with these properties.*

4.1 Looking for Partially Bounded Regions

Given an input image, the stages of our algorithm for automatically finding such regions are: *Stage 1:* Distinguishing BCCs; Stage 2: Minimal region containing a BCCs; *Stage 3:* Marking candidates as possible PBRs; and *Stage 4:* Choosing PBRs.

Stage 1: *Distinguishing BCCs.* The first step for marking the PBR is to identify the different BCCs of the image. The algorithm from [4] to solve H_0 problem assigns a different label to each of them. So, up to now, we consider we have applied the previous algorithm, i.e., we have all the BCC labeled, and each component with a different label.

Stages 2 and 3: *Minimal region containing a BCCs.* The minimal rectangle containing the connected component $\mathbb{C}$ is found. All the white pixels in such rectangle receive the same label as the black pixels of the BCC. (Rules R_1 to R_2 in the tissue P System described below.)

Marking candidates as possible PBRs. Each white pixel inside of a minimal rectangle is associated to a label. We associate the label N to this pixel if there is a black or a white pixel with this label above it, the label S if there is a black or a white pixel with this label below it; and we follow exactly the same way for the labels E (right) and O (left). (Rules R_3 to R_{10} in the tissue P system). We keep the pixels with at least the three labels from $\{N, S, E, O\}$ (Rule R_{11}).

We define a family of tissue P system to develop the steps considered in the stages 2 and 3. For each image $\mathcal{I}$ with size n^2, we define a tissue P system, $\Pi(n) = (\Gamma, \Sigma, \mathcal{E}, w_1, \mathcal{R}, i_{in}, i_{out})$ where:

- $\Gamma = \{w_{ij} : 1 \leq i, j \leq n\} \cup \{(a_{ij}, (k, l)) : a \in \{b, r, w\} \wedge 1 \leq i, j, k, l \leq n\} \cup \{N, S, O, E\}$, $\Sigma = \{(a_{ij}, (k, l)) : a \in \{b, w\} \wedge 1 \leq i, j, k, l \leq n\}$, $\mathcal{E} = \{(a_{ij}, (k, l)) : a \in \{b, r, w\} \wedge 1 \leq i, j, k, l \leq n\}$,
- $w_1 = \emptyset$,
- $i_{in} = i_{out} = 1$ is the output cell.

$\mathcal{R}$ is the following set of rules:

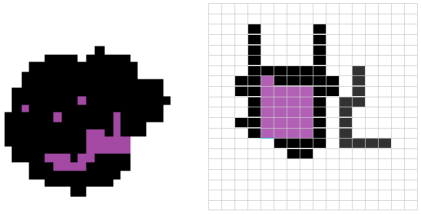
- First, we will paint the white pixels adjacent to black pixels, maintained the label:
- $R_1 \equiv (1, (w_{i'j'})(b_{ij}, (k, l))/(r_{i'j'}, (k, l))(b_{ij}, (k, l)), 0)$ for (i, j), (i', j') adjacent positions in the image.
- Now, we will find the corners of our new red line and we will expand this connected component using these corners. In this way, the red component will be expanded if and only if it is concave, so it will grow in the interior part of the black connected components. Anyway, it will be able to grow on the exterior part too, but it will stop (see Figure 2 (b)), because of it will not be bigger than the minimal rectangle that contains this connected component.
- $R_2 \equiv \begin{matrix}(1, & (x,(k,l))\ (y,(k,l)) \\ & (z,(k,l)) \quad\ w\end{matrix} \quad / \quad \begin{matrix}(x,(k,l))\ (y,(k,l)) \\ (z,(k,l))\ (r,(k,l))\ , 0)\end{matrix}$ where $x, y, z \in \{b, r\}$
- Once any rule of type R_2 cannot be activated, we will start working with the colored pixels with the same label as our connected component. The important key is to transfer to each red pixel in which direction it has black pixels with the same label.We will preserve the red pixels which has black pixels (with the same label) in the four directions. To do this, we will work with the following rules:

- $R_3 \equiv (1,(b_{ij},(k,l))(r_{ij+1},(k,l))/(b_{ij},(k,l))(r_{ij+1},(k,l),E),0)$
  $R_4 \equiv (1,(r_{ij},(k,l),u)(r_{ij+1},(k,l),v)/(r_{ij},(k,l),u)(r_{ij+1},(k,l),v\,E),0)$ for $u,v \in \{N,S,E,O\}^*$, $E \in u$ and $E \notin v$. In this case it is only possible to transfer from left to right. That is, if a pixel adjacent to the given one, that is not on the left of the first one, has the label E, this rule will not be executed.
- $R_5 \equiv (1,(b_{ij},(k,l))(r_{ij-1},(k,l))/(b_{ij},(k,l))(r_{ij-1},(k,l),O),0)$
  $R_6 \equiv (1,(r_{ij},(k,l),u)(r_{ij-1},(k,l),v)/(r_{ij},(k,l),u)(r_{ij-1},(k,l),v\,O),0)$ for $u \in \{N,S,E,O\}^*$ and $O \in u$ and $O \notin v$. In this case it is only possible to transfer from right to left. That is, if a pixel adjacent to the given one, that is not on the right of the first one, has the label O, this rule will not be executed.
- $R_7 \equiv (1,(b_{ij},(k,l))(r_{i+1j},(k,l))/(b_{ij},(k,l))(r_{i+1j},(k,l),S),0)$
  $R_8 \equiv (1,(r_{ij},(k,l),u)(r_{i+1j},(k,l),v)/(r_{ij},(k,l),u)(r_{i+1j},(k,l),v\,S),0)$ for $u \in \{N,S,E,O\}^*$ and $S \in u$ and $S \notin v$. In this case, the unique possibility is to transfer from down to up. That is, if a pixel adjacent to the given one, that is not at the bottom of the first one, has the label S, this rule will not be executed.
- $R_9 \equiv (1,(b_{ij},(k,l))(r_{i-1j},(k,l))/(b_{ij},(k,l))(r_{i-1j},(k,l),N),0)$
  $R_{10} \equiv (1,(r_{ij},(k,l),u)(r_{i-1j},(k,l),v)/(r_{ij},(k,l),u)(r_{i-1j},(k,l),v\,N),0)$ for $u \in \{N,S,E,O\}^*$ and $N \in u$ and $N \notin v$. In this case, it is only possible to transfer from up to down. That is, if a pixel adjacent to the given one, that is not at the top of the first one, has the label N, this rule will not be executed.
- Just the previous rules conclude, the colored pixels that have black pixels (with the same label) in their four directions, will be assigned a new label by rule R_{11}.
- $R_{11} \equiv (1,(r_{ij},(k,l),X\,Y\,Z)/(g_{ij},(k,l)),0))$ for $X \neq Y \neq Z$ and $X,Y,Z \in \{N,S,E,O\}$

Stage 4: *Choosing PBRs.* We take the pixels belonging to a candidate region and adjacent to a white pixel in the input image. If their number is lower than a threshold then this region is a PBR, but if number is greater than this threshold then it is rejected. This threshold is taken in function of the maximal distance between two pixels of the minimal region containing the BCCs. The definition of a family of tissue P systems to develop this stage is extremely easy. We can see a similar family in [7]. We can see in Figure 3 the results of our algorithm when it is applied in the images of Figure 2 (a) and (b).

5 Conclusions

Biological and medical images are intrinsically full of imperfect data. Even with the current technology, the noise cannot be fully avoided. The resolution of the camera, the absence of the proper light or an aqueous media can affect the accuracy of the image. In some technical areas, it is crucial to determine the interior of a connected set of pixels, even in case of it is not limited by a clear frontier. Solving such problem is a hard task even for expert human eyes.



(a) (b)

Fig. 3. (a) and (b) PBRs of the images in Figure 2 (a) and 2 (d), respectively.

In this paper, we propose a definition of partially bounded region which can help in this task. Finding a definition which cover each possible case is not possible, since the number of different cases from real-life images goes beyond any definition, but the proposed one can help in a vast amount of cases.

In this paper, we also use techniques from the bio-inspired paradigm of Membrane Computing. This new computational paradigm have features which makes it suitable for dealing with digital images and it has been successfully applied recently to Digital Image Analysis. For example, the subset of the integer plane or space taken to be the *support* of the image and the set of possible features associated to each 2D point can be considered finite and hence, the transformation of an image into another can be performed in a *discrete* way. Other of such features is that the treatment of the image can be parallelized and locally modified. Regardless how large is the picture, the process can be performed in parallel in different local areas of it. Another interesting feature is that the information of the image can also be easily encoded in the data structures used in Membrane Computing.

Acknowledgements

MAGN acknowledges the support of the project TIN2012-37434 of the Ministerio de Economía y Competitividad of Spain.

References

1. D'amico, G.: The commonest glomerulonephritis in the world: Iga nephropathy. QJM: An International Journal of Medicine 64(3), 709–727 (1987)
2. Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: Solving the independent set problem by using tissue-like P systems with cell division. In: J. Mira Mira *et al.* (ed.) IWINAC (1). Lecture Notes in Computer Science, vol. 5601, pp. 213–222. Springer (2009)
3. Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: A Linear Time Solution to the Partition Problem in a Cellular Tissue-Like Model. Journal of Computational and Theoretical Nanoscience 7(5, SI), 884–889 (2010)

4. Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Real, P., Sánchez-Canales, V.: Computing homology groups in binary 2D imagery by tissue-like P systems. Romanian Journal of Information Science and Technology 13(2), 141–152 (2010)
5. Feferman, S.: Mathematical Intuition vs. Mathematical Monsters. Synthese 125(3), 317–332 (2000)
6. Martín-Vide, C., Păun, G., Pazos, J., Rodríguez-Patón, A.: Tissue P systems. Theoretical Computer Science 296(2), 295–326 (2003)
7. Peña-Cantillana, F., Díaz-Pernil, D., Berciano, A., Gutiérrez-Naranjo, M.A.: A parallel implementation of the thresholding problem by using tissue-like P systems. In: P. Real *et al.* (ed.) CAIP (2). Lecture Notes in Computer Science, vol. 6855, pp. 277–284. Springer (2011)
8. Peña-Cantillana, F., Díaz-Pernil, D., Christinal, H.A., Gutiérrez-Naranjo, M.A.: Implementation on CUDA of the smoothing problem with tissue-like P systems. International Journal of Natural Computing Research 2(3), 25–34 (2011)
9. Păun, G.: Membrane Computing. An Introduction. Springer-Verlag (2002)
10. Păun, G., Rozenberg, G., Salomaa, A. (eds.): The Oxford Handbook of Membrane Computing. Oxford University Press, Oxford, England (2010)
11. Ross, F., Ross, W.T.: The Jordan curve theorem is non-trivial. Journal of Mathematics and The Arts 5, 213–219 (2011)