Membrane Computing for Real Medical Image Segmentation

Rafaa I. Yahya[1,2], Siti Mariyam Shamsuddin[2,3], Salah I. Yahya[4,5], Bisan Alsalibi[2,6], Ghada Al-Khafaji[2,7]

[1]Department of Computer, College of Science, University of Al-Mustansiriyah, Baghdad, Iraq

[2]UTM Big Data Center, Ibnu Sina Institute for Scientific and Industrial Research, Universiti Teknologi Malaysia, UTM Skudai, 81310 Johor, Malaysia

[3]Faculty of Computing, Universiti Teknologi Malaysia, UTM Skudai, 81310 Johor, Malaysia

[4]Department of Software Engineering, Faculty of Engineering, Koya University, University Park, Danielle Mitterrand Boulevard, Koya KOY45, Kurdistan Region, Iraq

[5]Department of Computer Science and Engineering, School of Science and Engineering, University of Kurdistan Hewler, Kurdistan region, Iraq

[6]School of Computer Science, Universiti Sains Malaysia, Malaysia

[7]Department of Computer Science, College of Science, University of Baghdad, Iraq

Abstract—In this paper, membrane-based computing image segmentation, both region based and edge based, is proposed for medical images that involve two types of neighborhood relations between pixels. These neighborhood relations - namely 4-adjacency and 8-adjacency of a membrane computing approach - construct a family of tissue-like P systems for segmenting actual two-dimensional (2D) medical images in a constant number of steps; the two types of adjacency were compared using different hardware platforms. The process involves the generation of membrane-based segmentation rules for 2D medical images. The rules are written in the P-Lingua format and appended to the input image for visualization. The findings show that the neighborhood relations between pixels of 8-adjacency give better results compared with the 4-adjacency neighborhood relations because the 8-adjacency considers the eight pixels around the center pixel, which reduces the required communication rules to obtain the final segmentation results. The experimental results proved that the proposed approach has superior results in terms of the number of computational steps and processing time. To the best of our knowledge, this is the 1st time an evaluation procedure is conducted to evaluate the efficiency of real image segmentations using membrane computing.

Index Terms—Edge-based segmentation, Medical images, membrane computing, P-Lingua, Region-based segmentation, tissue-like P system.

I. INTRODUCTION

Membrane computing (MC) is a fascinating and fast-growing area of research that takes its inspiration from the subdivisions of biological cells into compartments delimited by membranes. The computational models in MC, which are a new class of distributed and parallel systems, are known as P systems in honor of their initiator Paun (2000, 2002). The distinguishing hallmarks of P systems are the intrinsic parallelism, the locality of interactions, and the capacity of generating new cells in linear time. The basic ingredients of a membrane system consist of (1) the membrane structure and delimiting compartments, which either correspond to a tree-like hierarchical arrangement of membranes as in a cell or a net of membranes represented by a directed graph as in a tissue. In each membrane, (2) multisets of objects are placed in the compartments, which evolve according to specific and (3) evolution rules. Typically, the external membrane is called the skin membrane, which contains several internal membranes, of which the elementary membrane is the one without any other membranes inside it (Paun and Rozenberg, 2000). Several variants of P systems have been proposed: Cell-like P systems (inspired from the structure of the cell) (Paun, 2000), tissue-like P systems (inspired from the organization of cells in tissue) (Martı, et al., 2003), and spiking neural-like membrane systems (inspired from the way neurons are linked in neural nets) (Ionescu, et al., 2006). The tissue-like P system, one of the most widely investigated P systems, is inspired by the intercellular communication and cooperation between neurons. Its mathematical model is basically a network of processors working with objects and communicating these objects through communication channels specified in advance. From the computational point of view, the essential feature of this P system is that

membranes do not have electrical charges as in cell-like P systems (Paun, 2000). Most recently, it has been proved that MC has strong potential to be applied to different problems related to biology as well as to linguistics, computer graphics, cryptography, and image processing, to name a few.

Image segmentation is one of the common tools in the image processing spectrum. It partitions the digital image into several regions (sets of pixels) and assigns a specific label to every pixel such that pixels with the same labels typically share similar visual characteristics. Interestingly, segmentation of medical images is a significant step before computer-aided diagnosis, because it is essential for further medical image processing, such as cancer detection. The image segmentation process has different approaches and mechanisms such as thresholding, boundary tracking, clustering, and region-based mechanisms. The result of image segmentation is a set of segments that collectively cover the entire image or a set of contours extracted from the image. Each of the pixels in a region is similar with respect to a certain characteristic or computed property, such as color, intensity, or texture (Shapiro and Stockman, 2001). In recent years, substantial effort has been invested into the problem of medical image segmentation, and several approaches have been proposed: For instance, the probability density-based segmentation approach, histogram-based approaches, the region-based method, and fuzzy clustering.

This paper proposes membrane-based computing image segmentations, both region based and edge based, for medical images that involve two types of neighborhood relations between pixels. Superior results in terms of the number of computational steps and processing time have been achieved using real medical images of different formats.

The paper is organized as follows: Section 2 investigates the related works. Section 3 presents the formal framework of tissue-like P systems. Section 4 explains the proposed segmentation methods for real medical images. Section 5 evaluates the proposed approach. Section 6 analyzes the experiment results. Section 7 concludes the paper.

II. Related Works

MC possesses several interesting features including the encapsulation of data, a trivial way of representing information as well as achieving maximal parallelism, all of which are most proper when handling digital images. A new research line has been recently launched in which MC has been adapted to solve several problems related to digital imagery. For example, image segmentation was investigated in Christinal, et al. (2009; 2010; 2011). However, in Christinal et al's. work, they used artificial images rather than real images for testing. Furthermore, other problems related to digital images, such as homology (Diaz-Pernil, et al., 2010; Alsalibi, et al., 2014; and Christinal, et al., 2010) and smoothing (Pena-Cantillana, et al., 2011), have been proposed within the framework of MC. With regard to segmentation, the authors in Christinal, et al. (2009) have proposed a family of tissue-like P systems using communication and evolution rules to perform edge-based segmentation of a two-

dimensional (2D) image by employing 4-adjacency. Their results have been achieved in a constant number of steps in which the system was tested by the tissue simulator to check its validity. Christinal, et al. (2010) designed an MC approach to solve the thresholding problem using cell-like P system rules in which the massive parallelism feature of MC helps reach the solution in linear time according to the size of the input image. Carnero, et al. (2010) designed a new hardware tool - namely a field-programmable gate array unit (FPGA) - to conduct segmentation of digital images to address the problems of edge-based detection and noise removal. From a different angle, Reina-Molina, et al. (2010) proposed a new version of the tissue-like P system by replacing the concept of one cell with the use of multiple auxiliary cells to address segmentation problem and exploit all potential parallelization. Similarly, the authors in Díaz-Pernil, et al. (2010) proposed a new software tool for segmenting 2D digital images based on the tissue-like P system, wherein the object-oriented C++ programming language was used in the implementation part. However, they did not provide a clear explanation regarding the technical aspects of developing the proposed tool. Christinal, et al. (2011) introduced a tissue-like P system (using communication rules) to the problem of region-based segmentation. In their work, a 4-adjacency relationship between pixels was adapted to segment 2D digital images, whereas 6-adjacency was used for 3D digital images. The researchers in Peña-Cantillana, et al. (2011) proposed bioinspired MC software program to solve the thresholding problem, and it has been developed by an innovative device architecture called the Compute Unified Device Architecture (CUDA). The researchers of Carnero, et al. (2011) proposed FPGAs to implement tissue-like P system rules and solve segmentation problems. In the work of Sheeba, et al. (2011), the authors constructed a family of tissue-like P system to segment medical images (nuclei of white blood cells) of the peripheral blood smear images in a morphology segmentation technique, and their algorithm was implemented using MATLAB software.

The authors in Díaz-Pernil, et al. (2012) designed a new software tool to segment images, in which a tissue-like P system was used to solve the problem in constant time due to the intrinsic parallelism of MC computational models. In Zhang and Peng (2012), a novel infrared object segmentation based on a thresholding method was proposed using the cell-like P system to obtain the best set of parameters quickly. In Christinal, et al. (2012), a variant of the P system (tissue-like P system) was constructed using the rules to perform a parallel color segmentation of 2D images based on a thresholding method. The authors in Peng, et al. (2012) proposed a thresholding segmentation method based on cell-like-based membrane algorithm to improve the performance of the threshold segmentation. In the same manner, the authors of Yang, et al. (2013) proposed an image segmentation technique by the use of a tissue-like P system to perform conventional region-based image segmentation. In the work of Díaz-Pernil, et al. (2013), a novel device architecture called CUDA was proposed to implement tissue-like P system rules for segmenting images by the use of gradient-based edge

detection to enhance the classical methods of segmentation. In Peng, et al. (2014), the authors proposed novel segmentation by enhancing the conventional region-based color image segmentation method using tissue-like P systems. In the work of Isawasan, et al. (2014), tissue-like P system rules were used to perform region-based segmentation of 2D hexagonal images in which the segmentation was performed in seven steps. However, they did not illustrate how they used P-Lingua to perform the segmentation. In the work of Peng, et al. (2015), the authors presented a new method using a cell-like P system (membrane algorithm) to solve the optimal multilevel thresholding problem. Yahya, et al. (2015) proposed a classical region-based segmentation with tissue-like P system rules. In their proposed work, a simple artificial image was used to provide a detailed illustration of the basic idea of how the P system works. Furthermore, various color relations have been investigated to illustrate the effect of colors on the segmentation results. Yahya, et al. (2016) proposed a tissue-like P system, where variant of MC was used to segment 2D hexagonal artificial images by employing edge-based segmentation and region-based segmentation. P-Lingua programming language was used to implement and validate the proposed P system Yahya, et al. (2017).

III. Definition of Tissue-Like P Systems

Tissue-like P systems were first introduced in Martı, et al. (2003) and Martin-Vide, et al. (2002). In tissue-like P systems, the membrane structure is a general directed graph. The edges of such graphs are not given explicitly, but they are deduced from the set of rules. The biological inspiration behind this model is 2-fold: Intercellular communication and cooperation between neurons. From the computational perspective, the essential feature of this P system is that membranes do not have electrical charges as in the cell-like P systems (Diaz-Pernil, et al., 2010).

Basically, a tissue-like P system is a tuple of degree $q \geq 1$:

$$\Pi = (\Gamma, \Sigma, \varepsilon, w_1, \ldots, w_q, R, i_\Pi, o_\Pi)$$

Where,

a. Γ is the finite alphabet of objects, including two distinguished objects yes and no, occurring in at least one copy in some initial multisets M_i but not in ε;

b. $\Sigma (\subset \Gamma)$ is the input alphabet;

c. $\varepsilon \subseteq \Gamma$ is the list of objects in the environment, each one of arbitrarily infinite copies;

d. $w_1, \ldots, w_q$ are strings over Γ representing the multisets of objects associated with the cells at the initial configuration;

e. R is a finite set of rules of the form.
   1. Communication rule: $(i, u/v, j)$, for $i, j \in \{0, 1, 2, \ldots, q\}$, $i \neq j$, $u, v \in \Gamma$;
   2. Division rules: $[a]_i \rightarrow [b]_i [c]_i$, where $i \in \{1, 2, \ldots, q\}$ and $a, b, c \in \Gamma$;

f. $i_\pi \in \{1, 2, \ldots, q\}$ is the input membrane;

g. $o_\pi \in \{0, 1, 2, \ldots, q\}$ is the output membrane.

In the typical framework of MC, each cell is viewed as a computing unit working in a maximally parallel and non-deterministic manner. The configuration is an instantaneous description of the P system at a particular time, in which a sequence of computation steps can be applied in a parallel way to obtain a new configuration. A computation is said to be successful if it halts, reaching a specific configuration in which no more rules can be further applied to the current objects. With a halting computation, the associated output can be codified by the content of the output membrane. A more detailed description of tissue-like P systems can be found in Diaz-Pernil, et al. (2010) and Martin-Vide, et al. (2002).

IV. Methods

A. Tissue-like P system for real medical image segmentation

We define a family of tissue-like P systems as follows:

$$\Pi(n,m) = (\Gamma(n,m), \Sigma, \varepsilon(n), \mu, M_1, M_2, M_3, R, i_1, o_2)$$

Where,

- The working alphabet $\Gamma(n, m)$ is the set $\{B_{ij}, W_{ij}, Wx_{ij}: 1 \leq i \leq n, 1 \leq j \leq m\} \cup \{Z_i : 1 \leq i \leq 9\}$;
- The initial membrane structure is $\mu = [[\,]1\,[\,]2]0$;
- The input alphabet $\Sigma = \{B_{ij}, W_{ij}: 1 \leq i \leq n, 1 \leq j \leq m\}$;
- The environment $\varepsilon = \Pi(n,m) - \Sigma$;
- Initial multisets: $M_1 = M_2 = \varphi$;
- i_1, o_2 are the input and the output membranes, respectively;
- R is the set of communication rules as follows:

$[z\{i\}]'1 \longleftrightarrow [z\{i+1\}*2]'0: 1 \leq i \leq 8;$
$[W\{i,j\}, B\{i,j+1\}]'1 \longleftrightarrow [Wx\{i,j\}, B\{i,j+1\}]'0: 1 \leq i \leq n, 1 \leq j \leq m;$
$[W\{i,j\}, B\{i,j-1\}]'1 \longleftrightarrow [Wx\{i,j\}, B\{i,j-1\}]'0: 1 \leq i \leq n, 1 \leq j \leq m;$
$[W\{i,j\}, B\{i+1,j\}]'1 \longleftrightarrow [Wx\{i,j\}, B\{i+1,j\}]'0: 1 \leq i \leq n, 1 \leq j \leq m;$
$[W\{i,j\}, B\{i-1,j\}]'1 \longleftrightarrow [Wx\{i,j\}, B\{i-1,j\}]'0: 1 \leq i \leq n, 1 \leq j \leq m;$

The previously mentioned communication rules are used when the image has edge pixels (two adjacent pixels with different associated colors). In this case, the pixel with less associated color (white pixel) will be marked, and the system brings from the environment an object representing this marked edge pixel (Wx). However, the next set of rules will be used to mark with a bar all pixels that are adjacent to two pixels of the same color which were marked before, but with the condition that the marked objects are adjacent to another pixel with a different associated color. Those rules are as follows:

$[Wx\{i,j\}, W\{i,j+1\}, Wx\{i+1,j+1\}, B\{i+1,j\}]'1 \longleftrightarrow [Wx\{i,j\}, Wx\{i,j+1\}, Wx\{i+1,j+1\}, B\{i+1,j\}]'0;$
$[Wx\{i,j\}, W\{i-1,j\}, Wx\{i-1,j+1\}, B\{i,j+1\}]'1 \longleftrightarrow [Wx\{i,j\}, Wx\{i-1,j\}, Wx\{i-1,j+1\}, B\{i,j+1\}]'0;$
$[Wx\{i,j\}, W\{i,j+1\}, Wx\{i-1,j+1\}, B\{i-1,j\}]'1 \longleftrightarrow [Wx\{i,j\}, Wx\{i,j+1\}, Wx\{i-1,j+1\}, B\{i-1,j\}]'0;$
$[Wx\{i,j\}, W\{i+1,j\}, Wx\{i+1,j+1\}, B\{i,j+1\}]'1 \longleftrightarrow [Wx\{i,j\}, Wx\{i+1,j\}, Wx\{i+1,j+1\}, B\{i,j+1\}]'0;$
$[z\{9\}, Wx\{i,j\}]'1 \longleftrightarrow [\#]'2: 1 \leq i \leq n, 1 \leq j \leq m;$

B. Segmentation with a Tissue-Like P System

To work with real medical images, the P-Lingua programming language, including PLinguaCore4, is used in this work. The P-Lingua programming language pioneered the standardization of P systems by providing an open source,

plugin-based software architecture meant for its extension by interested developers. P-Lingua is a software tool that provides a speciation language in which designers can define and describe P systems. Furthermore, it provides a set of Java simulators such that users can select the simulator from those included that best suits their requirements. P-Lingua consists of Java standalone software, an application program interface (API) - namely PLinguaCore4 - which performs two operations:

1. Simulate a P system and display the results and
2. Translate a P system between two (presumably different) formats.

On a PLinguaCore4 simulation, the user specifies the location of the input, the format in which this input is encoded and the simulation algorithm. The API then reads through these files looking for the implementation of both the format parser and the simulation algorithm. First, it identifies the P system type defined in the input and checks the existence of the definition in its XML files for this type. If the definition does not exist, PLinguaCore outputs a failure message and halts. Otherwise, the API reads the input file. If this file contains any errors or does not comply with the restrictions defined for its type, the execution halts, and error messages are output.

This information is intended to serve as a guide for debugging the P system. On the other hand, if the file specification correctly describes a P system, then PLinguaCore4 simulates its P system according to the parameters above. The workflow of PLinguaCore4 is shown in Fig. 1.

More precisely, based on Fig. 1, the P-LinguaCore4 API reads the input file to check for syntax errors using the compiler. If there are no errors, a binary file will be generated, which, in turn, will be simulated to produce the final output file. The methodology of the proposed approach is shown in Fig. 2 in which P-Lingua is linked to C# to perform automatic segmentation of real images according to the detail process of each step described in the next subsection. Fig. 3 presents the algorithm of image segmentation.

Loading image

In the first step of the proposed approach, the input image will be loaded into the system using the load button.

The system accepts different image formats, so the image extension can be any of the most common raster image formats (.jpg.,png.,gif, etc.). Once the image is loaded into the system, it will be resized to a specific size for scalable computation. As shown in this work, real medical images were used to test the system; a skin cancer bmp-type image (1280 px × 1024 px), bones jpg-type image (890 px × 694 px), and lung jpg-type image (248 px × 189 px) (Fig. 4a-c), respectively.

Binarization

To work with color images in P-Lingua, many rules will be required to test all possibilities of color relationships between pixels, which, in turn, will impose high computational overhead. To address this drawback, the color images will be converted into binary (black and white) images. The binarization will reduce not only the number of required rules but also the computational time.

Therefore, the input image will be converted to binary with only black and white pixels to simplify the segmentation rules and speed up the execution. Binarization refers to the process in which each pixel in an image is converted into one bit with the value of one or zero depending on the specified threshold value of all pixels. If the pixel value is greater than the particular threshold value, then the pixel will be converted to one; otherwise, it will be zero. A binary image is a digital image with only two possible values for each pixel (black or white). In the proposed approach, the average RGB value of each pixel is computed and then compared with the threshold value (128). If it is greater than the threshold, then the pixel will be converted to white; otherwise, it will be black.

Converting an image to P-Lingua syntax format

Typically, in the standard P-Lingua syntax format, the pixels should have color and coordinates. Basically, the color and coordinates of each pixel will be read and converted to P-Lingua syntax. The standard syntax is to indicate the color of the pixel followed by curly brackets that contain the coordinate of the associated pixel, which, in turn, will be written into the.txt file. For example, in $B\{x,y\}$, x and y are the coordinates of the black pixel in the image as shown in Fig. 5. In this paper, all pixels are converted to the syntax of P-Lingua automatically in the system (Fig. 5).
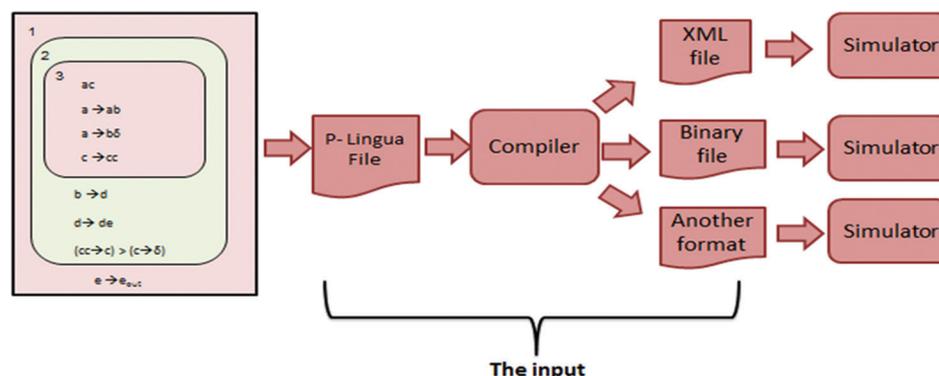
Fig. 1. Workflow of the PLinguaCore Java simulator (García-Quismondo, et al., 2009).
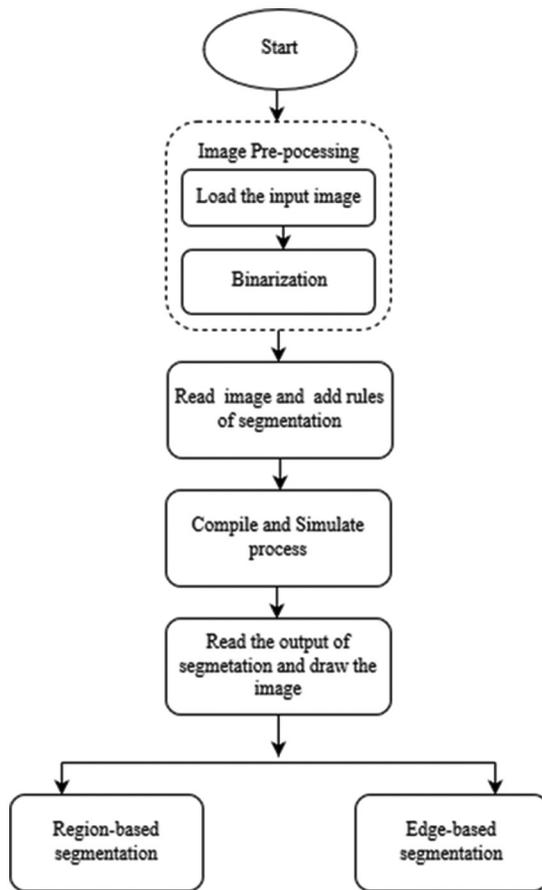
Fig. 2. Methodology of automatic segmentation of real image.

Fig. 3. Algorithm of image segmentation.

Writing the rules of segmentation in the P-Lingua format

In this paper, the segmentation rules follow the rules of MC for the 2D image segmentation method proposed in the work of Christina1, et al. (2011). The rules are written in the P-Lingua format, appended to the input image, and then saved in a text file, but with a (.pli) extension. This file is now ready to be executed.

Technically speaking, a 2D digital image can be encoded by a matrix in which each pixel in the image is an element of the matrix. Basically, this paper considers two types of neighborhoods surrounding a pixel - namely 4-adjacency and 8-adjacency. In 4-adjacency, four neighborhoods $\{(x-1,y), (x,y+1), (x+1,y),$ and $(x,y-1)\}$ contain only the pixels above, below, to the left, and to the right of the central pixel (x,y) as shown in Fig. 6. In 8-adjacency, eight neighborhoods include the four previous neighborhoods plus the four diagonal neighbors, which are "$\{(x-1,y-1), (x-1,y), (x-1,y+1), (x,y+1), (x+1,y+1), (x+1,y), (x+1,y-1),$ and $(x,y-1)\}$" as illustrated in Fig. 6.

For instance, if the image contains two colors, which are green and blue, if 4-adjacency is considered, the segmentation rules will be as follows:

$[G\{i,j\}, B\{i,j+1\}]'1 <--> [Gx\{i,j\},B\{i,j+1\}]'0;$
$[G\{i,j\}, B\{i,j-1\}]'1 <--> [Gx\{i,j\}, B\{i,j-1\}]'0;$
$[G\{i,j\}, B\{i+1,j\}]'1 <--> [Gx\{i,j\}, B\{i+1,j\}]'0;$
$[G\{i,j\}, B\{i-1,j\}]'1 <--> [Gx\{i,j\}, B\{i-1,j\}]'0;$

Along the same formulation, if 8-adjacency is considered, then four more rules will be added to cover the diagonal pixels as described in the following:

$[G\{i,j\}, B\{i,j+1\}]'1 <--> [Gx\{i,j\}, B\{i,j+1\}]'0;$
$[G\{i,j\}, B\{i,j-1\}]'1 <--> [Gx\{i,j\}, B\{i,j-1\}]'0;$
$[G\{i,j\}, B\{i+1,j\}]'1 <--> [Gx\{i,j\}, B\{i+1,j\}]'0;$
$[G\{i,j\}, B\{i-1,j\}]'1 <--> [Gx\{i,j\}, B\{i-1,j\}]'0;$
$[G\{i,j\}, B\{i-1,j+1\}]'1 <--> [Gx\{i,j\},B\{i-1,j+1\}]'0;$
$[G\{i,j\}, B\{i-1,j-1\}]'1 <--> [Gx\{i,j\}, B\{i-1,j-1\}]'0;$
$[G\{i,j\}, B\{i+1,j-1\}]'1 <--> [Gx\{i,j\}, B\{i+1,j-1\}]'0;$
$[G\{i,j\}, B\{i+1,j+1\}]'1 <--> [Gx\{i,j\}, B\{i+1,j+1\}]'0;$

Linking Java with C# and P-Lingua

As mentioned previously, P-Lingua is an official language for membrane computing. The P-Lingua program can encode a family of P systems (with the help of some parameters) in a flexible manner, whereas the object code generated by the compilation tool specifies only a single P system of the family. In this way, the applications which accept that object code is not required to process parametric systems, so their implementation is much easier.

Once the P-Lingua file with extension.pli is ready, it will be compiled and executed through PLinguaCore4. Recalling
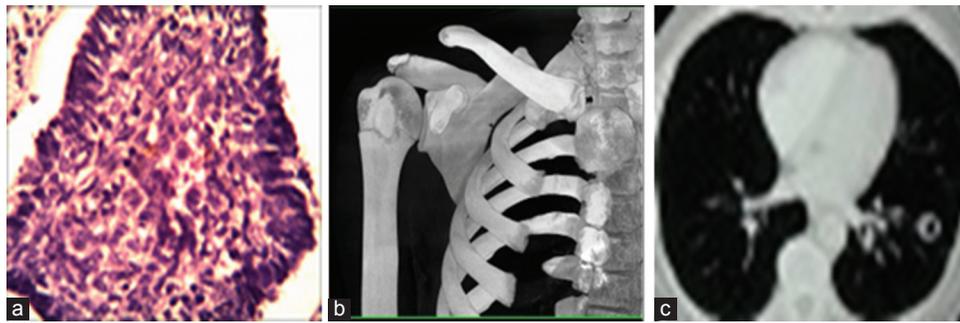
Fig. 4. Input images example; (a) skin cancer, (b) bones, and (c) lungs.
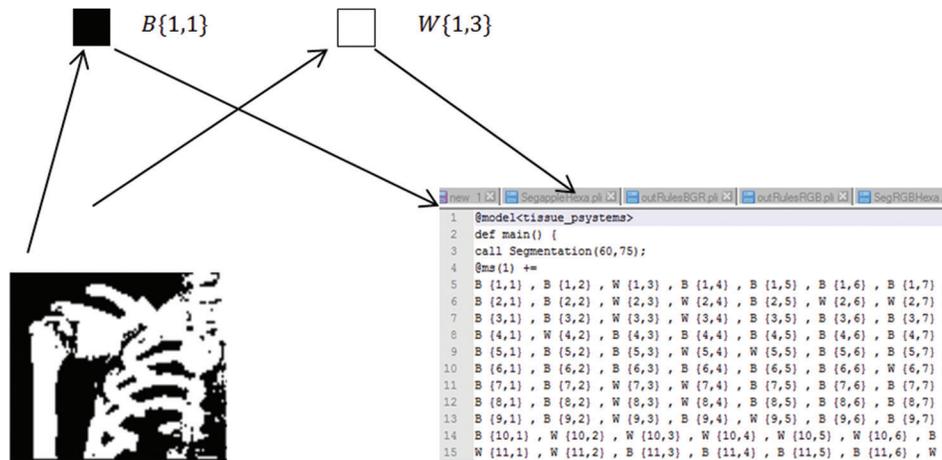
Fig. 5. Illustration of the binary image in P-Lingua.
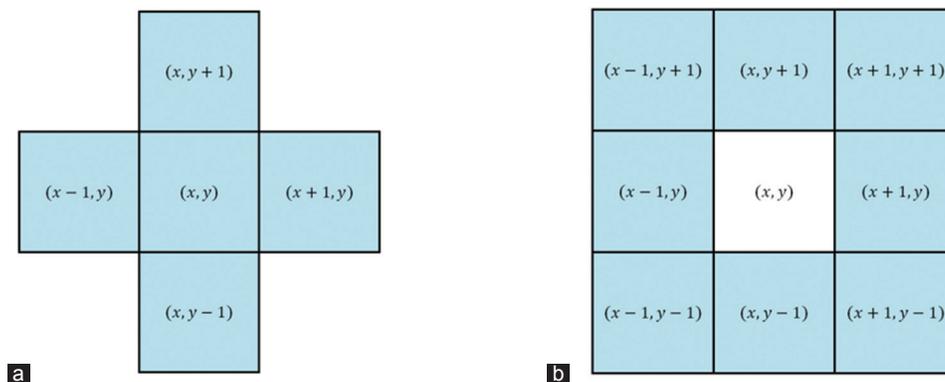
Fig. 6. 4 and 8-adjacency; (a) 4-neighborhood and (b) 8-neighborhood.

that C# should be linked to Java to perform this operation, a Java process (java.exe) will be invoked from the C# environment.

Reading output and visualization

The final result of the segmentation process is written to the file that is generated after segmentation (.txt). This file contains the details of every step of segmentation as well as information such as the configuration, input of the cell, output of the cell, environment, time of every step, memory usage, and final time of execution as shown in Table I.

Once the image has been segmented using MC, recalling that our model contains two different cells, the output can be read from any of these cells as needed. In the first case, if the

output is read from cell two, then edge-based segmentation will be obtained and visualized. It is worth mentioning that cell two contains the border and edge pixels, which is why edge-based segmentation can be obtained from this particular cell. Otherwise, in the second case, if the output is being read from cell one, then region-based segmentation will be obtained. More specifically, two types of segmentation strategies can be obtained regarding the philosophy of reading the output from the chosen cell.

Reading output and visualization from cell one

To draw the image according to the region-based criteria, the outputs of cell one and cell two must be read. To

Algorithm 2:   Drawing of the segmented image according to Region-Based Segmentation

	input : The generated output file in text
	output : The segmented image based on region –based segmentation
1	**begin**
2	**While** *outfile* contain *lines* **do**
3	Read the generated output file line by line
4	Create a string Output to store the edge from cell two
5	Create a string outputcellone to store the white pixels from cellone
6	Create a string outputcellone1 to store the black pixels from Cell one
7	Read the final execution time and print it
8	Read the memory and print it
9	**if** *Output* is not empty **then**
10	prepare the string output to store only the coordinates of the border pixels
11	**if** *Outputcellone* is not empty **then**
12	prepare the string outputcellone to store only the coordinates of the white pixels
13	**if** *Outputcellone1* is not empty **then**
14	prepare the string outputcellone1 to store only the coordinates of the black pixels
15	Create a bitmap image (outputting) to visualize the pixels
16	**for** (i=0;i<outputimg .Hight; i++) **do**
17	**for** (j=0; j<outputimg. Width; j++) **do**
18	**if** (output(i,j) = = 1) **then**
19	Draw this pixel as blue
20	color=color.FromArgb(0,0,255);
	outputimg.SetPixel(j,i,color);
21	**if** (outputcellone(i,j) == 1) **then**
22	Draw this pixel as blue
23	color=color.FromArgb (255,255,255);
	outputimg.SetPixel(j,i,color);
24	**else** (outputcellone1 (i,j) == 1) **then**
25	Draw this pixel as black
26	color=color.FromArgb(0,0,0);
	outputimg.SetPixel(j,i,color);
27	preview the outputimg in the picture box

Fig. 7. Algorithm of drawing the output image from cell one.

formulate this idea, three strings will be created to keep track of the output from the two cells. To be specific, the first string called the output will store the output of cell two. Similarly, the second and third string - namely outputCellone and outputCellone1 - will be used to store the contents of cell one. Technically speaking, outputCellone will be used to keep track of the white-colored pixels from input cell one. In the same context, outputCellone1 will be utilized to keep track of the black-colored pixels from input cell one. To obtain the output of cell two from Jout.txt, the algorithm will search for the last index of the word "Multiset" and take a substring starting from this index until the last index of "ENVIRONMENT;" this will return the multiset of cell two only. Fig. 7 explains the algorithm to perform region-based segmentation.

Reading output and visualization from cell two

Once the image has been segmented using MC rules in the P-Lingua environment, we must visualize the image after edge-based segmentation. To accomplish this, we must read the output P-Lingua file that contains the result of edge-based segmentation. As commonly known in MC, P-Lingua

Algorithm 3:   Drawing of the segmented image according to Edge-Based Segmentation

	input : The generated output file in text
	output : The segmented image based on edge –based segmentation
1	**begin**
2	**While** *outfile* contain *lines* **do**
3	Read the generated output file line by line
4	Create a string Output to store the edge from cell two
5	Read the final execution time and print it
6	Read the memory and print it
7	prepare the string output to store only the coordinates of the border pixels
8	**if** *Output* is not empty **then**
9	Replace all the "}," with "};"
10	Get a substring starting from the first occurance of ","
11	Replace all the "WX" with " "
12	Replace all the "}" ""
13	Split string output based on ;and store in temp
14	**for** (i=0; i<temp.length;i++)do
15	Index=temp[i].Split(',');
16	outArray[int.Parse(index[0]-1,int.Parse(index[1])-1] =1;
17	Create a bitmap image (outputting) to visualize the pixels
18	**for** (i=0;i<outputimg .Hight; i++) **do**
19	**for** (j=0; j<outputimg. Width; j++) **do**
20	**if** (outArray(i,j) == 1) **then**
21	Draw this pixel as black
22	color=color.FromArgb(0,0,0);
	outputimg.SetPixel(j,i,color);
23	**else** (outArray(i,j) == 0)
24	Draw this pixel as white
25	color=color.FromArgb(255,255,255);
	outputimg.SetPixel(j,i,color);
26	Preview the outputimg in the picture box

Fig. 8. Algorithm of drawing the output image from cell two.

generates an output file that stores the result of the input and the output cell as long as the configuration continues.

To draw the image according to the edge-based segmentation strategy, the output of only cell two must be read. In contrast, it is notable that in the region-based segmentation technique, the outputs of both cell one and cell two are read synchronously. To establish this idea technically, an output string will be created to keep track of the output from cell two. To be specific, this string, called "output," will store the output of cell two, which contains the coordinates of the edge pixels (borders). Specifically, to obtain the output of cell two from the generated output file "Jout.txt," the algorithm will search for the last index of the word "Multiset" and take a substring starting from this index until the last index of "ENVIRONMENT;" this will return the multiset of only cell two. Fig. 8 explains the algorithm to perform edge-based segmentation (Table II).

V. Efficiency of the Proposed Tissue-Like P System Segmentation Compared to Ground Truth Image

In this section, a thorough evaluation procedure is carried out to demonstrate the robustness of the proposed method on real image segmentation. To illustrate the effectiveness of the proposed segmentation approach, the segmentation results obtained from four adjacencies are compared with eight adjacencies. The algorithm is evaluated in terms of accuracy and efficiency, in which accuracy corresponds to the degree

TABLE I
COMPARING THE PERFORMANCE OF 4-ADJACENCY AND 8-ADJACENCY USING TWO DIFFERENT PLATFORMS BASED ON REGION-BASED SEGMENTATION IN TERMS OF TIME AND MEMORY SPACE

Images	Binarization	Segmentation		Intel Core i5			Intel Core i7	
				4-adjacency	8-adjacency		4-adjacency	8-adjacency
100×126 (bmp)			T	4852.885 s	4466.74 s	T	1689.833 s	1646.49 s
			M	423613 kb	645781 kb	M	825744 kb	654328 kb
100×77 (jpg)			T	1212.017 s	1180.237 s	T	610.749 s	608.125 s
			M	647924 kb	594864 kb	M	714432 kb	805258 kb
64×47 (jpg)			T	129.433 s	123.891 s	T	94.584 s	87.834 s
			M	891580 kb	568198 kb	M	607691 kb	695966 kb
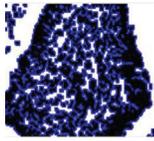
T: Time used for segmenting the image. M: Memory space used for segmenting the image

TABLE II
COMPARING THE PERFORMANCE OF 4-ADJACENCY AND 8-ADJACENCY USING TWO DIFFERENT PLATFORMS BASED ON EDGE-BASED SEGMENTATION IN TERMS OF TIME AND MEMORY SPACE.

Images	Binarization	Segmentation		Intel Core i5			Intel Core i7	
				4-adjacency	8-adjacency		4-adjacency	4-adjacency
100×126 (bmp)			T	4708.789s	4641.864s	T	3024.055 s	2568.154 s
			M	469123 Kb	652117kb	M	684737 kb	616432 kb
100×77 (jpg)			T	1659.376 s	1334.951 s	T	517.114 s	515.771 s
			M	802063 kb	581857 kb	M	713158 kb	858164 kb
64×47 (jpg)			T	137.026 s	125.355 s	T	82.119 s	70.67 s
			M	632686 kb	750024 kb	M	570762 kb	711489 kb
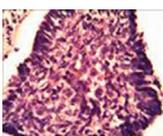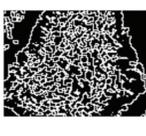
T: Time used for segmenting the image. M: Memory space used for segmenting the image

with which the delineation of the object corresponds to the truth. Efficiency corresponds to the amount of time required to perform the segmentation. The Jaccard index method (Shi, et al., 2014) is used to measure the accuracy of the segmentation for both region-based metrics and boundary-based metrics. Typical region-based evaluation metrics correspond to the ratio of the number of matching pixels to the total number of both matching pixels and mismatching pixels. Boundary-based metrics focus on the evaluation of the distorted segment result's boundary compared to the ground

truth image. To evaluate the segmentation quality, the Jaccard index method measures the similarity between the ground truth of the original image and that of the segmented image. The segmentation result and its corresponding ground truth are typically denoted by S and G, respectively. The Jaccard index is defined as follows:

$$\text{Jaccard Index} = \frac{A(G \mid S)}{A(G \cup S)} \qquad (1)$$

Where, A (.) is the operation of counting the amount. In Equation (1), the numerator corresponds to the number of matching pixels (true positives), whereas the denominator counts the total number of matching and mismatching pixels. Basically, it is an absolute "0" or "1," and the closer the value to one, the better the segmentation.

In the first experiment, we compare the performance of 4-adjacency and 8-adjacency relations for both region-based and edge-based real image segmentation, and the results are

presented in Tables III and Table IV, respectively. From the obtained results, it is evident that 4-adjacency outperforms 8-adjacency for region-based segmentation. However, for edge-based segmentation, 8-adjacency achieves better results than 4-adjacency in the skin cancer image and lung image, whereas for the bone image, the performance is similar, as shown in Fig. 9.

As can be depicted from the results in Fig. 9, changing the value of the threshold dramatically affects the accuracy of segmentation. For instance, considering the edge-based segmentation of the bone image, when the value of the threshold is set to 0.9, the accuracy is very low, whereas when setting the value of the threshold to 0.5, the best results are obtained.

In the next experiment, ground truth benchmark data are used to compare the performance between 4-adjacency and 8-adjacency in terms of accuracy. The benefit of using benchmark data is that one is able to compare results with those of other approaches in a significant way. Therefore, using ground truth data, the same experimental procedure is repeated for edge-based and region-based segmentation as
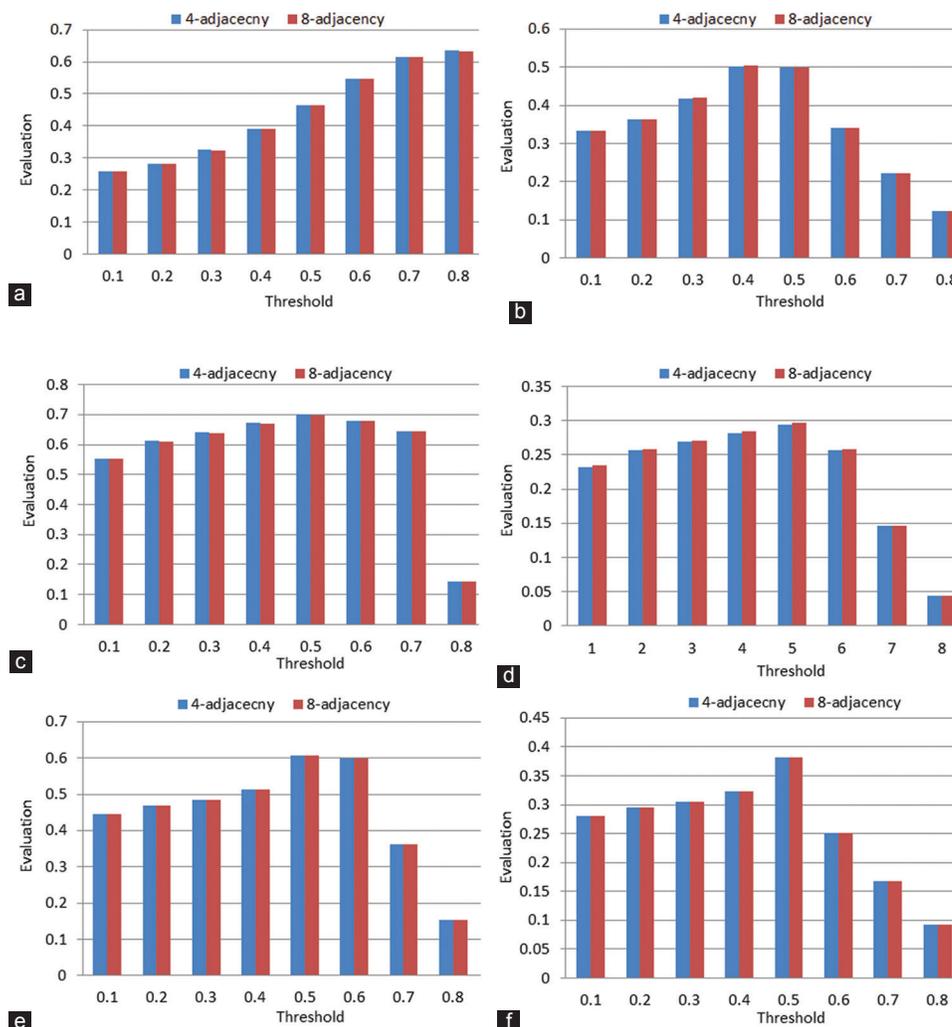
Fig. 9. Effect of the value of threshold parameter on region- and edge-based segmentation performance; (a) and (b) skin cancer image, (c) and (d) lung image, and (e) and (f) bone image.
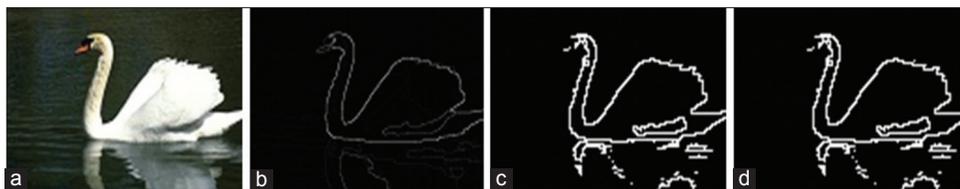
Fig. 10. Original image, ground truth, and results of the proposed edge-based segmentation.
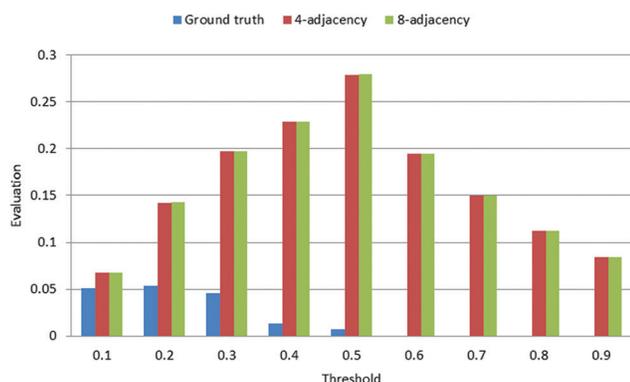
Fig. 11. Comparison between the ground truth benchmark with the proposed 4-adjacency and 8-adjacency edge-based segmentation in terms of evaluation against the threshold.

TABLE III
JACCARD INDEX ACCURACY FOR 4-ADJACENCY AND 8-ADJACENCY OF
REGION-BASED SEGMENTATION

Results	Skin cancer	Bone	Lung
4-adjacency	0.6344	0.6071	0.7011
8-adjacency	0.6336	0.6071	0.6987

TABLE IV
JACCARD INDEX ACCURACY FOR 4-ADJACENCY AND 8-ADJACENCY OF
EDGE-BASED SEGMENTATION

Results	Skin cancer	Bone	Lung
4-adjacency	0.5028	0.3815	0.2941
8-adjacency	0.5032	0.3815	0.2965

TABLE V
JACCARD INDEX VALUES OF THE GROUND TRUTH IMAGE, 4-ADJACENCY, AND
8-ADJACENCY (EDGE-BASED SEGMENTATION)

Type of image	Ground truth	4-adjacency	8-adjacency
Swan	0.0507	0.2791	0.2795

presented in the following.

Edge-based segmentation

In this experiment, the dataset used for evaluating the proposed system is a swan image of size 481 × 321 with its ground truth, as shown in Fig. 10. Note that this dataset has been taken from the Berkeley segmentation dataset (BSDS500) benchmark (Computer Vision Group, 2013).

The image of the ground truth data is compared to the original image of swan using the Jaccard index. The original image is segmented using the proposed tissue-like P system segmentation of two adjacency relationships - that is, 4-adjacency and 8-adjacency. The Jaccard index method is used to compare the result of the ground truth with 4-adjacency and 8-adjacency, as seen in Fig. 10. The results in Table V show that the segmentation using the 8-adjacency relationship outperforms the 4-adjacency counterparts for the swan image, and both 4- and 8-adjacency segmentations outperform the ground truth. In Fig. 11, the larger the threshold values, the better the segmentation accuracy of the proposed approach. However, with larger threshold values, the ground truth accuracy deteriorates, and the proposed tissue-like P system segmentation becomes better for deep visualization of the image including the shadow and the lake.

Region-based segmentation

In this experiment, the benchmark dataset used for evaluating the proposed system is a duck image (Rahtu, et al., 2010) with size of 400 × 300 pixels from the ground truth dataset was used for evaluation and the results as shown in Fig. 12. Region-based segmentation using 4-adjacency and 8-adjacency relationships was conducted using the proposed method.

As can be seen from the results in Table VI and Fig. 12, both types of relationship adjacencies achieved similar results. Fig. 12 reveals that the segmentation using ground truth is slightly better accurate than the proposed method, but visually in the ground truth image, the eye and peak of duck and the lines of sea were not accurately detected. For the proposed approach, the eye and peak of duck and the lines of sea were detected.

In Fig. 13, the effect of using different threshold values was evaluated and compared. The results show that, with a threshold value of 0.6–0.8, the best segmentation results were obtained.

VI. CONCLUSION

The effectiveness and robustness of the proposed work have been presented. Two different platforms were used in this experiment: A PC with an Intel Core i5-M430 processor running at 2.27 GHz and 4 GB of memory, and a PC with an Intel Core i7-M430 processor running at 2.27 GHz and 4 GB of memory.

In the first experiment, a region-based segmentation approach was evaluated, and the results showed that 8-adjacency relationships gave better performance in terms of computational time and memory usage. This was due to the capability of 8-adjacency with eight pixels around the center, which reduces the communication rules for obtaining the final segmentation output.

In the second experiment, an edge-based segmentation
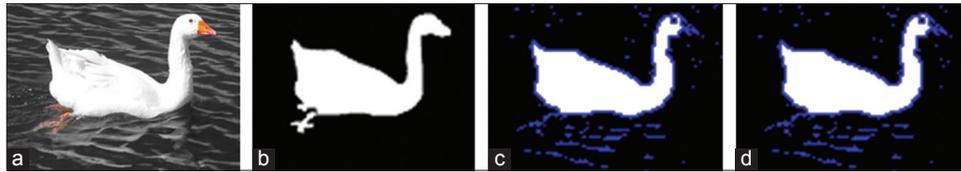
Fig. 12: Region-based segmentation results between original image, ground truth, and proposed method.
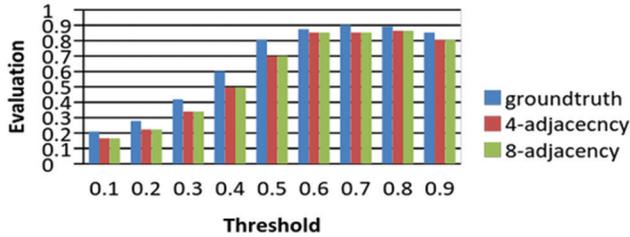
Fig. 13. Comparison between the ground truth benchmark and the proposed 4-adjacency and 8-adjacency region-based segmentation in terms of evaluation against thresholding.

TABLE VI
Jaccard Index Values of the Ground Truth Image, 4-Adjacency, and 8-Adjacency (Region-based Segmentation)

Type of image	Ground truth	4-adjacency	8-adjacency
Duck.png	0.9004	0.8642	0.8642

approach was evaluated, and once again the results of 8-adjacency are better, with significant time reduced using the Intel Core i7 platform.

The contributions of the paper are summarized as follows: First, the input image was codified automatically by linking P-Lingua with the C# platform. Second, two types of adjacency relationships were investigated and compared in terms of performance and speed on two different hardware platforms. The proposed methods were tested using real medical images from different image formats. Furthermore, two types of neighborhood adjacency were used to test the performance. It is worth mentioning that, to the best of our knowledge, no study has been performed on the efficiency of automatic real image segmentations using membrane computing. Our experimental results showed that the use of 8-adjacency achieved faster computational time due to the reduced number of rules needed to complete the segmentation process.

The limitation of our approach is on sequential architecture simulation for the experiments, which, in turn, does not exploit the massive parallelism inherent in P systems. In our future work, to make full use of the MC parallelism, a parallel architecture such as CUDA™ will be used to gain higher performance speedups over the typical serial implantation.

VII. Acknowledgment

References

Alsalibi, B., Venkat, I., Subramanian, K. and Christinal, H., 2014. *A Bio-Inspired Software for Homology Groups of 2D Digital Images*. Asian Conference on Membrane Computing ACMC 2014, Coimbatore, pp.1-4.

Carnero, J., Dıaz-Pernil, D. and Gutiérrez-Naranjo, M.A., 2011. Designing tissue-like P systems for image segmentation on parallel architectures. *Ninth Brainstorming Week on Membrane Computing*, 2011, pp.43-62.

Carnero, J., Díaz-Pernil, D., Molina-Abril, H. and Real, P., 2010. Image segmentation inspired by cellular models using hardware programming. *3rd International Workshop on Computational Topology in Image Context*, 1(3), pp.143-150.

Christinal, H.A., Díaz-Pernil, D. and Jurado, P.R., 2009. Segmentation in 2D and 3D image using tissue-like P system. *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Springer, Switzerland, pp.169-176.

Christinal, H.A., DíAz-Pernil, D. and Real, P., 2010. P systems and computational algebraic topology. *Mathematical and Computer Modelling,* 52(11), pp.1982-1996.

Christinal, H.A., Díaz-Pernil, D. and Real, P., 2011. Region-based segmentation of 2D and 3D images with tissue-like P systems. *Pattern Recognition Letters,* 32(16), pp.2206-2212.

Christinal, H.A., Dıaz-Pernil, D., Gutiérrez-Naranjo, M.A. and Pérez-Jiménez, M.J., 2010. Thresholding of 2D images with cell-like P systems. *Romanian Journal of Information Science and Technology (ROMJIST),* 13(2), pp.131-140.

Christinal, H.A., Díaz-Pernil, D., Jurado, P.R. and Selvan, S.E., 2012. Color segmentation of 2D images with thresholding. *Eco-friendly Computing and Communication Systems*, 305, pp.162-169.

Computer Vision Group. 2013. *Contour Detectionand Image Segmentation Resources*. University of California, Berkely. Available from: http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html. [Last accessed on 2016 Jan 10].

Díaz-Pernil, D., Berciano, A., Peña-Cantillana, F. and Gutiérrez-Naranjo, M.A., 2013. Segmenting images with gradient-based edge detection using membrane computing. *Pattern Recognition Letters,* 34(8), pp.846-855.

Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Molina-Abril, H. and Real, P., 2012. Designing a new software tool for digital imagery based on P systems. *Natural Computing,* 11(3), pp.381-386.

Diaz-Pernil, D., Gutiérrez-Naranjo, M.A., Real, P. and Sánchez-Canales, V., 2010. Computing homology groups in binary 2D imagery by tissue-like P systems. *Romanian Journal of Information Science and Technology,* 13(2), pp.141-152.

Díaz-Pernil, D., Molina-Abril, H., Real, P. and Gutiérrez-Naranjo, M., 2010. A bio-inspired software for segmenting digital images. Proceeding of the 2010 IEEE fifth international conference on Bio-inspired computing theories and applications BIC_TA, computer. *Society*, 2, pp.1377-1381.

García-Quismondo, M., Gutiérrez-Escudero, R., Pérez-Hurtado, I. and Pérez-Jiménez, M.J., 2009. P-lingua 2.0: New features and first applications. *Proceedings of the Seventh Brainstorming Week on Membrane Computing, Sevilla, Spain,* 1, pp.141-167.

Ionescu, M., Paun, G. and Yokomori, T., 2006. Spiking neural P systems. *Fundamenta Informaticae,* 71(2), pp.279-308.

Isawasan, P., Venkat, I., Subramanian, K., Khader, A., Osman, O. and Christinal, H., 2014. Region-based segmentation of Hexagonal digital images using membrane computing. *Asian Conference on Membrane Computing (ACMC),* pp.1-4.

Martı, C., Păun, G. and Pazos, J., 2003. Tissue P systems. *Theoretical Computer Science,* 296(2), pp.295-326.

Martin-Vide, C., Pazos, J., Păun, G. and Rodríguez-Patón, A., 2002. A new class of symbolic abstract neural nets: Tissue P systems. *Computing and Combinatorics,* 2327, pp.290-299.

Păun, G. and Rozenberg, G., 2002. A guide to membrane computing. *Theoretical Computer Science, Elsevier,* 287(1), pp.73-100.

Păun, G., 2000. Computing with membranes. *Journal of Computer and System Sciences,* 61(1), pp.108-143.

Păun, G., 2002. Introduction: Membrane computing—what it is and what it is not. *Membrane Computing.* Springer, London, pp.1-6.

Peña-Cantillana, F., Díaz-Pernil, D., Berciano, A. and Gutiérrez-Naranjo, M.A., 2011. A parallel implementation of the thresholding problem by using tissue-like P systems. *International Conference on Computer Analysis of Images and Patterns,* 29, pp.277-284.

Pena-Cantillana, F., Dıaz-Pernil, D., Christinal, H.A. and Gutiérrez-Naranjo, M.A., 2011. Smoothing problem in 2D images with tissue-like P systems and parallel implementation. *Proceedings of the Ninth Brainstorming Week on Membrane Computing,* Fénix Editora, pp.317-328.

Peng, H., Shao, J., Li, B., Wang, J., Pérez-Jiménez, M.J., Jiang, Y. and Yang, Y. 2012. Image thresholding with cell-like P systems. *Proceedings of the Tenth Brainstorming Week on Membrane Computing,* 2, p.3.

Peng, H., Wang, J. and Pérez-Jiménez, M.J., 2015. Optimal multi-level thresholding with membrane computing. *Digital Signal Processing,* 37, pp.53-64.

Peng, H., Yang, Y., Zhang, J., Huang, X. and Wang, J., 2014. A region-based color image segmentation method based on P systems. *Science and Technology,* 17(1), pp.63-75.

Rahtu, E., Kannala, J., Salo, M. and Heikkila, J., 2010. Segmenting salient objects from images and videos. ECCV, pp.366-379. Available from: http://www.cg.cs.tsinghua.edu.cn/people/~cmm/saliency2/.

Reina-Molina, R., Carnero, J. and Diaz-Pernil, D., 2010. Image segmentation using tissue-like P systems with multiple auxiliary cells. *Image-A,* 1(3), pp.143-150.

Shapiro, L. and Stockman, G.C., 2001. *Computer Vision.* 2001 ed. Prentice Hall, Englewood. Cliffs, NJ.

Sheeba, F., Thamburaj, R., Nagar, A.K. and Mammen, J.J., 2011. *Segmentation of Peripheral Blood Smear Images using Tissue-Like P Systems.* Bio-Inspired Computing: Theories and Applications (BIC-TA), 2011 Sixth International Conference on, IEEE, pp.257-261.

Shi, R., Ngan, K.N. and Li, S., 2014. Jaccard index compensation for object segmentation evaluation. *IEEE International Conference on Image Processing,* 71, pp.4457-4461.

Yahya, R.I., Hasan, S., George, L.E. and Alsalibi, B., 2015. Membrane computing for 2D image segmentation. *International Journal of Advances in Soft Computing and its Applications,* 7(1), 1-15.

Yahya, R.I., Shamsuddinirst, S.M. Hasan, S., Yahyam, S.I., Hasan, S., Salibi B. and Al-Khafajiand, G., 2017. Image segmentation using membrane computing: A literature survey. Chapter: Bio-inspired computing – Theories and applications. *Communications in Computer and Information Science,* 368, pp.314-335.

Yahya, R.I., Shamsuddinirst, S.M., Hasan, S. and Yahya, S.I., 2016. Tissue-like P system for Segmentation of 2D hexagonal images. *ARO-The Scientific Journal of Koya University,* 4(1), pp.35-42. doi: http://dx.doi.org/10.14500/aro.10135

Yang, Y., Peng, H., Jiang, Y., Huang, X. and Zhang, J., 2013. A region-based image segmentation method under P systems. *Journal Information Computer Sciences,* 10(10), pp.2943-2950.

Zhang, Z. and Peng, H., 2012. Object segmentation with membrane computing. *Journal of Information and Computational Science,* 9(17), pp.5417-5424.