# Artificial Cells for Information Processing: Iris Classification

Enrique Fernandez-Blanco, Julian Dorado, Jose A. Serantes, Daniel Rivero, and Juan R. Rabuñal

University of A Coruña, Campus Elviña, A Coruña, Spain
{efernandez,julian,jserantesv,drivero,juanra}@udc.es

**Abstract.** This paper presents a model in the Artificial Embryogene (AE) framework. The presented system tries to model the main functions of the biological cell model. The main part of this paper describes the Gene Regulatory Network (GRN) model, which has a similar processing information capacity as Boole's Algebra. This paper also describes how to use it to perform the Iris Classification problem which is a pattern classification problem. The aim of this work is to show that the model can solve this kind of problems.

**Keywords:** Artificial Embryogeny, Genetic Algorithms.

## 1 Introduction

Nature presents a lot of different systems that can be used in Computer Science as an inspiration to develop new tools. Examples like Artificial Neural Networks (ANNs) or Genetic Algorithms are well-known. This work starts from the idea that any cell of a biological tissue has to communicate and process the signals of its environment. This behavior can be seen as distributed computation, where each cell plays the character of a single processor and it has to coordinate its computation with its neighbor cells. Nature just needs a few signals from the environment and the information contained in the DNA to develop and coordinate the most complicated structures.

The objective of the present work is to develop a model inspired in embryological cells, which have features like self-organization, self-reparation, etc. To develop the computer adaptation, the biological model was simplified by identifying and modeling the essential elements. In this way, some parts of the computer adaptation have very similar functions to the biological ones (DNA, gene or cytoplasm, etc.). The main objective of this paper is to adapt this model in order to apply it to information processing problems and, in particular, to classification and pattern recognition problems.

## 2 Background

In 2003, Stanley and Miikmulainen [1] developed a methodology to classify the different AE models that appear in Evolutionary Computation (EC). These models are

based on abstractions of the embryological cells, which can be classified into two main types. On one hand, some works follow a grammatical approach, where the most important works are related to L-systems [2]. On the other hand, other studies have a chemical-oriented approach based on Turing's ideas [3].

The works related to the grammatical approach have been mostly used to develop ANN. Kitano's work [4] shows how the connectivity matrix of an ANN is evolved with a set of rules. Another remarkable work is [5], in which the authors develop both the control and the structure of the robot using L-systems.

For the chemical approach, the first work to be mentioned is [6], in which Kauffmann develops his Gene Regulatory Networks [6]. The objective of the works which follow this approach is usually its application to different problems, such as approximating a simple figure/structure in a 3D space, or the development of evolutionary hardware [7]. One of the most important works which tries to solve the previously mentioned problems is described in [8]. The most interesting part is the usege of fractal proteins for the communication among the cells of the model.

The model presented in the current paper can be included into the chemical approach. The model has included most of the concepts present in the previously mentioned works, like the concept of operon or the cellular division and death. The most novel concept is that never before a chemical approach has been used to solve an information processing problem.

## 3 Model

Every cell of any biological tissue has as antecessor: a unique cell, called zygote, which generates other cells and they can coordinate their behavior using the information present in DNA. Each cell knows its purpose from its DNA and the proteins that it receives. Therefore, it can be considered that each cell of the tissue works as a processor and all of them operate with proteins using the DNA as operator set. Self-reparation, self-organizationand parallel information processing are some features of the structures generated with this computation model [9].

Below, the structures of the artificial model that arise from the study of the biological issues are explained.

**Protein**
Protein is the basic piece of information. In this model, proteins are a string of bits that identifies each one of the different proteins and has a time to live (TTL). Due to this, the system has a memory of previous generated proteins, until they are used or degraded.

**Cytoplasm**
Cytoplasm is the part of the artificial cell which has the responsibility of managing the information inside the cell. The responsibility of this part is to manage the proteins needed for a transcription in the cell and to check the concentration level of the proteins inside and outside the cell to decide which proteins will be communicated.

**Gene**
Each gene of the system represents a rule, where some conditions have to be fulfilled to perform a certain computation or process. The genes are strings of bits which contain two parts: promoters and a gene identifier.

- **Promoter region**. This part identifies the proteins needed to activate this gene. This section can appear several times and it is composed of two parts:
  - **Promoter Sequence**. This section identifies the required protein sequences to activate the gene.
  - **Concentration lock**. Each of the activation proteins needs a certain concentration level identified in this field in order to activate this gene.
- **Gene identifier**. This part identifies which is the protein generated by the gene and the type of the gene. It is composed of two subparts.
  - **Generated sequence**. When the gene is activated, the result of that activation is a protein with this sequence.
  - **Constitutive mark**. This bit indicates if the gene is a constitutive gene (explained below).

The required proteins have to be at least in a certain proportion inside the cell. This minimum concentration is stored in its concentration lock part. The protein does not need to be identical to the activation protein. As in Nature, high concentrations of similar proteins can activate a gene. This fact is modeled using the following condition:

$$\text{Protein Concentration Percentage} >= (\text{Distance} + 1) * \text{Concentration Lock} \quad (1)$$

In the previous condition, *Protein Concentration Percentage* represents the concentration of a protein inside the cell. *Distance* is the hamming distance between that protein and the activation protein. Finally, *Concentration Lock* is the concentration required for the promoter sequence. If the condition is met for all of the promoters, then the gene generates a protein with the gene's generated sequence.

This is the normal behavior of a gene but when it is marked as constitutive its behaviour changes drastically. A constitutive gene is constantly generating proteins, until its activation proteins appear. In this case, these are called inhibitor proteins. When the condition Eq. 1 is met by the inhibitor proteins, the gene stops producing proteins during a certain period.

**Operon**

Operon is a group of genes which codify a task. In Nature, these genes codify the most complex tasks and act all or none of them. This idea was adapted by a structure which applies conditions to a group of genes. This structure has the same parts as a gene and acts in the same way but, instead of a generated sequence, it has a group of genes. This allows the activation of those genes from that moment for a period of time.

**DNA**

The DNA is composed of genes and operons. Its responsibility is to select the allowed genes which can be activated, when the cell asks for them in a certain moment of its development.

**Cell**

Cells are the basic element of the system and contain all the previously described parts (DNA, Gene, etc.). The expression of the DNA can induce the cell to communicate with its environment. The expression of the DNA can also induce two special behaviors: division or death of the cell.

Cells define a concept to regulate the different actions, which is the cellular time or *cellular cycles.* These *cellular cycles* contain all the tasks that a cell can do at the same time. For example, a cell can communicate proteins to the environment many times, but it can only divide itself once in a cellular cycle. The steps of a cycle are the following:

1. Update the concentration of the proteins and delete those whose TTL reaches zero.
2. Process the DNA with the cytoplasm proteins to generate new proteins.
3. Check the concentration of the proteins and execute the communications of the cell, or execute one of the special actions (divide and death) if it is necessary.

In short, after the actualization of the proteins' TTL, when a cell activates a gene, four different actions can be performed: storage the protein in the cytoplasm; communicationamong cells the proteins; division of the cell or death of the cell. All of these actions are performed by each cell in such a way that the cellular cycle is used to coordinate the actions.

**Environment**

The environment is where the cells are and it determines the type of communication among cells. The main purpose of the environment is to manage the free proteins. The free proteins are those which have been set out by a cell and no cell has already required them. The environment determines how these proteins move until they are required by a cell or when they are deleted because their TTL has expired.

**Communication Model**

Once the parts and the environment have been defined, the next step is to define how these elements interact among them. Each cell can communicate with its neighborhood by using the proteins. Proteins are set into the environment when their concentration inside the cell is higher than a certain threshold.

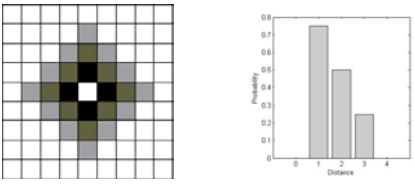


**Fig. 1.** Reception Probability of a Protein

The proteins set into the environment modify the return value of a probability function associated to its type. This function gives the probability of finding a protein in a point far away from the emission point. When a new protein is set into the environment, the probability associated to that point is increased. This function has two parameters: the number of proteins in the environment's position and the distance between that point and the checked point. The probability of finding a protein decreases with the distance (Fig. 1). The decreasing value is represented in Fig. 1 by the darkness in the squares that surrounds the emission point. Cells check in each *cellular cycle,* the proteins that have any possibility to be caught by them. If the protein is taken by a cell, then the value of the function is decreased to represent the reduction of the amount.

## 4 Instruction Search Method

To search the instructions with the shape of an artificial DNA strand, a Genetic Algorithm (GA) has been used [10]. The reasons for choosing a GA are: it is one of the most robust and adaptable methods and the features of a GA match the dynamics of the presented system.
The GA's DNA is not a fixed-length array of variables because the number of rules and the number conditions of these rules (promoter sequence) are unknown This paper proposes the subdivision of the DNA in functional sections, (see Fig. 2):

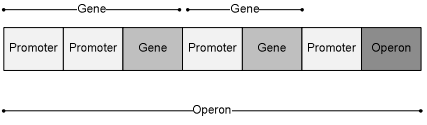


**Fig. 2.** Compiling a DNA

- **Promoter** sections are sections that codify a gene's promoter region.
- **Gene** sections identify a gene of the cellular system and codify the gene identifier. This section is associated to the previous promoter sections that appear in the GA individual until another Gene identification section or Operon section is found.
- **Operon** section is a mark of the creation of an operon of the cellular system. This operon has as promoters the previous promoter sections, until a Gene or operon identification section appears. The genes contained into the operon are those which have been identified before the operon. The operon contains the genes, until a maximum number of genes is reached, or it finds another operon identification section is found.

These sections and the association show in Fig. 2 are able to search all the possible combinations of genes. To simplify the GA's functioning, the sections used by the GA have all the fields of the three functional sections, but these sections have two extra bits to determine the type and which are the valid fields. The next step is to adapt the GA crossover and mutation operators to work with the variable length.

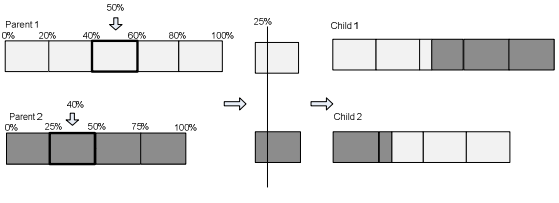


**Fig. 3.** Crossover Example

The crossover operator used is a one point crossover, which selects the crossover section according to the length of each individual. A random percentage over the length, which is the number of functional sections, is generated for each individual

and the section in that position is selected. Both parents randomly choose the same point inside the sections to make the crossover and the children are the combination of the parents' genetic material as shows Fig. 3.

The mutation operator can execute one of these three possible operations:

- Add a new section to an individual.
- Delete a section.
- Change a bit inside a section.

In the mutation operator, the operation which changes a bit needs more explanation because each section has two bits that identify its type at the beginning of the binary string. The first bit identifies if it is a promoter section or not. The second bit determines if it is an operon or a gene (when it is not a promoter). A change in any of these bits makes the information of the section to be reinterpreted in its new role because the new role activates different fields inside the section used by the GA. Other changes in bits only change the associated value of the field. Note that the GA sections have all the fields of the functional sections to simplify this step.

Finally, the probabilities of executing each of the actions, empirically obtained, are: Addition (20%), deletion(20%) and change (60%). The percentage is the probability of executing each operation each time a mutation is executed.

## 5 Test

This DNA model defines a Gene Regulatory Network (GRN). This kind of structure has proved to have the same capacities as Bool's Algebra [6].

To adapt the model to process information problems, the inputs and outputs of the system have to be defined. As inputs, some source points put in the environment new proteins each *cellular cycle*. The outputs are measured by setting some sinks which get the proteins from the environment. The final element of the system is the set of cells into the environment, which process the inputs and set the outputs. The objective of the test is to search DNA which minimizes the output error in the sinks when the inputs are present in the source points. The DNA is put into a cell and the input values are put in the source points. After a number of cellular cycles, the output values are checked in the sink points.

The iris classification problem is a non-linear classification problem. Four parameters were measured in millimeters on 50 iris specimens from each of three iris species, *Iris setosa*, *Iris versicolor* and *Iris virginica [11]*. Given the four parameters, one should be able to determine which of the three classes a specimen belongs to. The patterns are distributed in three areas, one for each class. One of the types presents a clear difference with the other two, but the other two classes are very close and they present a conflict area with no clear differentiation.

$$fitness = \sum_i \begin{cases} 100 & \text{no protein registered} \\ 1 & \text{no desired protein} \\ 0 & \text{desired protein in the first position of the list} \\ 0.5 & \text{desired protein in the second position of the list} \\ 0.75 & \text{desired protein in the third position of the list} \\ 0.85 & \text{desired protein in other position in the list} \end{cases} \quad (2)$$

The model tries to find a DNA which classifies the different patterns of iris flowers. To adapt the model to this problem, the values of the variables were normalized between 0 and 1. One sink and 4 source points, one for each variable, were used. The source points put into the environment 4 different proteins, one for each variable. The normalized value of a variable is the peek probability of receiving it around the source point. This probability decreases linearly from the source point to the surrounding area, and it determines the probability to find a protein in that point. Three sequences are identified as the desired outputs and they determine the predicted class of the input data: Iris Setosa (1111), Iris Versicolor (1001) and Iris Virginica (1010). These desired sequences will be the principal component of the fitness function. The GA will seach a DNA which maximizes the explresion of this desired sequences. The ordered list of received proteins is needed to check the position of the desired sequence. The function will add a different penalization depending on the position into the ordered list as shown in Eq. 2. The penalization was selected empirically.

The letter i represents a pattern of the training set. The fitness is the addition of the penalizations for each pattern. These penalization values are determined by the position on the list. There are two special cases: when the desired protein is not present the penalization is 1 and when no protein is received then the error is 100.

Finally, the GA used to search the DNA, which allows this classification, has shown a better behavior with a 70% crossover rate and a 30% mutation probability in populations with a size between 50 and 100 individuals.

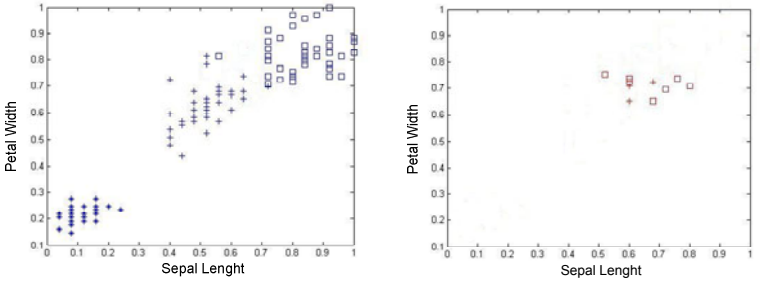


**Fig. 4.** Iris Flower correct classification. Correctly classified (left), wrongly classified (right).

The best individual has a fitness of 8.85 and it is able to classify 140 of 150 patterns. Fig 4 shows the three kinds of iris flowers represented by a star, a cross and a square. These results were obtained using only one cell to solve the problem. More cells would increase the complexity of the solution, but the resulted structure may classify better the patterns of the conflicted area by the cooperation among cells.

## 6 Conclusions

This paper presents a model which follows a different approach to previous works. It has two objectives. The first one is to be able to be applied to any kind of problem. The second one is to capture the interesting features of the biological model.

These features, already mentioned in the text, (like self-organization, self-repairation, distributed control, parallel information processing, etc) are those that this model tries to take from Biology and to use in the resolution of problems. The presented system is the base to future developments in the knowledge area of self-organization and distributed computation. The systems built with this cellular model present a high self-organizing and distributed computation level. Future developments with this system could be framed into Autonomic Computing proposed by IBM in 2003[12] because they present many of the features that these systems require. This model could be the base for the communication between different elements in a Autonomic Computing system.

Finally, the tests, in this paper, have shown how the model can be applied to solve different information processing problems. If these results are added to the previous ones [9], a new research area is found. The aim of this area is to develop self-organizing structures which can process information.

## Acknowledgement

This work was partially supported by the Spanish Ministry of Education and Culture (Ref TIN2006-13274) and the European Regional Development Funds (ERDF), grant (Ref. PIO52048 and RD07/0067/0005) funded by the Carlos III Health Institute, grant (Ref. PGIDIT 05 SIN 10501PR) and (Ref. PGDIT 07TMT011CT) from the General Directorate of Research of the Xunta de Galicia and grant (File 2006/60, 2007/127 and 2007/144) from the General Directorate of Scientific and Technologic Promotion of the Galician University System of the Xunta de Galicia. Thanks to the Galician Supercomputation Center (CESGA) which has execute a great part of the tests.

## References

1. Stanley, K., Miikkulainen, R.: A Taxonomy for Artificial Embryogeny. In: Proceedings of Artificial Life 9, pp. 93–130. MIT Press, Cambridge (2003)
2. Lindenmayer, A.: Mathematical models for cellular interaction in development: Part I and II. Journal of Theorical Biology 18, 280-299, 300–315 (1968)
3. Turing, A.: The chemical basis of morphogenesis. Philosofical Transactions of the Royal Society B 237, 37–72 (1952)
4. Kitano, H.: Designing neural networks using genetic algorithm with dynamic graph generation system. Complex Systems 4, 461–476 (1990)
5. Hornby, G.S., Pollack, J.B.: Creating high-level components with a generative representation for body-brain generation. Artificial Life 8(3) (2002)
6. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of Theoretical Biology 22, 437–467 (1969)
7. Steiner, T., Jin, Y., Sendhoff, B.: A Cellular Model for the Evolutionary development of Lightweight Material with an Inner Structure. In: Proceedings of Genetic and Evolutionary Computation Conferences 2008, Atlanta, pp. 851–858 (2008)
8. Kumar, S.: Investigating Computational Models of Development for the Construction of Shape and Form. PhD Thesis. Department of Computer Science, University College London (2004)

9. Fernandez-Blanco, E., Dorado, J., Rabuñal, J.R., Pedreira, N., Gestal, M.: A Computational Approach to Simple Structure Development. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 825–834. Springer, Heidelberg (2007)
10. Holland, J.H.: Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor (1975)
11. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html
12. Kephart, J.O., Chess, D.M.: The vision of Autonomic Computing. IEEE Computer Society Press, Los Alamitos (2003), 0018-9162. 41-50