# Genetic Programming-Based Approach for System Identification

Jose Aguilar, Mariela Cerrada

Departamentos de Computación y Sistemas de Control, CEMISID.
Facultad de Ingeniería, Universidad de los Andes
Avenida Tulio Febres Cordero
Mérida 5101-VENEZUELA

*Abstrac:.-* In this work, an approach based on Genetic Programming is proposed for the input-output systems identification problem. Laguerre's functions and the ARX method have been commonly used to solve the systems identification problem. Recently, approaches based on Artificial Neural Networks have been used to solve this problem. Genetic Programming is an Evolutionary Computation technique based on the evolution of mathematical symbols (constants, functions, variables, operators, etc.). To achieve the identification, a set of analysis trees is used to describe the different models (individuals) that our approach proposes during its execution. At the end of the evolutionary process, an input-output model of the system is proposed by our approach (it is the best individual).

*Key-words:* Genetic Programming, Evolutionary Computation, Identification Systems

## 1. Introduction

The Evolutionary Computation (EC) proposes computational schemes based on the evolutionary behavior of the species in order to solve particular problems [3],[5]. The EC includes the Genetic Algorithms, the Evolutionary Programs, the Evolutionary Programming, the Evolutionary Strategies and the Genetic Programming (GP). Particularly, the GP guides the evolution of a set of procedures to solve a particular problem, such that the best procedure that solves the problem is the final solution [7], [8], [12].

On the other hand, in processes control is required models, which describe the dynamic behavior of the system in order to carry out control tasks [2], [11]. The System Identification (SI) problem consists in proposing an approximated model of a real system [2], [9], [11]. Models that only manipulates input and output variables is one of the possible identification schemes (Input-Output Identification problem). In control theory, there are many techniques to solve this problem [2], [9], [11]: Laguerre's Functions, the ARX Method, etc. Recently, other techniques based on Intelligent Computation [1], [10], [12], [13] (Artificial Neural Networks, Evolutionary Programming, etc.) have been used.

In this work, we propose a mechanism based on the GP for the input-output identification problem of dynamic systems. This approach guides the evolution of a function set toward an input-output mapping of the system by using PG. Our approach manipulates a population of analysis trees (each analysis tree is an individual), and the crossover and mutation operators are used to change the population. At the end of the evolutionary process, the best individual represents the system identification model. The computational implementation of this mechanism is made using the genetic programming library "The Genetic Programming Kernel" [4].

## 2. Theoretical Background

In this section, we present the different theoretical aspects used in this work.

### 2.1 Genetic Programming (GP)

The GP is an EC technique. In general, EC techniques manipulate sets of individuals (possible solutions of a given problem) by using genetic operators, which are inspired in biological operators, in order to propose better individuals. The general algorithm is:

```
1. Define the initial population.
2. Repeat until the system convergence:
        2.1 Evaluate the individuals.
        2.2 Reproduce the best individuals by using genetic operators.
        2.3 Replace the worst old individuals by the best new individuals.
```

In GP, the individuals do not represent the solution of a given problem. Now, they represent algorithms/procedures to solve such problem [7], [8]. Thus, the GP find out the best procedure to solve a problem. In GP, the individuals are defined by a function set (subprograms, mathematical functions, etc.) and a terminal set (constants, variables, etc.). The PG uses analysis trees to codify the individuals (solutions), where each tree node is a symbol that belongs to one of the sets presented above. An example of an analysis tree of the mathematical equation A+B is shown in the figure 1.

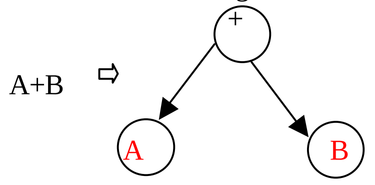

**Fig. 1.** Example of an analysis tree

Normally, the GP uses two operators: *crossover* and *mutation* [7], [8]. The crossover operator exchanges two sub-trees from two tree randomly selected. In this way, two new trees are created. The mutation operator randomly selects a tree and creates a new tree by taking a sub-tree and replacing it by other one, which is randomly generated. These operators preserve the syntactic constraints of the models.

### 2.2 Systems Identification

In several control tasks, it is necessary to known the system model that describes the behavior of the system [2], [11]. This model can be defined as a non-linear function of the current input and previous inputs and outputs. An input-output identification technique permits to define a model of an unknown system from the historical data of its inputs and outputs [2], [9], [11]. Such a model can be used in control tasks, fault tolerance, etc. The classic scheme for the systems identification is shown in the figure 2. The error signal between the real output and the estimated output is used to update the model parameters. There are many identification methods, several of them based on the classic control theory [2], [9], [11], an others based on the intelligent computation [1], [10], [12], [13].

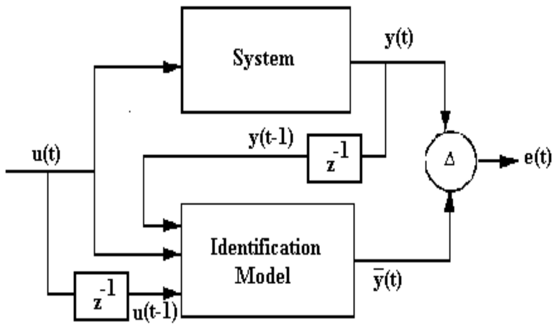

**Fig. 2.** Systems Identification Scheme.

## 3. Our Identification Methods Using PG

In this work is proposed a method based on PG to develop identification models. In our approach, each individual is defined by a Multiple Interaction Programs (MIP) model (see figure 3.a). In the MIP model, each node is one equation, which is represented by an analysis tree (see figure 3.b).

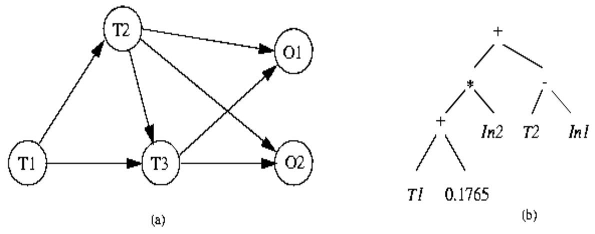

**Fig. 3.** MIPs Model.

In our model, the terminal set of each node has input variables, constants or outputs from some precedent equations. In the figure 3.b is shown an analysis tree for T3, where *In1* y *In2* are input values of the problem. *T1* y *T2* are the outputs of these equations, which precede T3 (see figure 3.a). This model is easy to implement in GP, through the utilization of the ADF (Automatic Definition Function) technique. This extension of GP permits to define functions to evolve in parallel with the main procedure. These functions can be called by other functions, or by the main procedure, during the evolution. In our case, the MIP model defines the relationship among the functions. The population evolution follows the next algorithm:

1. Define a given MIP model for the individuals.
2. Generate, randomly, a population of individuals. Each one of the individuals is defined by a set of analysis trees according to the MIP model.
3. Evaluate each individual. The evaluation function is the average error between the historical output of the system and the output of the identification model (individual).
4. Select the parents (individuals with the smallest average error).
5. Apply the genetic operators to these parents in order to generate new individuals.
6. Replace the old worst individuals for the new individuals.

## 4. Experiment

In this section, an example is presented in order to prove the performance of our method. The system is a distillation system that uses a binary distillation column in continuous operation of multiple stages [6]. To develop the computational program, we have used the "The Genetic Programming Kernel" library, designed by A. Fraser en 1994 [4]. This library permits the utilization of ADFs.

### 4.1 The System Description

The objective of a distillation system is to separate a mixture in two or more fractions with different boiling points. The function of the continuous distillation system can be seen with details in [6]. In the figure 4 is shown the structure of this distillation column. The feeding input is introduced in the second plate, and the distilled product is obtained in the first plate on the top of the column.

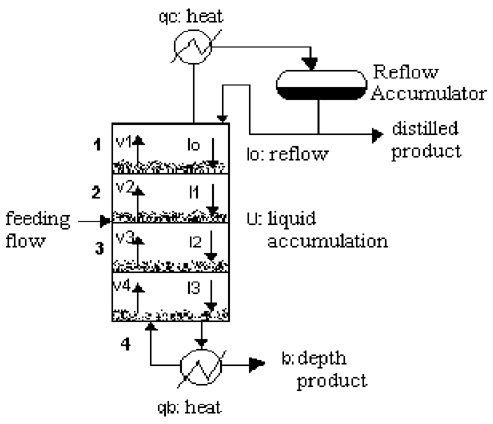

**Fig. 4.** Distillation Column

The constant input signal (feeding) is modeled with a step function with amplitude equal to ten (U(t) = 10). The theoretical model of this system is given by the equation (1) [6]:

$$X(t) = 1.1148*X(t-1) + 0.2525*X(t-2) - 0.3823*X(t-3) + 0.3294e-4*U(t-1)$$

where X(t) represents the output of the system. The output is the concentration of benzene on the top. The output signal from this model is shown in the figure 5.

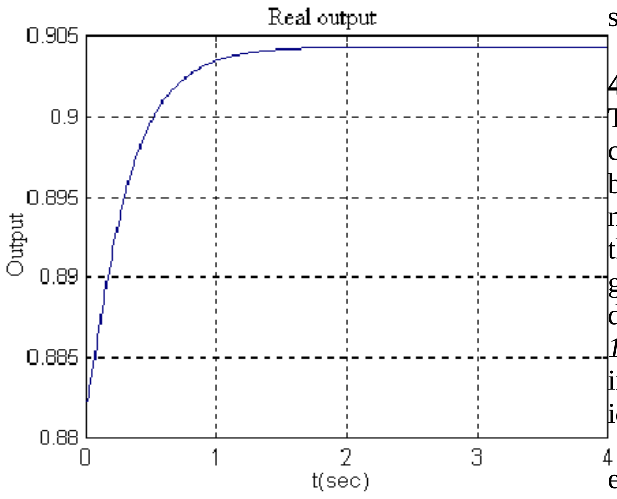

**Fig. 5.** Output signal of the system

## 4.2 The System Identification using our Approach

In this experiment, the MIP model is composed by two equations (M1 y M2), where M2 represents the ADF and M1 represents the main program (main tree), which can depend of M2 or not. The function set used by M1 and M2 is {+,-,*, %, *sin*, *cos*}. The terminal set of the main tree is composed by St(M1)={*u*, *xa1*, *xa2*, *xa3*, *xa4*, *xa5*, *s_M2*}, where *u* is the input signal at the time *t*, *xa1* is the output signal at the time t-1 *(X(t-1))*, *xa2* is the output at the time t-2 *(X(t-2))*, and so forth, and s_M2 is the output of the ADF. The terminal set of the ADF only has two elements St(M2)={*xa1*, *xa2*}. The trigonometric functions are supposed with input values given in radians.

The historical values of the input and output signals have been obtained using the theoretical model defined by the equation (1). The aptitude of each individual was determined based on the average error between the output historical values and the outputs of the model proposed by the individual for the same set of input signals. A population of 300 individuals was evolved through 50 generations. Finally, the individual with the smallest average error is selected. In the table 1 is shown the models obtained (the best individuals) using our identification method, for different terminal sets.

## 4.3 Results Analysis

The identification models obtained in the cases 1 and 4 are similar, and they are the best models. In the second case, the ADF model is different to the previous ones, but the value of the error is acceptable. In general, in all cases the best individual depends of the output signal at the times (*t-1*) and (*t-2*), and it does not depend of the input signal. In the second case, the identification model is very complex.

In the figure 6 is shown the identification error signal obtained by the model proposed in the case 2. The input signal is a constant function U(t)=10, and the initial conditions for the variables xa1 y xa2 was randomly selected near of the real initial conditions. At t=2 sec., the identification error converges to zero. In the figure 7 is shown the identification error for the same previous case, but with initial conditions equals to zero. The identification error quickly converges to zero (at t = 0.05 sec.).

## 5 Conclusions

In this work has been shown that our identification method using PG can propose suitable input-output identification models for dynamic systems. It is desirable to carry out comparisons with others identification techniques (classics or based on intelligent systems) in order to establish the advantages of this approach.

According to our results, this method is efficient as identification technique. This efficiency depends of the function and terminal sets that are used, and the relationship established in the MIP model. In the future, we are going to test one extension of our approach where the MIP model evolves such that the proper evolution determines the optimal relation between the equations.

**Table 1.** Identification Models

| Cases | Identification Models | Identification Error |
|---|---|---|
| St(M1)={u, a1, xa2, xa3, s_M2} St(M2)={x1, x2} | M1 = 2*xa1 – xa2*s_M2 M2 = $(xa1)^2$/ xa2 | 1.59254e-4 |
| St(M1)={u, xa, xa2, s_M2} St(M2)={x1, x2} | M1 = 2*xa1 – xa2* s_M2 M2 = Equation (2) | 2.0965e-4 |
| St(M1)={u, xa1, a2, xa3, xa4, s_M2} St(M2)=={x1, x2} | M1 = (xa1 / xa2)*xa1 M2 = xa1+xa2-xa3 | 2.86043e-4 |
| St(M1)={u, xa1, xa2, xa3, xa4, xa5, s_M2} St(M2)={x1, x2} | M1 = 2*xa1 – xa2 *s_M2 M2 = $(xa1)^2$ / xa2 | 1.59254e-4 |

where,

M2 = xa1 – sin(sin(sin(sin(sin(sin(sin(sin(sin(sin(sin xa1)))))))))))))

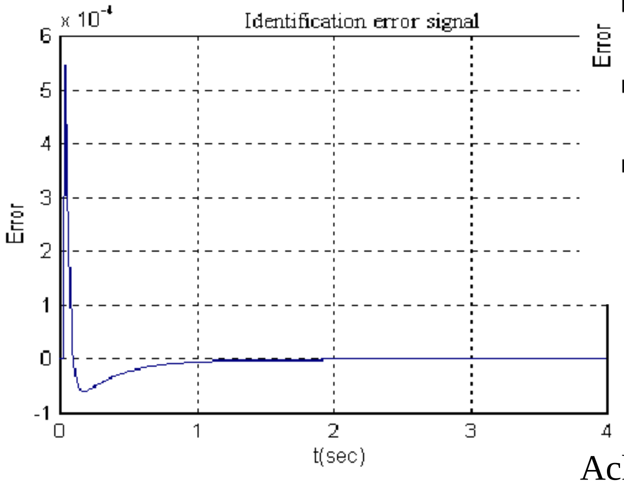

**Fig. 6**. Identification error using the second model

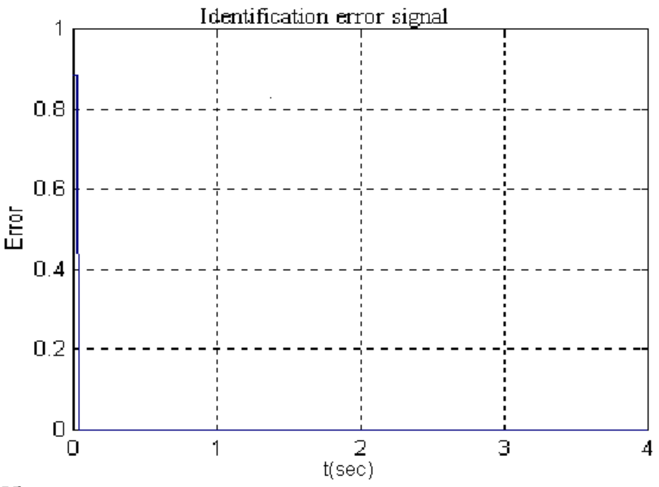

**Fig. 7.** Identification error with initial conditions xa1=xa2=0

## Acknowledgment

This work was partially supported by CONICIT-CONIPET grant *97003817* and CDCHT-ULA grant *I-621-98-02-A.*

*References*

[1] Angeline P., Fogel D.: An evolutionary program for the identification of dynamical systems. *Technical Report, Natural Selection*, Inc (1998).

[2] Wittenmark A.: *Computer Control Systems*. Addison Wesley New York (1989).

[3] Fogel D.: *Evolutionary Computation: Toward a New Philosophy through Simulated Evolution.* IEEE Press (1995).

[4] Fraser A.: *Genetic programming in C++*. Technical Report 040, University of Salfrod, Cybernetics Research Institute (1994).

[5] Goldberg D.: *Genetic Algorithms in Search,* Optimization and Machine Learning. Addison Wesley New York (1989).

[6] Holland C.: *Fundamentos y Modelos de Procesos de Separación.* Prentice Hall New York (1997).

[7] Koza J.: *Genetic Programming: On the programming of computers by means of natural selection.* MIT Press (1992).

[8] Koza J.: *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press (1994).

[9] Lennart L.: *System Identification, theory for the user*. Prentice Hall (1997).

[10] Lippmann R.: Review of neural networks for speech recognition. *Neural Computation*, Vol. 1. (1989) 1-38.

[11] Ogata K.: *Ingeniería de Control Moderna.* Prentice Hall (1992).

[12] Rodriguez K., Fleming P.: Genetic Programming for Dynamic Chaotic Systems Modelling. In: Angeline P., Michalewicz Z., Schoenauer M., Yao X., Zalzala A. (eds.): Proceeding of the 1999 Conference on Evolutionary Computation, Vol. 1. (1999) 22-28.

[13] Sastry P., Santharam G., Unnikrishnan K.: Memory neuron networks for identification and control of dynamical systems. *IEEE Trans. on Neural Networks*, Vol. 5. (1994) 306-319.