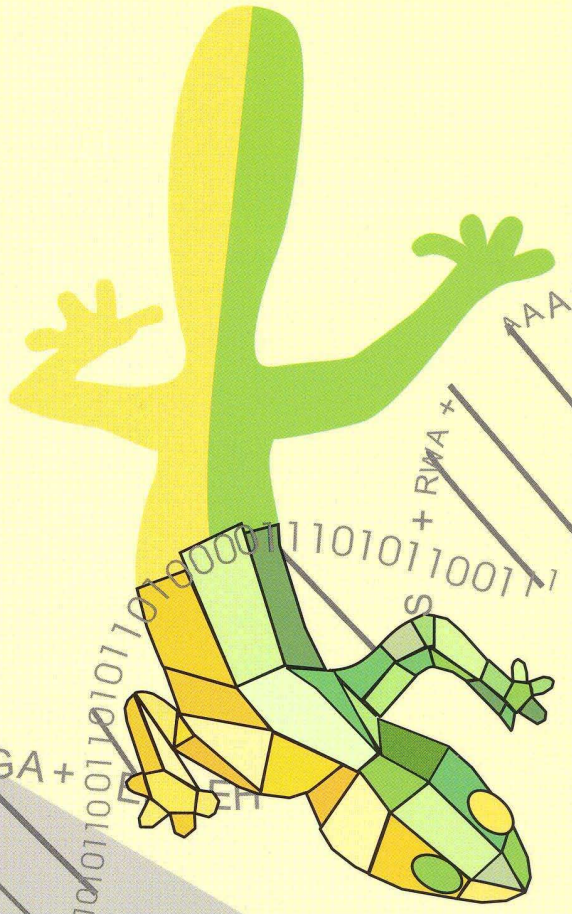
GECCO = GP + GA + ES + EP + EH + ER + DNA + LCS + RWA +

GECCO 2005

Genetic and Evolutionary Computation COnference

Volume 2

Hans-Georg Beyer et al. (Editors)



June 25-29, 2005
(Saturday - Wednesday)
Washington, D.C. USA

Sponsored by

ACM SIGEVO

Fractional Dynamic Fitness Functions for GA-based Circuit Design

Cecília Reis
Polytechnic Institute of Porto
Porto
351-228340500
cecilia@dee.isep.ipp.pt

J. A. Tenreiro Machado
Polytechnic Institute of Porto
Porto
351-228340500
jtm@dee.isep.ipp.pt

J. Boaventura Cunha
Univ. of Trás-os-Montes Alto Douro
Vila Real
351-259350339
jboavent@utad.pt

ABSTRACT
This paper proposes and analyses the performance of a Genetic Algorithm (GA) using two new concepts, namely a static fitness function including a discontinuity measure and a fractional-order dynamic fitness function. The GA is adopted for the synthesis of combinational logic circuits. In both cases, experiments reveal superior results in terms of speed and convergence to achieve a solution.

Categories and Subject Descriptors
B.6.3 [**Logic Design**]: Design Aids – *automatic synthesis, optimization.*

General Terms: Algorithms, Design.

Keywords: Fractional calculus, Genetic algorithms and Logic circuit design.

1. INTRODUCTION
In the last decade genetic algorithms (GAs) have been applied in the design of electronic circuits, leading to a novel area of research called Evolutionary Electronics (EE) or Evolvable Hardware (EH) [4]. EE considers the concept for automatic design of electronic systems. Instead of using human conceived models, abstractions and techniques, EE employs search algorithms to develop good designs.

Several authors applied GAs to the combinational circuit design problem using different techniques and approaches in order to achieve high-quality circuit designs [3].

Following this line of research, and looking for better performance GAs, this paper proposes a GA for the design of combinational logic circuits using an error discontinuity measure and fractional-order dynamic fitness functions.

The area of Fractional Calculus (FC) deals with the operators of integration and differentiation to an arbitrary (including noninteger) order and is as old as the theory of classical differential calculus [2]. The theory of FC is a well-adapted tool to the modelling of many physical phenomena, allowing the description to take into account some peculiarities that classical integer-order models simply neglect. Nevertheless, the application of FC has been scarce until recently, but the advances on the theory of chaos motivated a renewed interest in this field.

Bearing these ideas in mind the article is organized as follows. Section 2 describes the adopted GA as well as the fractional-order dynamic fitness functions. Section 3 presents the simulation results and finally, section 4 outlines the main conclusions.

GECCO'05, June 25-29, 2005, Washington, DC, USA.

ACM 1-59593-010-8/05/0006.

2. THE GENETIC ALGORITHM
A truth table specifies the circuit and the goal is to implement a functional circuit with the least possible complexity. Two sets of logic gates have been defined, Gate Set a = {AND,XOR,WIRE} and Gate Set b = {AND,OR,XOR,NOT,WIRE}, being Gset a the simplest one and Gset b a more complex gate set. The gate named WIRE means a logical no-operation [1].

In the GA scheme the circuits are encoded as a rectangular matrix of logic cells where three genes represent each cell: <*input1*><*input2*><*gate type*>, where *input1* and *input2* are one of the circuit inputs, if they are in the first column, or, one of the previous outputs, if they are in other columns. The gate type is one of the elements adopted in the gate set. The chromosome has many triplets of this kind as the matrix size demands (e.g., fig. 1).

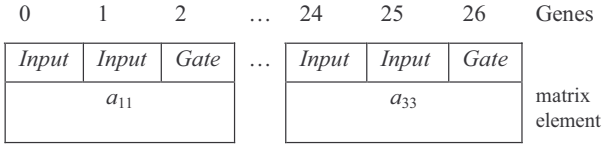


Figure 1

The initial population *P* of circuits (strings) is generated at random, then it is used a tournament selection to select the strings that are going to form the offspring. Single point crossover is applied and a mutation operator. Moreover, it is applied an elitist algorithm and, consequently, the best solutions are always kept for the next generation.

The crossover rate *CR* represents the percentage of the population *P* that reproduces in each generation. Likewise, the mutation rate *MR* is the percentage of the population *P* that can mutate in each generation.

In the paper, we propose two concepts for the fitness function, namely the error discontinuity measure static fitness function F_s and the dynamic fitness function F_d.

The calculation of F_s in (1) has two parts, f_1 and f_2, where f_1 measures the functionality and the error discontinuity and f_2 measures the simplicity. In a first phase, we compare **Y,** the

output produced by the GA-generated circuit, with $\mathbf{Y_R}$, the required values according with the truth table, on a bit-per-bit basis. By other words, f_{11} is incremented by one for each correct bit of the output until f_{11} reaches the maximum value f_{10}, that occurs, when we have a functional circuit. After this, f_{11} is decremented by $\delta \in [0, 1]$ for each error discontinuity. Once the circuit is functional, in a second phase, the GA tries to generate circuits with the least number of gates. This means that the resulting circuit must have as much genes *<gate type>* ≡ *<wire>* as possible. Therefore, the index f_2, that measures the simplicity (the number of null operations), is increased by *one* for each *wire* of the generated circuit, yielding:

$$f_{10} = 2^{ni} \times no \quad \textbf{(1a)}$$

$$f_{11} = f_{11} + 1 \text{ if } \{\text{bit } i \text{ of } \mathbf{Y}\} = \{\text{bit } i \text{ of } \mathbf{Y_R}\}\ ,\ i = 1, \ldots, f_{10} \quad \textbf{(1b)}$$

$$f_1 = f_{11} - \delta \text{ if error}_i \neq \text{error}_{i-1},\ i = 1, \ldots, f_{10} \quad \textbf{(1c)}$$

$$f_2 = f_2 + 1 \text{ if } \textit{gate type} = \textit{wire} \quad \textbf{(1d)}$$

$$F_s = \begin{cases} f_1, & F_s < f_{10} \\ f_1 + f_2, & F_s \geq f_{10} \end{cases} \quad \textbf{(1e)}$$

where *ni* and *no* represent the number of inputs and outputs of the circuit.

The concept of dynamic fitness function F_d results from an analogy with control systems where we have a variable to be controlled similarly with the GA case where we control the population through the fitness function. The simplest control system is the proportional algorithm; nevertheless, there can be other control algorithms, like the differential and the integral schemes. Therefore, applying the static fitness function corresponds to using a kind of proportional algorithm. If we want to implement a more sophisticated control, the fitness function needs a scheme of the type:

$$F_d = F_s + K\, D^{\alpha}[F_s] \quad \textbf{(2)}$$

where $-1 \leq \alpha \leq 1$ is the differential (integral) fractional-order for positive (negative) values of α and K is the 'gain' of the dynamical term.

In order to implement D^{α} we adopt a discrete-time calculation algorithm, based on the approximation of the time increment h through the sampling period T and an r-term truncated series yields (3):

$$D^{\alpha}[x(t)] \approx \frac{1}{T^{\alpha}} \sum_{k=0}^{r} \frac{(-1)^k \Gamma(\alpha+1)}{k!\ \Gamma(\alpha-k+1)} x(t-kT) \quad \textbf{(3)}$$

3. EXPERIMENTS AND RESULTS

The experiments consist on running the GA to generate a typical combinational logic circuit, namely a 2-to-1 multiplexer using the fitness schemes described previously. The circuits are generated with the gate sets a and b for *CR* = 95%, *MR* = 20% and *P* = 100.

Figure 2 shows the average number of generations to achieve the solution *AV*(*N*) using the static fitness function *versus* the discontinuity factor $\delta = \{0, 0.25, 0.5, 0.75, 1\}$, using Gset a.

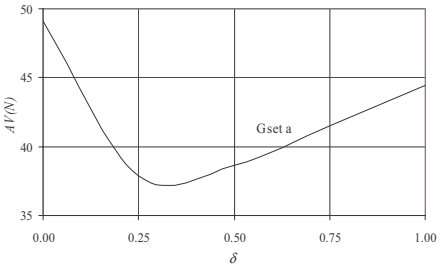


Figure 2

Figure 3 shows *AV*(*N*) using the dynamic fitness function *versus k* for $\alpha = \{0, 0.25, 0.5, 0.75, 1\}$, using Gset b.

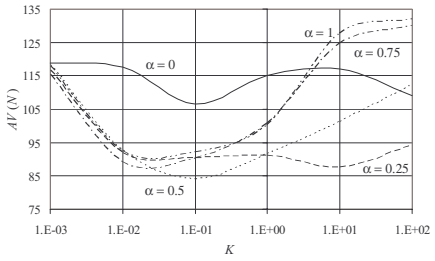


Figure 3

In both cases, the proposed schemes outperform the standard fitness algorithms.

4. CONCLUSIONS

This paper presented two techniques for improving the GA performance. In what concerns to the classical static fitness function we conclude that it is possible to get superior results by measuring the error discontinuity. On the other hand, the new concept of fractional-order dynamic fitness function demonstrates to be an important method to improve the GA scheme.

5. REFERENCES

[1] Cecília Reis, J. A. Tenreiro Machado, and J. Boaventura Cunha. Evolutionary Design of Combinational Logic Circuits, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Fuji Technology Press, Vol. 8, No. 5, pp. 507-513, Sep. 2004.

[2] J. A. Tenreiro Machado. Analysis and Design of Fractional-Order Digital Control Systems. *SAMS Journal Systems Analysis, Modelling, Simulation*, vol. 27: 107-122, 1997.

[3] Louis, S.J. and Rawlins, G. J. Designer Genetic Algorithms: Genetic Algorithms in Structure Design. In *Proc. of the Fourth Int. Conference on Genetic Algorithms*, 1991.

[4] Zebulum, R. S., Pacheco, M. A. and Vellasco, M. M., *Evolutionary Electronics: Automatic Design of Electronic Circuits and Systems by Genetic Algorithms*, CRC Press, 2001.