# Evolvable Hardware

**André Macário Barros**
*Universidade Tecnológica Federal do Paraná - UTFPR, Brazil*

**Heitor Silvério Lopes**
*Universidade Tecnológica Federal do Paraná - UTFPR, Brazil*

## INTRODUCTION

Evolvable Hardware (EHW) is an intelligent technology which belongs to a large area called Evolutionary Electronics (EEL), which includes applications of Evolutionary Computation (EC) in the domain of electronics. EHW combines concepts from Evolutionary Computation (EC) and Electronic Design (ED), and it has been an active area of research in the last years. This article provides an updated review of this area. First, EEL with focus on EHW is overviewed, pointing out the applicability of this technology. A brief review of two important EC paradigms is provided: GAs (Genetic Algorithms) and Genetic Programming (GP), since they are the most used in EHW. On the other hand, the basic aspects of ED are also shown with focus on Reconfigurable Computing (RC) and the Field Programmable Gate Arrays (FPGAs) technology. Next, the usual taxonomy found in the literature is presented: by design type - analog and digital, and by evolution type: extrinsic, intrinsic and mixtrinsic. Some relevant applications are presented and discussed. Finally, future trends are presented.

## BACKGROUND

As previously described, this subject merges two areas of study. A brief background of both areas is presented here to provide the necessary comprehension.

### Evolutionary Computation

Problems considered complex and hard (e.g., multimodality, large search space, large number of constraints) to be solved using conventional optimization techniques of Computer Science are being addressed by EC, which employs a collection of algorithms called Evolutionary Algorithms (EAs). EAs imitate the nature, specifically the Darwinian principle of survival of the fittest. In EC, an individual is the representation of a possible solution. A set of individuals forms the initial population, randomly created. The individuals of this population eventually produce descendants by means of selection, reproduction and mutation, according to the survival rule in which the best fitted individuals are those who will have more chances to reproduce. The descendants form a new population and the process is repeated for many generations. When a stopping criterion is met, the evolution stops, and the best individual ever found is the solution for the problem being handled.

The main EAs are: Evolutionary Programming (EP), (Fogel, 1962), (Fogel, Owens & Walsh, 1966), Evolution Strategies (ES) (Rechenberg, 1965, 1973), Genetic Algorithms (GAs) (Holland, 1975) (Goldberg, 1989), and Genetic Programming (GP) (Koza, 1992, 1999). GAs and GP are the most frequently used in EHW.

Considering P(t) a population of individuals at time t, based on the concepts previously presented, GAs and GP can be represented by the algorithm shown in Figure 1 and described as follows (Bäck, Fogel & Michalewicz, 2000a), Goldberg (1989):

### Representation

Before running the EA, it is necessary to choose the representation, i.e., how the candidate solutions are represented in during the execution of an EA. Data structures are the commonly representations used by computer programs. Depending on the problem, the data structure itself can be the real-world solution. But there are applications in which such a direct representation is not possible, for instance, a list of tasks, a mechanism, or an electronic circuit. Binary strings, finite state

DOI: 10.4018/978-1-4666-5888-2.ch703

S

*Figure 1. An evolutionary algorithm*

```
1- t=0
2- initialize P(t)
3- evaluate P(t)
4- while termination condition is false do
5-    select P(t+1) from P(t)
6-    crossover P(t+1)
7-    mutate P(t+1)
8-    evaluate P(t+1)
9-    t = t + 1
```

representations and parse trees are the data structures processed by the EAs. The term used for this indirect representation in EC is genotype, usually composed by one or more chromosomes and the one employed for the solution in the real-world is phenotype.

## First Generation and Evaluation

For algorithm presented on Figure 1, the initial population of individuals is randomly generated. Each individual of the population needs to be evaluated to inform the EA how good the individual is. This is called fitness evaluation and can be processed in two steps: a) the genotype to phenotype conversion – in the case of an indirect representation; and b) the fitness computation of the phenotype related to that individual. This computation is a procedure using expressions related to the problem to produce usually a number, which will be used as a measurement of quality. An example is an individual such as a sequence of cities to deliver a product. This sequence is used in an expression looking for higher profits. A sequence that provides a good profit receives a better evaluation than another individual that provides lower profits. Each individual of the population enters the while loop of the algorithm after have being evaluated.

## Selection

In this phase, the individuals of the population are selected for crossover according to their fitness value. This selection uses probability. So, the selected individuals can be the ones with higher fitness values, but there can be too some individuals with lower fitness values. The most known selection mechanisms are: proportional (or roulette wheel), tournament, truncation, linear rank, and exponential rank.

## Crossover

This operation consists of the exchange of genetic information between the selected individuals. They are called parents. For instance, a crossover between two parents "111111" and "000000" can produce two possible offspring such as "001100" and "110011." The two bits in their middle portion were changed. This operation is probabilistic. Crossover types most known are: one point, two point and uniform. A chromosome is composed of genetic information (subsets of bits or numbers) called building blocks. It is desirable that crossover operations allow the exchange of information between parents preserving their building blocks, promoting the positive evolution, i.e., descendants with better values of fitness then their ancestors. Crossover is an operator that performs local search in the search space of solutions.

## Mutation

After crossover, the descendants are probabilistically submitted to the mutation operator, which consists in changing the value of a single locus (a locus is a position where the symbol is in a chromosome) to another value. Example: a mutation at locus 0 in the binary chromosome "111000111," results in "111000110." Mutation provides diversity in the search for the solution, since it performs global search, an attempt to avoid the AE to be stuck at local maxima.

## Genetic Algorithms

GAs (Holland, 1975), (Goldberg, 1989), (Bäck, Fogel, & Michalewicz, 2000a), are the most popular EAs and all the features described previously are found in it.

## Genetic Programming

GP (Koza, 1992, 1999) is an EA that usually evolves computer programs. The chromosome is encoded in a tree structure. The tree is composed by function nodes and terminal nodes. The function nodes are, in most cases, arithmetic or mathematical operations and Boolean or conditional or iteration functions. The function nodes are responsible for connecting the terminal nodes, forming the tree, thus generating an evolved program in a LISP-like form.