

EVOLVING OPTIMAL MULTI-OBJECTIVE HARDWARE USING STRENGTH PARETO EVOLUTIONARY ALGORITHMS

N. Nedjah^a and L. M. Mourelle^b

^aDepartment of Electronics Engineering and Telecommunications,
Faculty of Engineering, State University of Rio de Janeiro,
Rio de Janeiro, Brazil
nadia@eng.uerj.br

^bDepartment of Systems Engineering and Computation,
Faculty of Engineering, State University of Rio de Janeiro,
Rio de Janeiro, Brazil
ldmm@eng.uerj.br

ABSTRACT

In this paper, we focus on engineering *Pareto-optimal* digital circuits given the expected input/output behaviour with a minimal design effort. The design objectives to be minimised are: hardware area, response time and power consumption. We do so using the Strength Pareto Evolutionary Algorithms. This is novel application of multi-objective optimisation to circuit design. The performance and quality of the circuits evolved for some benchmarks are presented then compared to those of single objective genetic algorithms as well as to the circuits obtained by human designers. We show that the evolutionary hardware is far better with respect to all objectives than those designed using traditional methods.

Keywords: Evolvable hardware, Multi-objective optimisation, Digital circuits, SPEA.

1. INTRODUCTION

Designing digital circuits is a well-established area with a variety of on-the-shelf design methods and techniques. These tools, however, worry only about fulfilling the expected input/output behaviour of the circuit with the exception of some of them, which allow engineering circuits of relatively reduced size, i.e. the number of gate used. With the latter, the designer counterpart is a considerable designing effort.

The problem of interest consists of how can one design optimal circuits that implement a given input/output behaviour without much designing effort. The obtained circuits are expected to be minimal in terms of space, time and power requirements: The circuits must be *compact* i.e. uses a reduced number of gates, *efficient*, i.e. produces the output in a short time and *easy* or *not demanding*, i.e. consumes little power. The response time and power consumption of a circuit depends on the number and the complexity of the gates forming the longest path in it. The complexity of a gate depends solely on the number of its inputs. Furthermore, the design should take advantage of the all the kind of gates available on reconfigurable chip of field programmable gate array (FPGAs).

Evolutionary hardware is yield using simulated evolution as an alternative to conventional-based electronic circuit design. Genetic evolution is a process that maintains a set of *individuals*, which constitutes the *generational population*, producing a new population. Here, individuals are digital circuit designs. The more the design obeys the constraints, the more it is selected to participate of the *reproduction* process. Besides the input/output behaviour, design constraints are expressed in terms of hardware area, response time and power requirements. The freshly produced population is yield using some genetic operators such as *crossover* and *mutation*

that attempt to simulate the natural breeding process in the hope of generating new design that are *fitter* i.e., respect more the design constraints. Genetic evolution is usually implemented using genetic algorithms.

In this work, we design innovative and efficient evolutionary digital circuits. Circuit evaluation is based on their possible implementation using CMOS technology. The produced circuits are balanced i.e., the trade-off between the required hardware area, the propagation time of the circuit output signals and the power consumption is the best. We do so using multi-objective evolutionary optimisation. We exploit the Strength Pareto Evolutionary Algorithm (SPEA) presented in (Zitzler and Thiele, 1999). SPEA is the most recent and efficient multi-objective evolutionary algorithm (Coello, 2001).

The rest of this paper is organised in five sections. First, in Section 2, we define multi-objective evolvable hardware and introduce the characteristics of the basic components allowed. Subsequently, in Section 3, we present the multi-objective evolutionary algorithm used to perform the evolution i.e., the strength Pareto evolutionary algorithm (Zitzler and Thiele, 1999). Thereafter, in Section 4, we describe the circuit encoding and the genetic operator used followed with the definition and implementation of the circuit fitness evaluation with respect to all three considered objective. Then, in Section 5, we evaluate the performance of the evolutionary process and assess the quality of the evolved Pareto-optimal digital circuits. Also, we compare the area, time and power requirements to those of circuit designs engineered by human designers and single objective genetic algorithm. Last but not least, in Section 6, we summarise the content of the paper and draw some useful conclusions.

2. PRINCIPLES OF MULTI-OBJECTIVE EVOLUTIONARY HARDWARE

Evolutionary hardware consists simply of hardware whose designs were evolved using genetic algorithms, wherein individuals represent circuit designs. In general, evolutionary hardware designs offer a mechanism to get a computer to provide a design of circuit without being told exactly how to do it. In short, it allows one to automatically create circuits. It does so based on a high level statement of the constraints the yielded circuit must respect. The input/output behaviour of the expected circuit is generally considered as an omnipresent constraint. Furthermore, the generated circuit should have a minimal size, minimal response time or minimal power consumption. However, this is the case in single objective evolutionary computation.

Starting from random set of circuit designs, which is generally called *initial population*, evolutionary hardware design breeds a population of designs through a series of steps, called *generations*, using the Darwinian principle of natural selection. Circuits are selected based on how much they adhere to the specified constraints. *Fitter* individuals are selected, recombined to produce off-springs which in turn should suffer some mutations. Such off-springs are then used to populate of the next generation. This process is iterated until a circuit design that obeys to all prescribed constraints is encountered within the current population.

Each circuit within the population is assigned a value, generally called *fitness*. A circuit design is fit if and only if it satisfies the imposed input/output behaviour. In single objective optimisation, a circuit design is considered *fitter* than another if and only if it has a smaller size, shorter response or consumes less power, depending of the optimisation objective size, time or power consumption minimisation respectively. In multi-objective optimisation, however, the concept of fitness is not that obvious. It is extremely rare that a single design optimises all objectives simultaneously. Instead, there normally exist several designs that provide the same *balance*, *compromise* or *trade-off* with respect to the problem objectives.

Both in single and multi-objective optimisation, an important aspect of evolutionary hardware design is thus to provide a way to evaluate the adherence of evolved circuits to the imposed constraints as well as the corresponding quality. First of all, the evolved circuit design must fulfil the input/output behaviour, which is given in a tabular form of expected results given the inputs. This is the truth table of the expected circuit. Second, the circuit must Pareto optimal. This constraint allows us to yield digital circuits with optimal trade-offs regarding area, time and power consumption.

We estimate the necessary area for a given circuit using the concept of *gate equivalent*.

This is the basic unit of measure for digital circuit area complexity (Ercegovac et. al., 1999). It is based upon the number of logic gates that should be interconnected to perform the same input/output behaviour. This measure is more accurate than the simple number of gates (Ercegovac et. al., 1999). For response time estimation purposes, the circuit is viewed as a pipeline. Each stage in this pipeline includes a set of gates. We approximate the propagation delay of the output signals by the maximum delay imposed by the pipeline. The number of gate equivalents and output signal propagation delay for each kind of gate are given in Table 1. The data were taken from (Ercegovac et. al., 1999). Note that only 2-input gates NOT, AND, OR, XOR, NAND, NOR, XNOR and 2:1-MUX are allowed. The power consumption of a circuit is computed using a rough estimation of the circuit switching activity (Hsiao, 1999, Devadas, 1992, Monteiro, 1997, Najm, 1994). Details of how the multiple objectives of the optimisation are evaluated are given in Section 4.

Table 1. Node operators together with the corresponding symbol, size and delay

| Name | Gate Equivalent | Delay |
|------|-----------------|--------|
| NOT | 1 | 0.0625 |
| AND | 2 | 0.2090 |
| OR | 2 | 0.2160 |
| XOR | 3 | 0.2120 |
| NAND | 1 | 0.1300 |
| NOR | 1 | 0.1560 |
| XNOR | 3 | 0.2110 |
| MUX | 3 | 0.2120 |

3. SPEA: STRENGTH PARETO EVOLUTIONARY ALGORITHMS

In single-objective optimisation, the concept of optimality is clear. The optimal solution is the one that satisfies all the imposed constraints and optimise the unique objective. In multi-objective optimisation, however, that concept is not that obvious. In the section, we give the necessary definitions and the inherent terminology to formalise the concept of optimality within a multi-objective optimisation problem.

The multi-objective problem constraints define the feasible region \mathcal{F} and any point $S \in \mathcal{F}$ defines a *feasible* solution. The set of feasible solutions \mathcal{F} is mapped into the set \mathcal{Z} which is the set of all possible values of the objective functions. Normally, we never have a situation in which all the objective functions $f_i(S)$ values have an optimum at a common point $S \in \mathcal{F}$. Therefore, we have to establish certain criteria to determine what would be considered an optimal solution. The usual interpretation of the term optimum in multi-objective optimisation is the *Pareto* optimum that was first proposed by Francis Y. Edgeworth (Edgeworth, 1967) and later generalised by Vilfredo Pareto (Pareto, 1896).

Definition 1. A circuit design D_1 *dominates* another design D_2 , denoted by $D_1 \succ D_2$ and interchangeably design D_2 is *dominated* by design D_1 if and only if D_1 is no worse than D_2 with respect to all objectives, i.e. $area(D_1) \leq area(D_2)$, $time(D_1) \leq time(D_2)$ and $power(D_1) \leq power(D_2)$, and D_1 is strictly better than D_2 in at least one objective, i.e. $area(D_1) < area(D_2)$, $time(D_1) < time(D_2)$ or $power(D_1) < power(D_2)$. Otherwise, design D_1 does not dominate solution D_2 and interchangeably D_2 is not dominated by D_1 .

Definition 2. A circuit design $D \in \mathcal{F}$ is *Pareto optimal* if and only if there exists no other design $D' \in \mathcal{F}$ such that $D' \succ D$.

The strength Pareto genetic approach (SPEA) was proposed by Zitzler and Thiele (Zitzler and Thiele, 1999). It uses the Pareto dominance to preserve the population diversity. Besides the

generational population, it maintains an external continuously updated population. This population is kept up-to-date with evolved solutions that are non-dominated within the generational population. When the number of non-dominated solutions, which are stored externally, exceeds a pre-specified size, a clustering is applied to prune the population. The fitness evaluation of individuals is done considering only the individuals of the external population. The individuals selected to participate of the reproduction process are drawn from both the generational and external populations. SPEA applies a new sharing method to compute the fitness of individuals that appear in the same niche. The SPEA proceeds as described in Algorithm 1.

In Algorithm 1, function *Initialise(P)* allows the building of an initial population in *P*, function *Prune(PS, Psize)* prunes the Pareto set *PS* reducing its size down to *Psize*. It does so using a clustering procedure, which is explained later in this section and function *Fitness(P, PS)* computes the fitness of the individuals in *P* and *PS*. The tournament selection is used.

Algorithm 1. SPEA procedure

Input. Tournament size *Tsize* and generation number *Gsize*

Output. Pareto set *PS*

1. Initialise(*P*);
 2. *generation* := 1;
 3. *PS* := \emptyset ;
 4. do
 5. *PS* := {*s* | $\exists r \in P, s \succ r$ };
 6. *PS* := *PS* \ {*s* | $\exists r \in PS, r \succ s$ };
 7. if |*PS*| < *Psize* then Prune(*PS*);
 8. Fitness(*P, PS*);
 9. *generation* := *generation* + 1;
 9. Mutation(Crossover(Select(*P, PS, Tsize*)));
 10. while *generation* \neq *Gsize*;
 11. return *PS*;
- End.

The clustering performed by SPEA uses cluster analysis (Morse, 1980) and (Rosenman and Gero, 1985). This computation is described in Algorithm 2. In this algorithm, Δ_{l_1, l_2} represents the distance between clusters l_1 and l_2 , which is computed as the average distance between all the possible pairs of individuals across the two clusters. The measure $\delta_{i,j}$ is the Euclidean distance between individuals i and j (Horn et. al., 1994). The computation in line 7 of the clustering algorithm allows the construction of the reduced Pareto set by selecting from each cluster a representative individual, i.e. that with minimal average distance to all other individuals within the cluster.

Algorithm 2. SPEA clustering procedure - *Prune*

Input. Pareto set *S*, target size *Psize*

Output. Pruned Pareto set *P*

1. $L := \bigcup_{s \in S} \{s\}$; *P* := \emptyset ;
 2. while |*L*| > *Psize* do
 3. for each pair of clusters (l_1, l_2) $L \times L$ do
 4. $\Delta_{l_1, l_2} := \frac{1}{|l_1| + |l_2|} \sum_{(s_1, s_2) \in l_1 \times l_2} \delta_{s_1, s_2}$;
 5. $L = L \setminus \{l_1, l_2\} \cup \{l_1 \cup l_2\}$ | $\Delta_{l_1, l_2} = \min_{(x, y) \in L \times L} \Delta_{x, y}$;
 6. for each cluster $s \in C$ do
 7. $P := P \cup \{s\}$ | $\frac{\sum_{x \in l} \delta_{x, s}}{|l|} = \min_{y \in l} \left(\frac{\sum_{z \in l} \delta_{y, z}}{|l|} \right)$
 8. return *P*;
- End.

A central concept to the way SPEA evaluates individual fitness consists of the so-called individual *strength*. This is defined only for the individuals that are part of the Pareto set (external population). It is defined as in Definition 3.

Definition 3. The *strength* of an individual S in the Pareto set P is the non-negative real number in $[0,1)$, proportional to the number, say N , of individuals in P that are dominated by S , defined as below in Equation (1).

$$\theta_S = \frac{N}{|P|+1} \quad (1)$$

The fitness evaluation in SPEA is computed for all the individuals including those in the generational and Pareto set. There is a difference, however. The individuals in the external population are ranked using the corresponding strength while those that belong to the generational population are given a specific fitness value instead. The fitness of an individual is obtained summing up the strengths of all the individuals in the Pareto set that dominates it and adding 1 so that solutions in the external population will always have a better fitness. This is because we assume a minimisation problem and so individuals with smaller fitness are better. Function *Fitness* is described in Algorithm 3.

Algorithm 3. SPEA fitness evaluation procedure - *Fitness*

Input. Population P , Pareto set PS

Output. Fitness F

1. for each individual $S \in PS$ do
 2. $F[S] := \theta_S$;
 3. for each individual $S \in P$ do
 4. $F[S] := 1$;
 5. for each individual $R \in PS \mid R \succ S$ do
 6. $F[S] := F[S] + \theta_R$;
 7. $F[S] := F[S] + 1$;
 8. return F ;
- End.

4. EVOLVING PARETO-OPTIMAL DIGITAL CIRCUITS

In general, two main important concepts are crucial to any evolutionary computation: individual encoding and fitness evaluation. In evolutionary multi-objective optimisation, individual fitness is understood as its dominance regarding the solutions of the pool. Considering Definition 1 and Definition 2, one needs to know how to appreciate the solutions with respect to each one of the multiple objectives. So In the section, we concentrate on these two aspects for evolving Pareto-optimal digital circuits.

4.1. Circuit Encoding

We encode circuit schematics using a matrix of cells that may be interconnected. A cell may or may not be involved in the circuit schematics. A cell consists of two inputs or three in the case of a MUX, a logical gate and a single output. A cell may draw its input signals from the output signals of gates of previous rows. The gates includes in the first row draw their inputs from the circuit global input signal or their complements. The circuit global output signals are the output signals of the gates in the last row of the matrix. A chromosome with respect to this encoding is given in Figure 1.

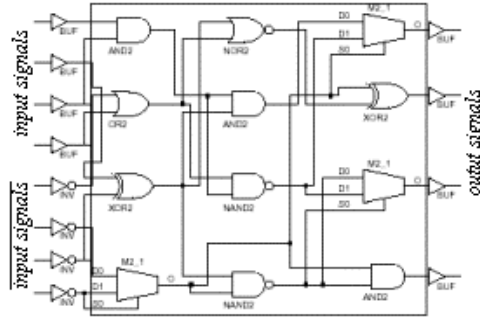


Figure 1. Digital circuit encoding

Crossover of circuit schematics, as for specification crossover, is implemented using a variable four-point crossover. The mutation operator can act on two different levels: gate mutation or route mutation. In the first case, a cell is randomised and the corresponding gate changed. When a 2-input gate is altered by another 2-input gate, the mutation is thus completed. However, when a 2-input gate is changed to a 3-input gate (i.e. to a MUX), the mutation operator randomise an additional signal among those allowed (i.e. all the input signals, their complements and all the output signals of the cells in the rows previous). Finally, when a MUX is mutated to a 2-input gate, the selection signal is simply eliminated. The second case, which consists of a route mutation, is quite simple. As before, a cell is randomised and one of its input signals is chosen randomly and mutated using another allowed signal. (For more details on the genetic operators used see (Nedjah and Mourelle, 2004a).)

4.2. Circuit Evaluation

As introduced, we aim at evolving Pareto-optimal digital circuit regarding four objectives: soundness, hardware area, response time and power dissipation. In order to appreciate the qualities of an evolved circuit with respect to the optimisation objectives, let C be a digital circuit that uses a subset (or the complete set) of the gates given in Table 1. Let $gates(C)$ be a function that returns the set of all gates of C . On the other hand, let $value(T)$ be the Boolean value that C propagates for the input Boolean vector T assuming that the size of T coincides with the number of input signal required for C . The soundness of circuit C is evaluated as described in Equation (2), wherein X represents the input values of the input signals while Y represents the expected output values of the output signals of C , n denotes the number of output signals that C has. Function *soundness* allows us to determine how much an evolved circuit adheres to the prescribed input/output behaviour. For each difference between the evolved circuit output values and the expected output signals, penalty ξ is accumulated. Note that the smaller the returned value the sounder is the considered circuit. Thus, for sound circuits, (i.e. those that implement the specified input/output behaviour) function *soundness* returns 0. Note that objective *soundness* is not negotiable, i.e. only sound individuals are considered as solutions.

$$soundness(C) = \sum_{j=1}^n \left(\sum_{i | value(X_i) \neq Y_{i,j}} \xi \right) \quad (2)$$

The hardware area required for the implementation of circuit C is evaluated as described in Equation (3), wherein function $gates(C)$ returns the set of all gates of circuit C while function $gatequiv(g)$ provides the number of gate equivalent to implement gate g .

$$area(C) = \sum_{g \in gates(C)} gatequiv(g) \quad (3)$$

The response time of circuit C is given in Equation (4) $levels(C)$ be a function that returns the set of gates of C grouped by level. Levels are ordered. Notice that the number of levels of a circuit coincides with the cardinality of the list expected from function $levels$. Function $delay$ returns the propagation delay of a given gate as shown in Table 1.

$$time(C) = \sum_{l \in levels(C)} \max_{g \in l} delay(g) \quad (4)$$

Unlike soundness, area and response time, the evaluation of the dissipated power by a given digital circuit is not simple as it depends on the values of the input signals. Several estimation model were elaborated (Devadas, 1992) (Najm, 1994). The average power dissipation for a digital circuit C is given in Equation (5) wherein V_{dd} is the supply voltage, T is the global clock period, $transitions(g)$ represents the times the output of gate g switches from 0 to 1 and vice versa and $capacitance(g)$ represents the output or load capacitance of gate g .

$$power = \frac{V_{dd}^2}{2 \times T} \times \sum_{g \in gates(C)} |transitions(g)| \times capacitance(g) \quad (5)$$

In the Equation (5), the first term of the product is the same for all circuits so can be discarded. Thus the average power can be represented by the summation term. The Switching activity at gate g , which determines $transitions(g)$, depends on the input signal changes. To avoid this dependency, we enumerate all possible transition times for each gate. A gate output switches each time one of its input signals switch. For a two-input gate g , if its input signals switch at times $\{t_1^{(1)}, t_1^{(2)}, \dots, t_1^{(m)}\}$ and $\{t_2^{(1)}, t_2^{(2)}, \dots, t_2^{(n)}\}$ respectively, then the output signal of gate g will switch at times $\{t_1^{(1)} + delay(g), t_1^{(2)} + delay(g), \dots, t_1^{(m)} + delay(g)\} \cup \{t_2^{(1)} + delay(g), t_2^{(2)} + delay(g), \dots, t_2^{(n)} + delay(g)\}$. Note that the primary inputs are supposed to switch only once during a given cycle period. So assuming that these input signals all switch at time 0, consequently gate g at the first level of the circuit will switch only once at time $delay(g)$. For the sake of practicality and without loss of generality, we assumed that the load capacitance of a gate by the corresponding number of fanouts.

5. PERFORMANCE RESULTS

In this section, we compare the Pareto-optimal evolutionary circuits yield by our multi-objective optimisation to those designed by a human as well as to those evolved by single objective genetic algorithms (Coelho et. al., 2001, Nedjah and Mourelle, 2004b).

The truth table of the first benchmark is given in Table 2. It has three-bit input signal $X = \langle x_2 x_1 x_0 \rangle$ and propagates a single-bit output signal Y .

Table 2. Truth table of example one

| x_2 | x_1 | x_0 | Y |
|-------|-------|-------|-----|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

We were able to evolve several Pareto-optimal digital circuit. The specifications if some of these circuits are given by the signal assignments of Equations (6), (7) and (8).

$$Y \Leftarrow ((x_0 \text{ nand } x_1) \text{ xor } x_2) \text{ nor } (x_1 \text{ nor } x_0) \quad (6)$$

$$Y \Leftarrow (x_1 \text{ nand } x_0) \text{ xor } (\text{not } x_2 \text{ nor } (x_0 \text{ nor } x_1)) \quad (7)$$

$$Y \Leftarrow \text{mux}(x_0, \text{not } x_2 \text{ nor } x_0, x_2 \text{ xnor } x_1) \quad (8)$$

Both the second and third benchmarks need a 4-bit input signal $X = \langle x_3x_2x_1x_0 \rangle$ and yield a single-bit output signal Y . The fourth and fifth benchmarks also require a 4-bit input signal X but the respective circuits propagate a 4-bit and 3-bit output signal respectively. The truth tables of the four benchmarks are summarised in Table 3. Note that the fourth benchmark is a simple 2-bit multiplier of $X = \langle x_3x_2 \rangle$ times $Y = \langle x_1x_0 \rangle$. (The notation of the input signals is purely for space sake!)

For the second benchmark as for the first, we were able to evolve several Pareto-optimal digital circuits. The specifications if some of these circuits are given by the signal assignments of Equations (9) and (10).

$$Y \Leftarrow ((x_2 \text{ and } x_3) \text{ and } x_0) \text{ xnor } ((x_1 \text{ xnor } x_2) \text{ nor } (x_2 \text{ or } x_3)) \quad (9)$$

$$Y \Leftarrow ((x_2 \text{ nand } x_0) \text{ xnor } x_1) \text{ xnor } \text{mux}(x_1, x_0 \text{ nand } x_3, x_3 \text{ nand } x_2) \quad (10)$$

For the third benchmark, we were able to evolve several Pareto-optimal digital circuits. The specifications if some of these circuits are given by the signal assignments of Equations (11), (12) and (13).

$$Y \Leftarrow \text{mux}(x_0 \text{ xnor } x_3, x_1, x_2) \text{ or } (x_1 \text{ nor } x_3) \quad (11)$$

$$Y \Leftarrow (x_0 \text{ nor } x_3) \text{ or } ((x_1 \text{ nor } x_3) \text{ or } \text{mux}(x_0 \text{ and } x_3, x_1, x_2)) \quad (12)$$

$$Y \Leftarrow \text{mux}(\text{not } x_0 \text{ nor } \text{not } x_1) \text{ nand } \text{not } x_2, \text{mux}(x_0, x_1, x_2), x_3) \quad (13)$$

Table 3. Truth tables of example 2, example 3, example 4 and example 5 respectively

| Input | | | | Examples 2, 3 | | Example 4 | | | | Example 5 | | |
|-------|-------|-------|-------|---------------|---|-----------|-------|-------|-------|-----------|-------|-------|
| X_3 | X_2 | X_1 | X_0 | Y | Y | P_3 | P_2 | P_1 | P_0 | Y_2 | Y_1 | Y_0 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |

For the fourth benchmark, we were able to evolve several Pareto-optimal digital circuits. The specifications if some of these circuits are given by the signal assignments of Equations (14) and (15).

$$\begin{aligned}
 y_3 &\Leftarrow (x_2 \text{ and } x_0) \text{ and } (x_3 \text{ and } x_1) \\
 y_2 &\Leftarrow (x_3 \text{ and } x_1) \text{ and } (x_0 \text{ nand } x_2) \\
 (14) \\
 y_1 &\Leftarrow (x_3 \text{ nand } x_0) \text{ xor } (x_2 \text{ nand } x_1) \\
 y_0 &\Leftarrow (x_2 \text{ and } x_0) \text{ or } x_0
 \end{aligned}$$

$$\begin{aligned}
 y_3 &\Leftarrow (x_3 \text{ nand } (\text{not } x_2 \text{ nor not } x_1)) \text{ nor not } x_0 \\
 y_2 &\Leftarrow x_2 \text{ and } (x_3 \text{ nand } (\text{not } x_2 \text{ nor not } x_1) \text{ and } x_0) \\
 y_1 &\Leftarrow (\text{not } x_3 \text{ nor not } x_0) \text{ xor } (\text{not } x_2 \text{ nor not } x_1) \\
 (15) \\
 y_0 &\Leftarrow x_3 \text{ and } x_1
 \end{aligned}$$

For the fifth benchmark, we were able to evolve several Pareto-optimal digital circuits. The specifications if some of these circuits are given by the signal assignments of Equations (16) and (17).

$$\begin{aligned}
 y_2 &\Leftarrow (x_0 \text{ or not } x_2) \text{ and } ((\text{not } x_0 \text{ or } x_2) \text{ nand } (x_1 \text{ nand not } x_3)) \\
 y_1 &\Leftarrow (\text{not } x_0 \text{ or } x_2) \text{ and } ((x_0 \text{ or not } x_2) \text{ nand } (x_1 \text{ or not } x_3)) \\
 (16) \\
 y_0 &\Leftarrow ((\text{not } x_0 \text{ or } x_2) \text{ nand } (x_1 \text{ nand not } x_3)) \text{ nor } ((x_0) \text{ or not } x_2) \text{ nand } (x_{\{1\}} \text{ or not } x_3)
 \end{aligned}$$

$$\begin{aligned}
 y_2 &\Leftarrow (x_0 \text{ or not } x_2) \text{ and } ((\text{not } x_0 \text{ or } x_2) \text{ nand } (x_1 \text{ nand not } x_3)) \\
 y_1 &\Leftarrow (\text{not } x_0 \text{ or } x_2) \text{ and } ((x_0 \text{ or not } x_2) \text{ nand } (x_1 \text{ or not } x_3)) \\
 (17) \\
 y_0 &\Leftarrow (x_0 \text{ or not } x_2) \text{ and } (x_1 \text{ or not } x_3)
 \end{aligned}$$

Table 4 shows a comparison between the fittest circuits engineered by a human designer, Coello's genetic algorithm and our genetic algorithm, which is based on genetic programming. For each benchmark, the required hardware area, the necessary propagation delay and the product *area*×*performance* are detailed. The graphical representation of these figures is shown in the chart of Figure 2.

Table 4. Numerical comparison of the three methods

| | area | | | Delay | | | power | | |
|------------------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|
| | <i>POH</i> | <i>CGA</i> | <i>CDM</i> | <i>POH</i> | <i>CGA</i> | <i>CDM</i> | <i>POH</i> | <i>CGA</i> | <i>CDM</i> |
| 1st. | 6 | 9 | 12 | 0.524 | 0.637 | 0.637 | 4 | 7 | 9 |
| 2nd. | 13 | 16 | 20 | 0.783 | 0.918 | 0.702 | 11 | 17 | 26 |
| 3rd. | 9 | 15 | 20 | 0.639 | 0.699 | 0.912 | 7 | 13 | 23 |
| 4th. | 16 | 16 | 24 | 0.425 | 0.842 | 0.853 | 13 | 11 | 15 |
| 5th. | 17 | 21 | 34 | 0.685 | 1.065 | 0.703 | 12 | 42 | 22 |

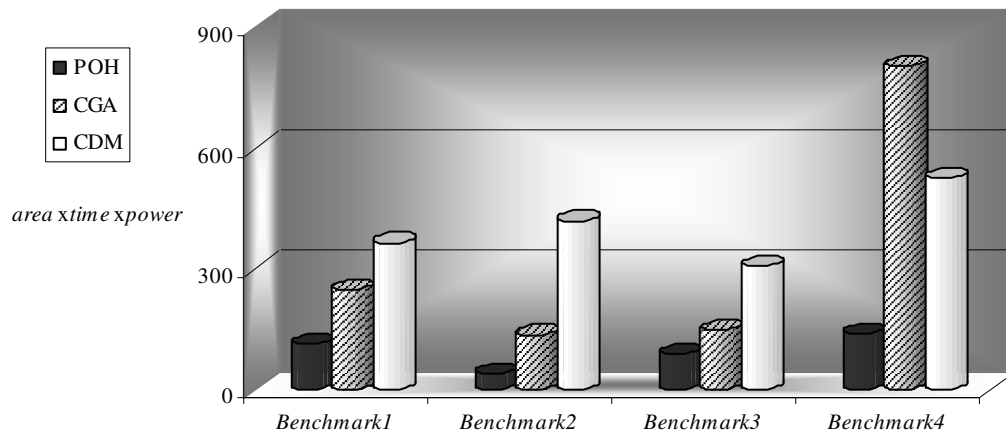


Figure 2. Comparison of the circuit characteristics

7. CONCLUSIONS

In this paper, we designed Pareto-optimal innovative evolutionary digital circuits. The produced circuits are balanced in the sense that they exhibit the best trade-off between the required hardware area, propagation time of output signals and power dissipation. We did so exploiting the strength Pareto evolutionary algorithm which is the most recent and efficient multi-objective evolutionary algorithm. The Pareto-optimal circuits obtained for a set of benchmarks present the best $area \times time \times power$ factor when compared to both circuits that were designed using conventional methods as well as to those genetically evolved by Coello's in (Coelho et. al., 2001).

The big majority of the circuits we evolved, dominates the ones that were compared with and the rest of the circuits is not dominated by neither of the circuits in comparison. A future work consists in performing a massive evolution for the used benchmarks which should yield all Pareto-optimal circuits for each benchmark. This would allow us to identify the Pareto fronts and its 3D representations.

ACKNOWLEDGMENTS

The authors would like to acknowledge the valuable support of both CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) and FAPERJ (Fundação de Amparo à Pesquisa no Estado do Rio de Janeiro).

REFERENCES

- Coello, C.A. (2001), *A Short Tutorial on Evolutionary Multi-objective Optimisation*, Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimisation.
- Coelho, A.C., Christiansen, A.D. and Aguirre, A.H. (2001), "Towards Automated Evolutionary Design of Combinational Circuits", *Comput. Electr. Eng.*, **27**, 1–28.
- Devadas, S., Keutzer, K. and White, J. (1992), "Estimation Of Power Dissipation in CMOS Combinational Circuits Using Boolean Function Manipulation", *IEEE Transactions on Computer-Aided Design*, **2**(3), 373–383.
- Edgeworth, F. Y. (1967), *Mathematical Psychics: An Essay on the Application of Mathematics to the Moral Sciences*, Augustus M. Kelley, New York. (Originally published in 1881)
- Ercegovac, M.D., Lang, T. and Moreno, J.H. (1999), *Introduction to Digital Systems*, John Wiley

- Horn, J., Nafpliotis, N. and Goldberg, D.E. (1994), *A Niche Pareto Genetic Algorithm for Multi-Objective Optimisation*, Proc. the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, **1**, 82–87, IEEE Press.
- Hsiao, M.S. (1999), *Peak Power Estimation Using Genetic Spot Optimization for Large VLSI Circuits*, Proceedings of European Conference on Design, Automation and Test, 175–179.
- Monteiro, J., D. Devadas, Gosh, A. Keutzer, K. and White J. (1997), “Estimation Of Average Switching Activity in Combinational Logic Circuits Using Symbolic Simulation”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **6**(1), 121–127.
- Morse, J.N. (1980), *Reducing the Size of the Non-Dominated Set: Pruning by Clustering*, Computers and Operations Research, **7**(1–2), 55–66.
- Najm, F. N. (1994), “A Survey of Power Estimation Techniques in VLSI Circuits”, *IEEE Transactions on VLSI Systems*, **2**(4), 446–455.
- Nedjah, N. and Mourelle, L.M. (2004), *Comparison of Two Encodings for Evolutionary Digital Circuit Design*, Proceedings of 17th. International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Lecture Notes in Computer Science, vol. 3029, 594-604.
- Nedjah, N. and Mourelle, L.M. (2004), *Real-World Evolutionary Designs: Secure Evolvable Hardware for Public-Key Cryptosystems*, In *Evolutionary Machine Design: Methodologies & Application*, in N. Nedjah and L.M. Mourelle (eds.), 111–142, Nova Science Publishers, New York.
- Pareto, V. (1896), *Cours d'Économie Politique*, volume I and II, F. Rouge, Lausanne.
- Rosenman, M.A. and Gero, J.S. (1985), *Reducing the Pareto Set in Multi-Criterion Optimisation*, Engineering Optimisation, **8**(3), 189–206, 1985.
- Zitzler, E. and Thiele L. (1999), *MultiObjective Evolutionary Algorithms: A Comparative Case Study and The Strength Pareto Approach*, IEEE Transactions on Evolutionary Computation, **3**(4), 257–271.