Automatic Design of Approximate Circuits by Means of Multi-Objective Evolutionary Algorithms

Radek Hrbacek
Brno University of Technology
Faculty of Information Technology
Czech Republic
ihrbacek@fit.vutbr.cz

Vojtech Mrazek
Brno University of Technology
Faculty of Information Technology
Czech Republic
imrazek@fit.vutbr.cz

Zdenek Vasicek
Brno University of Technology
Faculty of Information Technology
Czech Republic
vasicek@fit.vutbr.cz

Abstract—**Recently, power efficiency has become the most important parameter of many real circuits. At the same time, a wide range of applications capable of tolerating imperfections has spread out especially in multimedia. Approximate computing, an emerging paradigm, takes advantage of relaxed functional requirements to make computer systems more efficient in terms of energy consumption, speed or complexity. As a result, a variety of trade-offs between error and efficiency can be found. In this paper, a design method based on a multi-objective evolutionary algorithm is proposed. For a given circuit, the method is able to produce a set of Pareto optimal solutions in terms of the error, power consumption and delay. The proposed design method uses Cartesian Genetic Programming for the circuit representation and a modified NSGA-II algorithm for design space exploration. The method is used to design Pareto optimal approximate versions of arithmetic circuits such as multipliers and adders.**

I. Introduction

Approximate computing, an emerging paradigm, takes advantage of relaxed functional requirements to make computer systems more efficient in terms of energy consumption, computing speed or complexity. Error resilient applications can achieve significant savings while still serving their purpose with the same or a slightly degraded quality.

The complexity of computer systems is permanently growing and thus, automated design tools have to deal with more and more complex problems specified on higher level of abstraction than before. The same holds true for approximate computing. Even though new methods are emerging, there is a lack of methods for approximate computing offering a numerous set of trade-off solutions.

Evolutionary algorithms (EAs) have been confirmed to bring innovative solutions to complex problems. Recently, complex digital circuits have been optimized by means of EAs while the scalability of the methods has been improved substantially [5], [12]. Multi-objective EAs have been used to design simple approximate circuits from scratch [4].

In this paper, we propose an evolutionary based approach to design approximate circuits starting from a set of conventional fully working circuits. The method is evaluated in the task of approximate 8-bit adders and multipliers design.

II. Approximate Computing

Recently, power efficiency has become the most important parameter of almost every computing platform. At the same time, a wide range of applications capable of tolerating imperfections in computations has spread out. As a consequence, a new research field – *approximate computing* – has emerged to investigate how computer systems can be made more efficient in terms of energy consumption, computing speed or complexity assuming that some errors are acceptable. It has been believed, that significant savings can be achieved by relaxing the requirement of perfect functionality thanks to the *error resilience* of some applications. Therefore, the *accuracy* of the system can be used as a design metric and inaccurate solutions can be accepted if an improvement in other parameters occurs.

The approximation can be introduced at various levels including the entire computer system architecture [6], particular components (e.g. ALU) [3], operating system, algorithm or even programming language [1]. As the complexity of today's computer systems grows, manual approximation is not an efficient design method. Hence, several automated approximate design methods have been introduced. The design of approximate circuits is typically based on modifying fully functional circuits.

The Systematic methodology for Automatic Logic Synthesis (SALSA) uses a quality function which decides whether a predefined quality constraint is met or not. The algorithm is allowed to modify the circuit as long as the quality constraint is not violated. SALSA has been applied to a number of problems, e.g. 32-bit adders, 8-bit multipliers, FIR filters, DCT blocks and others [18].

Another approach, Substitute-and-Simplify (SASIMI), looks for signal pairs having similar values with a high probability. By substituting one signal for the other, a part of the circuit can be removed resulting in area and power savings at the cost of an error introduced to the output. Moreover, SASIMI further extends the approach to synthesize quality configurable circuits, where at runtime, processing of selected input vectors is given an additional cycle to correct errors due to approximations [17].

Unlike the aforementioned methods, ABACUS (Automated

Behavioral Approximate Circuit Synthesis) operates directly on the behavioral descriptions of circuits. ABACUS automatically generates approximate circuits from input behavioral descriptions by performing global transformations on an abstract synthesis tree (AST) created from the behavioral description. The outcome approximate circuits are still expressed in behavioral code and can be synthesized by means of standard synthesis tools. Complementary approximate computing methods, e.g. voltage over-scaling or manually created approximate components, may be still used [9].

Although most of the design methods deal with combinational circuits, there are methods capable of approximating sequential circuits. As an example, the Automatic Methodology for Sequential Logic Approximation (ASLAN) creates an approximate version of a sequential circuit that consumes lower energy, while meeting a specified quality constraint. ASLAN identifies combinational block in the sequential circuit that are amenable to approximation and iteratively approximates the entire sequential circuit using a gradient-descent approach [10].

III. Evolutionary Design and Optimization

In our previous work, we used evolutionary algorithms to either design digital circuits from scratch [5] or to optimize existing circuits [12]. Recently, the evolutionary approach has been applied in the task of approximate circuits design with respect to multiple objectives [4].

A. Cartesian Genetic Programming

The proposed method is based on CGP, in which a circuit is represented as a fixed-sized cartesian grid of $N_\mathrm{r} \times N_\mathrm{c}$ nodes interconnected by a feed-forward network (see Figure 1). Node inputs can be connected either to one of N_i primary inputs or to an output of a node in preceding L columns. Each node has a fixed number of inputs N_ni and outputs N_no and can perform one of the functions from the set Γ. Each of N_o primary circuit outputs can be connected either to a primary input or to a node's output. The area and delay of the circuit can be constrained by changing the grid size and the L-back parameter.

The genotype is of fixed length, whereas the phenotype is of variable length depending on the number of inactive nodes, i.e. nodes whose output is not used by any other node or primary output (see Figure 2). This implies the existence of individuals with different genotypes but the same phenotypes, which is usually referred to as neutrality. It was shown that for certain problems the neutrality significantly reduces the computational effort and helps to find more innovative solutions [7].

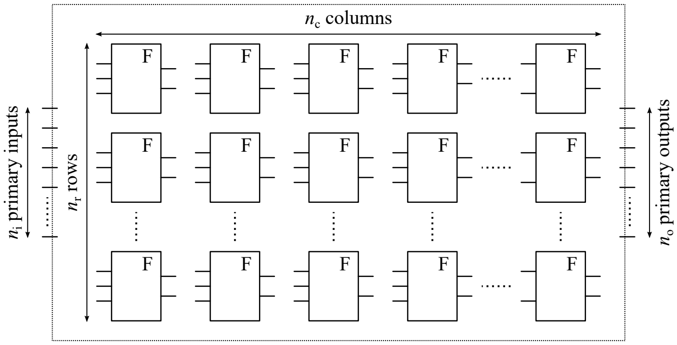

Fig. 1. Cartesian Genetic Programming.

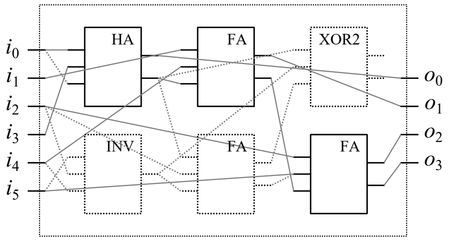

Fig. 2. Example of a CGP representation of 3-bit ripple-carry adder.

Standard (single-objective) CGP uses a simple mutation based $(1 + \lambda)$ evolutionary strategy as a search mechanism, the population size $1 + \lambda$ is mostly very small, typically, $\lambda = 4$. The initial population is constructed either randomly (evolutionary design) or by mapping of a known solution to the CGP chromosome (evolutionary optimization). In each generation, the best individual is passed to the next generation unmodified along with its λ offspring individuals created by means of point mutation operator. In case more individuals with the best fitness exist, a randomly selected one is chosen. The mutation rate m is usually set to modify up to 5 % randomly selected genes.

B. Multi-Objective CGP

Unlike the single-objective optimization, which enables to compare any two candidate solutions and decide which one is better, the multi-objective optimization leads to the existence of a whole range of trade-off solutions, if the objectives are conflicting. In the case of digital circuits design, the better the circuit works, the larger area and power consumption it has.

Many multi-objective evolutionary algorithms have been proposed, most of them are based on the idea of *Pareto dominance*. The solution p *dominates* the solution q if p is no worse than q in all objectives and p is strictly better than q in at least one objective. The Pareto optimal solutions are

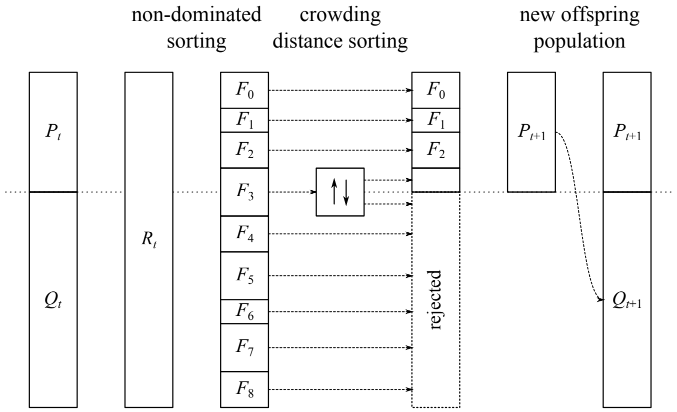

Fig. 3. Non-dominated Sorting Genetic Algorithm II.

not dominated by any other solutions and form the so called *Pareto front*.

One of the most popular multi-objective evolutionary algorithms is the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [2]. It is based on sorting individuals according to the dominance relation into multiple fronts. The first front F_0 contains all Pareto optimal solutions. Each subsequent front F_i is constructed by removing all the preceding fronts from the population and finding a new Pareto front. Each solution is assigned a *rank* according to the front it belongs to, the solutions from the front F_i have the rank equal to i. The NSGA-II fast non-dominated sort is very efficient, the overall complexity is $\mathcal{O}(MN^2)$, where N is the population size and M is the number of objectives. The principle of the algorithm can be seen in Figure 3. NSGA-II was recently applied to design approximate digital circuits from scratch, the convergence of the method was improved by using multiple islands [4]. The multi-objective approach was compared to the single-objective CGP in the task of approximate circuits design, however, the estimation of power consumption and delay of the circuits was rough [15].

C. Function set

Since the goal of this paper is to optimize the circuits as much as possible, we use a subset of functions from a 180nm technology process library. The function cells have one, two or three inputs (e.g. full adder) and one or two outputs. Complete list of the functions including their area and leakage power can be found in Table I.

Function	Description	Area [μm^2]	Leakage power [nW]
BUF	Buffer (2x/4x)	24/32	0.066/0.113
INV	Inverter (1x/2x/4x/8x)	16/16/24/40	0.022/0.036/0.073/0.147
AND2	2-input AND (1x/2x)	32/32	0.075/0.090
OR2	2-input OR (1x/2x)	32/32	0.075/0.090
XOR2	2-input XOR (1x)	56	0.161
NAND2	2-input NAND (1x)	24	0.039
NOR2	2-input NOR (1x)	24	0.035
XNOR2	2-input XNOR (1x)	56	0.161
NAND3	3-input NAND (1x)	36	0.056
NOR3	3-input NOR (1x)	64	0.055
MUX2	Multiplexor (1x)	48	0.087
AOI21	3-input AND/NOR (1x)	32	0.052
OAI21	3-input OR/NAND (1x)	23	0.048
FA	Full adder (1x)	120	0.231
HA	Half adder (1x)	80	0.161

TABLE I
LIST OF USED FUNCTION CELLS.

Some of the functions (e.g. BUF, INV) have multiple sizes which differ in the maximum output load, area, power consumption and delay. During the evaluation, proper size is selected depending on the output load of the gate. The dynamic power and delay of the gates depend on the output load as well.

D. Output Error

In the case of digital circuit evolution, the output error of the candidate circuit is often measured as the number of correct output bits compared to a specified truth table (i.e. the Hamming distance). In order to obtain a fully working circuit, 2^{N_i} test vectors have to be evaluated so as to compute the fitness value. It can be sped up by applying parallelism at various levels [5] or by introducing formal methods, e.g. SAT solvers [12] or Binary Decision Diagrams (BDD) [14].

In the case of approximate circuits, Hamming distance is often not suitable. Instead, metrics based on the arithmetical distance, such as the worst case error, mean absolute error, relative error or others are usually used [13]. In this paper, we use the mean relative error:

$$f_\mathrm{mre} := \frac{\sum_{\forall i} \frac{\left|O_\mathrm{orig}^{(i)} - O_\mathrm{approx}^{(i)}\right|}{\max(1, O_\mathrm{orig}^{(i)})}}{2^{N_\mathrm{i}}}, \qquad (1)$$

where $O_\mathrm{orig}^{(i)}$ is the decimal representation of the i-th circuit correct output and $O_\mathrm{approx}^{(i)}$ is the individual's i-th output. In addition to that, we constrain the worst absolute and relative errors.

E. Power Estimation

In order to estimate the power consumption of a candidate circuit, we propose to use a method based on the switching activity.

The power consumption of digital circuits can be divided into two different parts: dynamic and static power components. The first one occurs every time the output of a gate changes its logic value. In fact a low resistance path between the power rails is created during switching. Static power consumption is caused mainly by the leakage current which exists even when the circuit is in a stable state, i.e. not switching. Although the static power component has always been present, it has gained importance in sub-micrometer and nanometer devices [20].

Thus, the total power consumption has to be optimized by reducing static as well as dynamic part of the power consumption.

The power consumption $P = P_\mathrm{s} + P_\mathrm{d}$ of a candidate circuit is calculated as follows. Because the static part of the power consumption depends only on a function of a logic gate, the total static power consumption P_s can be obtained by summing static leakage for all gates of the candidate circuits. The leakage of each gate is defined by the technology specification file (so-called liberty file) for the target technology. The dynamic part P_d is defined as follows:

$$P_d = 0.5 \times C_\mathrm{load} \times V_\mathrm{dd}^2 \times f \times E(\mathrm{transitions}), \qquad (2)$$

where C_load is the total load capacitance of the output (i.e. the sum of all input capacitances of the connected inputs defined in the liberty file), V_dd is the supply voltage, f is target frequency and $E(\mathrm{transitions})$ is the expected value of the output transitions per global clock cycle (switching activity) [8].

We have used zero-delay model, i.e. glitches are not considered. Thus, the switching activity can be obtained using simulation of all input vectors, which is done during the function verification. Total switching activity of a gate is calculated as follows:

$$E(\text{transitions}) = 2 \cdot (p_0 \cdot p_1) = 2 \cdot p_1 \cdot (1 - p_1), \quad (3)$$

where p_0 is probability that output of a considered gate is equal to logical zero, similarly p_1 is probability that the output is equal to logical one. There are more ways to determine transition probabilities. The simplest approach is to use the simulation and count the number of cases for which the output value was equal to 1.

F. Propagation Delay Estimation

The delay of a candidate circuit is calculated using the parameters defined in the liberty timing file available for the utilized semiconductor technology. The delay t_d of a cell c_i is modeled as a function of its input transition time t_s and capacitive load C_l on the output of the cell, i.e. $t_\mathrm{d}(c_i) = f(t_\mathrm{s}^{c_i}, C_\mathrm{l}^{c_i})$. The delay of the circuit C is determined as the delay of the longest path:

$$\mathrm{Delay}(C) = \max_{\forall p \in \mathrm{path}} \sum_{c_i \in p} t_\mathrm{d}(c_i). \quad (4)$$

The capacitive load on the circuit outputs is chosen to be equal to the input capacitance of a buffer cell. The transition time on primary inputs corresponds to the transition time on the output of a buffer cell.

IV. Experimental Results

In this section, experiments regarding the multi-objective design of arithmetical circuits are presented. The method was evaluated in the task of approximate 8-bit adders and multipliers design. The CGP parameters were set as follows: 500 individuals in the population, 5000000 generations, 10 islands, mutation rate 5 %, number of rows $N_\mathrm{r} = 1$. The number of columns was $N_\mathrm{c} = 200$ in the case of the adders and $N_\mathrm{c} = 1000$ in the case of the multipliers.

The circuits were designed with respect to 3 objectives – the mean relative error (MRE), the power consumption of the circuit and the delay. The MRE was constrained to be at most 10 %, the worst case error was constrained to be at most 5 % of the output range and the worst case relative error was limited to 1000 %, i.e. all candidate solutions violating these requirements are discarded.

A. Initial population

In our previous research, we used random initial population to design simple digital circuits from scratch [5], [4]. For complex circuits, we seeded the initial population with a single known solution and optimized the circuit using CGP [12], [11], [15].

In this paper, we use a set of conventional circuits as the initial population. CGP chromosomes for 13 different adder and 6 different multiplier architectures were generated [19]. The power, area and delay estimates of those circuits can be found in Tables II, III. The adders include Ripple-Carry Adder (RCA), Carry-Select Adder (CSA), Carry-Lookahead Adder (CLA), multiple Tree Adder (TA) and Higher Valency Tree Adder (HVTA) architecures. The multipliers include Ripple-Carry Array, multiple Carry-Save Array and Wallace Tree architectures. All parameters in this section are related to the Ripple-Carry Adder and Ripple-Carry Array Multiplier architectures, since they are the most power efficient conventional architectures.

Architecture	Power	Area	Delay
Ripple-Carry Adder	**100.00 %**	**100.00 %**	**100.00 %**
Carry-Select Adder	201.18 %	174.78 %	61.15 %
Carry-Lookahead Adder	414.74 %	334.78 %	61.99 %
HVTA (Brent-Kung)	286.00 %	201.74 %	68.52 %
HVTA (Han-Carlson)	286.00 %	201.74 %	68.52 %
HVTA (Kogge-Stone)	371.48 %	257.39 %	59.77 %
HVTA (Sklansky)	305.07 %	215.65 %	60.45 %
TA (Brent-Kung)	282.99 %	201.74 %	67.25 %
TA (Han-Carlson)	295.74 %	212.17 %	61.87 %
TA (Knowles)	362.25 %	257.39 %	59.94 %
TA (Kogge-Stone)	342.20 %	243.48 %	57.68 %
TA (Ladner-Fischer)	282.99 %	201.74 %	67.25 %
TA (Sklansky)	298.34 %	212.17 %	57.84 %

TABLE II
Power, delay and area of various conventional 8-bit adders compared to Ripple-Carry Adder.

Architecture	Power	Area	Delay
Ripple-Carry Array	**100.00 %**	**100.00 %**	**100.00 %**
Carry-Save Array using RCA	102.30 %	100.00 %	71.16 %
Carry-Save Array using CSA	108.42 %	106.16 %	62.03 %
Wallace Tree using RCA	104.29 %	107.39 %	68.91 %
Wallace Tree using CLA	116.10 %	148.48 %	51.26 %
Wallace Tree using CSA	120.12 %	122.35 %	53.28 %

TABLE III
Power, delay and area of various conventional 8-bit multipliers compared to Ripple-Carry Array Multiplier.

B. Results

Figure 4 shows 473 Pareto optimal 8-bit approximate adders evolved from the initial population of 13 conventional adders. Parameters of 9 selected evolved circuits can be found in Table IV. It can be seen that the Ripple-Carry Adder is optimal in terms of power consumption among the conventional architectures, but significant savings can be achieved when relaxing

MRE	Power	Delay
0.000 %	244.78 %	38.92 %
0.135 %	89.81 %	79.93 %
0.273 %	85.99 %	99.73 %
0.396 %	79.08 %	96.29 %
0.678 %	71.89 %	73.06 %
0.942 %	61.70 %	59.59 %
1.918 %	47.66 %	46.12 %
2.939 %	35.97 %	33.92 %
4.280 %	33.39 %	33.92 %

TABLE IV
Parameters of evolved approximate 8-bit adders.

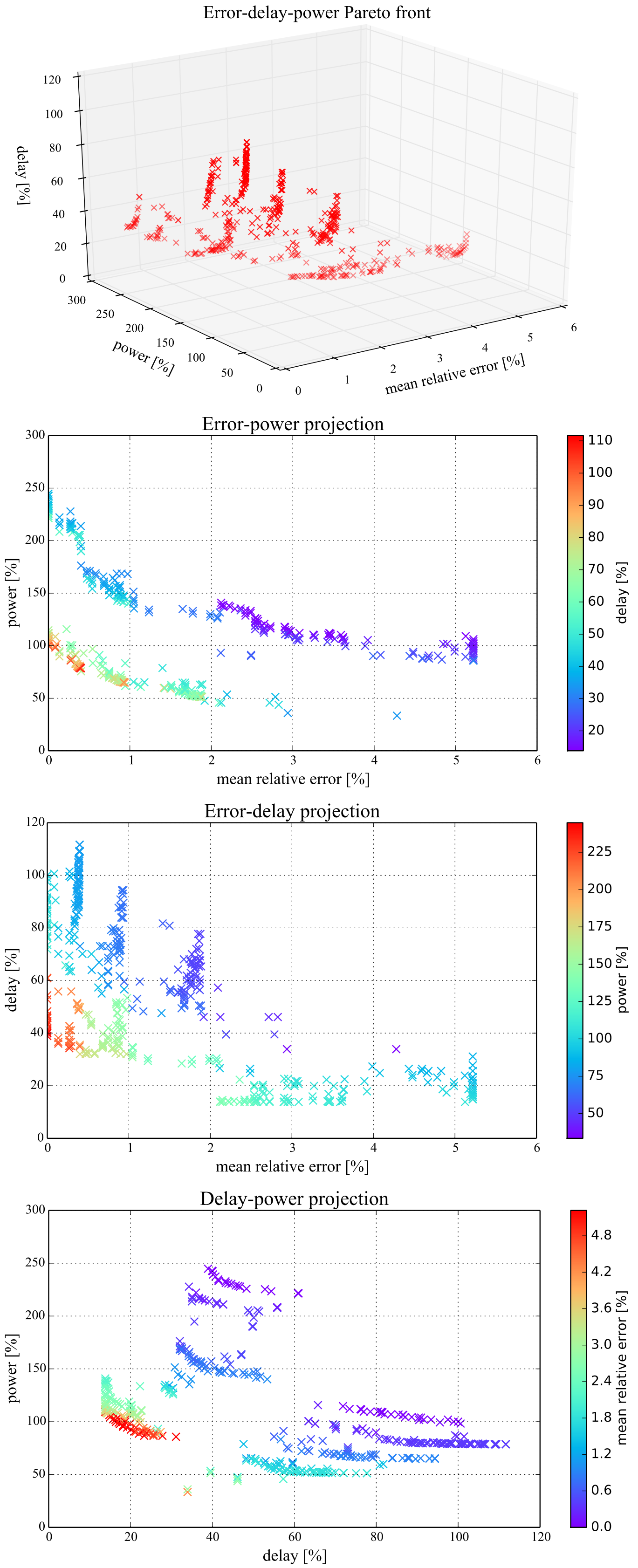

Fig. 4. Pareto front of evolved approximate 8-bit adders.

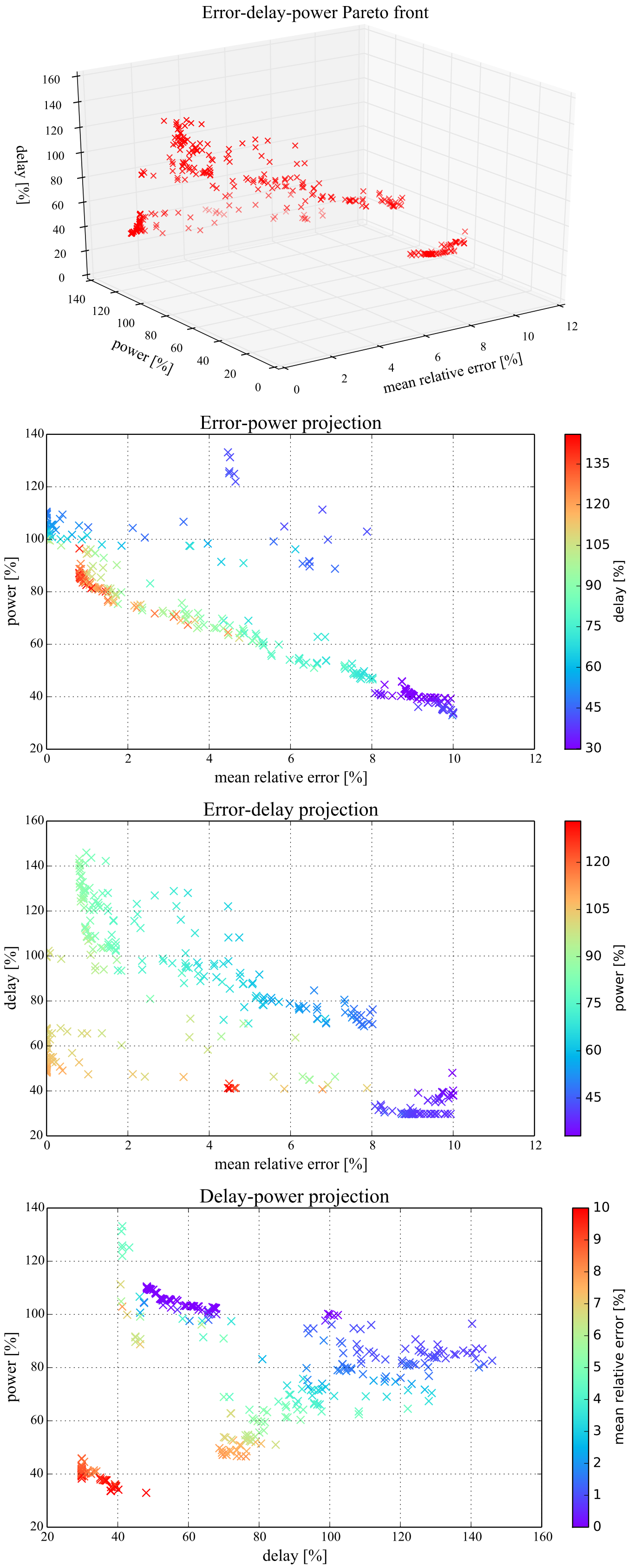

Fig. 5. Pareto front of evolved approximate 8-bit multipliers.

the requirement of perfect functionality. The delay of the Tree Adder with Sklansky architecture was overcome with multiple evolved circuits (at the cost of increasing the delay). The most efficient 8-bit adders have power consumption 33–36 % of the RCA with MRE of 3–4 %.

Similarly, Figure 5 shows 433 Pareto optimal 8-bit approximate multipliers that were evolved from 6 conventional circuits. Table V shows the parameters of 11 selected evolved multipliers. The Ripple-Carry Array Multiplier architecture was not overcome in terms of the power consumption when considering no error. The delay of Wallace Tree multipliers was improved to 48.23 % at the cost of a higher power consumption. The power savings are lower in comparison with the adders, for the same savings the error must be higher.

MRE	Power	Delay
0.000 %	110.52 %	48.23 %
0.813 %	88.70 %	130.51 %
0.951 %	83.69 %	113.02 %
1.511 %	76.15 %	120.06 %
3.092 %	71.61 %	96.79 %
4.177 %	66.19 %	90.54 %
5.334 %	59.66 %	80.60 %
6.579 %	51.01 %	84.70 %
8.218 %	40.98 %	33.94 %
10.000 %	33.74 %	38.02 %

TABLE V
PARAMETERS OF EVOLVED APPROXIMATE 8-BIT MULTIPLIERS.

V. CONCLUSIONS

Recently, complex digital circuits were optimized by means of evolutionary algorithms [12]. Both single-objective and multi-objective approaches were applied to design approximate circuits from scratch [16], [4].

In this paper, the multi-objective approach was improved by seeding the initial population with a set of conventional fully working circuits instead of starting with a single conventional circuit or a random initial population. The method uses CGP for circuit representation and NSGA-II algorithm to handle multiple objectives.

The proposed method was evaluated in the task of approximate 8-bit adders and multipliers design. The circuits were designed with respect to three objectives – mean relative error, power consumption and delay. Contrasted to previous work, the method was able to evolve hundreds of Pareto optimal circuits with significant power consumption savings.

In our future research, we will focus on increasing the scalability of the method in order to design complex circuits. For that purpose, the use of formal methods will be investigated.

ACKNOWLEDGMENTS

This work was supported by the Czech Science Foundation project 16-17538S, Brno University of Technology project FIT-S-14-2297 and the IT4Innovations excellence in science project (IT4I XS LQ1602).

REFERENCES

[1] M. Carbin, S. Misailovic, and M. C. Rinard. Verifying quantitative reliability for programs that execute on unreliable hardware. In *Proceedings of the 2013 ACM SIGPLAN International Conference on Object Oriented Programming Systems Languages &#38; Applications*, OOPSLA '13, pages 33–52, New York, NY, USA, 2013. ACM.

[2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii, Apr. 2002.

[3] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy. Low-power digital signal processing using approximate adders. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 32(1):124–137, Jan 2013.

[4] R. Hrbacek. Parallel multi-objective evolutionary design of approximate circuits. In *GECCO '15 Proceedings of the 2015 conference on Genetic and evolutionary computation*, pages 687–694. Association for Computing Machinery, 2015.

[5] R. Hrbacek and L. Sekanina. Towards highly optimized cartesian genetic programming: From sequential via simd and thread to massive parallel implementation. In *GECCO '14 Proceedings of the 2014 conference on Genetic and evolutionary computation*, pages 1015–1022. Association for Computing Machinery, 2014.

[6] S.-L. Lu. Speeding up processing with approximation circuits. *Computer*, 37(3):67–73, Mar 2004.

[7] J. Miller and S. Smith. Redundancy and computational efficiency in cartesian genetic programming. *Evolutionary Computation, IEEE Transactions on*, 10(2):167–174, 2006.

[8] J. Monteiro, S. Devadas, A. Ghosh, K. Keutzer, and J. White. Estimation of average switching activity in combinational logic circuits using symbolic simulation. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 16(1):121–127, Jan 1997.

[9] K. Nepal, Y. Li, R. Bahar, and S. Reda. Abacus: A technique for automated behavioral synthesis of approximate computing circuits. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, pages 1–6, March 2014.

[10] A. Ranjan, A. Raha, S. Venkataramani, K. Roy, and A. Raghunathan. Aslan: Synthesis of approximate sequential circuits. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, pages 1–6, March 2014.

[11] Z. Vasicek. Cartesian gp in optimization of combinational circuits with hundreds of inputs and thousands of gates. In *Genetic Programming, 18th European Conference, EuroGP 2015*, LCNS 9025, pages 139–150. Springer International Publishing, 2015.

[12] Z. Vasicek and L. Sekanina. Formal verification of candidate solutions for post-synthesis evolutionary optimization in evolvable hardware. *Genetic Programming and Evolvable Machines*, 12(3):305–327, 2011.

[13] Z. Vasicek and L. Sekanina. Evolutionary design of approximate multipliers under different error metrics. In *17th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems*, pages 135–140. IEEE Computer Society, 2014.

[14] Z. Vasicek and L. Sekanina. How to evolve complex combinational circuits from scratch? In *2014 IEEE International Conference on Evolvable Systems Proceedings*, pages 133–140. Institute of Electrical and Electronics Engineers, 2014.

[15] Z. Vasicek and L. Sekanina. Circuit approximation using single- and multi-objective cartesian gp. In *Genetic Programming*, LNCS 9025, pages 217–229. Springer International Publishing, 2015.

[16] Z. Vasicek and L. Sekanina. Evolutionary approach to approximate digital circuits design. *IEEE Transactions on Evolutionary Computation*, 99(99):1–13, 2015.

[17] S. Venkataramani, K. Roy, and A. Raghunathan. Substitute-and-simplify: A unified design paradigm for approximate and quality configurable circuits. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, pages 1367–1372, March 2013.

[18] S. Venkataramani, A. Sabne, V. Kozhikkottu, K. Roy, and A. Raghunathan. Salsa: Systematic logic synthesis of approximate circuits. In *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, pages 796–801, June 2012.

[19] N. Weste and D. Harris. *CMOS VLSI Design: A Circuits and Systems Perspective*. Addison-Wesley Publishing Company, USA, 4th edition, 2010.

[20] A. Wiltgen, K. Escobar, A. Reis, and R. Ribas. Power consumption analysis in static cmos gates. In *Integrated Circuits and Systems Design (SBCCI), 2013 26th Symposium on*, pages 1–6, Sept 2013.