Book Reviews

Evolutionary Algorithms for VLSI CAD—R. Drechsler. (Boston, MA: Kluwer, 1998, ISBN 0-7923-8168-8, 183 pp., $97.50) *Reviewed by Jason D. Lohn.*

One of the major challenges in the field of integrated circuit design is coping with difficult combinatorial optimization problems. Simply finding the minimum length of wire needed to connect a block of transistors is NP-hard. When factoring in a handful of other simultaneous optimization dimensions such as connections to other blocks of components and area minimization, it is easy to see the difficulty facing circuit designers. Since many of the problems encountered do not have polynomial-time solutions, very large scale integration (VLSI) algorithm designers experiment with various optimization techniques such as integer linear programming and simulated annealing. Optimization methods for VLSI computer-aided design (CAD) incorporating evolutionary search began appearing in research articles in the late 1980's. As the first body of work devoted exclusively to evolutionary algorithms (EA's) in VLSI CAD, this monograph attempts to fill a noticeable void in the literature.

The book is divided into two parts. Part I "Basic Principles" provides an overview of the book and discusses the underlying principles of EA's and algorithm performance issues. The treatment of EA's here is cursory at best and concentrates mainly on the classic genetic algorithm. An overview of some aspects of VLSI CAD is also included; however, only the latter phase of the circuit design process is covered, and analog integrated circuit design is not touched upon.

The second part entitled "Practice" comprises roughly three-quarters of the book and deals with tools and applications. A software tool called the Genetic Algorithm Managing Environment (GAME) is described in one chapter. It provides a flexible environment in which an algorithm designer can easily interface EA's to VLSI CAD tools to facilitate experimentation and production runs. One of the author's themes is that problem-specific knowledge is necessary for EA's to be competitive with other optimization approaches. Support for this argument is provided by the numerous case studies examined in a lengthy chapter concerning applications of EA's to logic synthesis, mapping, and testing. In logic synthesis, one wishes to implement a Boolean function in hardware while satisfying certain constraints (e.g., power, delay area). A category of logic realization called Fixed Polarity Reed Muller expressions is described. Such expressions are difficult to minimize and an approach is described that uses a hybrid EA (one that incorporates problem-specific heuristics). For large Boolean functions, it is shown that the EA presented can achieve smaller expressions as compared to standard tools, although the EA required more computer time.

The section on mapping includes EA applications in partitioning, floorplanning, and placement and routing problems. Partitioning consists of mapping blocks of circuit components to two-dimensional regions on the surface of a chip. Floorplanning then determines the shape (aspect ratio) and position of the blocks within each region. If the aspect ratios are fixed, this activity is called placement. The problem of routing interconnections is tackled next: unused regions between blocks are divided into rectangular areas and the optimization algorithm must find pathways to interconnect the blocks. In each of these tasks, numerous physical design rules must be obeyed for the circuit to be fabricated properly. VLSI circuit testing using EA's is covered last and deals with automatic test pattern generation to detect defects that may enter during chip fabrication. Each of these subtopics represents very difficult optimization problems, and results using EA's are presented and compared to other algorithms.

Overall, this book will be useful to VLSI algorithm designers who are familiar with EA's and wish to learn more about applications and recent experimental results. With its large bibliography it could be used as a good entry point into the literature. It might also serve those EA practitioners who are interested in learning about the hard optimization problems encountered in the VLSI domain and the hybrid EA techniques employed to solve them. On the downside, one gets the sense that it is mainly collection of the author's previous publications and therefore does not cover EA's in VLSI CAD in a comprehensive manner. Also, the text is riddled with awkward English prose, redundant sentences, and a large number of grammatical and spelling mistakes. A more thorough treatment of EA's would likely benefit VLSI circuit designers since they presumably know little about EA's and stand to benefit the most from applying these techniques. Likewise, more background on VLSI design would be helpful for application-oriented EA researchers looking into this domain. In the end VLSI designers care most about the quality of the solutions their algorithms produce and the resources required to produce them. This book will certainly raise their awareness of EA techniques and may help them to find higher quality solutions more effectively.

Genetic Programming III: Darwinian Invention and Problem Solving—J. R. Koza, F. H Bennett III, D. Andre, and M. A. Keane, Eds. (San Francisco, CA: Morgan Kaufmann, 1999) *Reviewed by Adrian Stoica.*

The fundamental idea of applying evolutionary principles to automatically create computer programs can be traced back to Turing [1]–[2], who identified the "genetical or evolutionary search" as a means of creating an intelligent machine. Genetic programming (GP) is a domain-independent method relying on genetic breeding of populations of computer programs to automatically create a computer program from high-level requirements. This book is the third in a special series dedicated to GP. The first book [3] introduced the basic

Manuscript received April 18, 1999.
The reviewer is with Caelum Research Corp., Moffett Field, CA 94035-1000 USA (e-mail: Jlohn@ptolemy.arc.nasa.gov).
Publisher Item Identifier S 1089-778X(99)07200-8.

Manuscript received June 12, 1999.
The reviewer is with the Jet Propulsion Laboratory, Pasadena, CA 91109 USA.
Publisher Item Identifier S 1089-778X(99)07079-4.

principles of GP, and the second book [4] focused on subroutines (automatically defined functions) and scalability of GP. This third book brings further evidence that GP possesses a characteristic set of attributes of system for automatically creating computer programs.

The book is organized in ten parts: Introduction, Background on Genetic Programming and Evolutionary Computation, Architecture Altering Operations, Genetic Programming Problem Solver (GPPS), Automated Synthesis of Analog Electrical Circuits, Evolvable Hardware, Automatic Discovery of Cellular Automata Rules, Discovery of Motifs and Programmatic Motifs for Molecular Biology, Parallelization and Implementation Issues and finally the Conclusion. The book is accompanied by a 50-min videotape [5].

The book starts by presenting capabilities list of 16 attributes reasonably expected to be possessed by a system for automatically creating computer programs. These attributes are: 1) it starts with "what needs to be done;" 2) tells us "how to do it," 3) produces a computer program, 4) automatic determination of program size, 5) code reuse, 6) parameter reuse, 7) internal storage, 8) iterations, loops, and recursions, 9) self-organization of hierarchies, 10) automatic determination of program architecture, 11) wide range of programming constructs, 12) well-defined, 13) program independent, 14) wide applicability, 15) scalability, 16) competitive with human-produced results. The main point of the book is that GP unconditionally possesses 13 out of 16 attributes of a system for automatically creating computer programs and that GP at least partly satisfies the remaining three attributes. Some of the above-mentioned attributes have received justification in prior work on GP and are not detailed explicitly, except in the concluding chapter where the evidence for each individual attribute is reviewed.

The creative aspect of GP is given special consideration, in particular its capability to produce solutions competitive with human-produced solutions to important, real-world problems. A solution is considered competitive if it satisfies a set of criteria, for example, whether it is patentable or publishable in a peer-reviewed journal independently of the fact that it was automatically created. Fourteen instances in which GP determines competitive solutions are presented in the book, ten of which relate to previous patents. Nine of these results that rediscover patented solutions are in the area of analog circuit design, which is a particularly challenging field for human designers.

Automated design of electrical circuits is in fact the very core of the book, with half of the over 1100 pages being dedicated to it. The extensive treatment of the subject makes the book likely to be the most comprehensive writing on evolutionary circuit design of electrical circuits. Not only the final result of experiments are presented, but also the intermediary steps, discussion of efforts, and the reasoning behind the experiments leading progressively to the final result are given. The natural drawback of having a continuity in argumentation across several chapters is that it requires reading the chapters in the order they appear, otherwise the reader would miss part of the reasoning that determined choices for the starting conditions, fitness evaluation, etc. Although from the GP perspective the parts dedicated to automated circuit synthesis and evolvable hardware are intended to demonstrate the depth of GP with architecture-altering operations, I think that the material provided could have very well been organized as a monograph in the field of evolution of electronic circuits. The only missing thing from this different perspective is gathering the lessons learned and conclusions over the whole set of experiments.

One characteristic of the technique used in the work described in this book is the use of an embryonic circuit, which is placed in a test fixture and further grown. This is different than most other approaches to evolvable hardware and fits well with the context of genetic programming. Genetic programming, as opposed to evolutionary programming and evolution strategies, considers an important role for the crossover operator, and a special chapter of the book is dedicated to analyzing this role. The results of GP experiments with, and without, crossover indicate that crossover indeed accelerates convergence to the desired solution. In a common framework for both crossover and mutation, mutation is seen as a crossover with the statistical equivalent of a randomly chosen parent from the first generation, and in this context it appears more useful to have the crossover with a fitness-selected parent from the current generation.

Very interesting issues are related to the possibility of using GP as an invention machine, to discover novel solutions to problems, or to find ways around patented solutions (e.g., by rewarding those topologies that differ from patented circuit topologies). The examples in the book demonstrate this possibility of automatically discovering patentable electronic circuits. Subcircuits—part of the resulting circuits—may also be patentable, but their identification may be more involved. An example of such a subcircuit selected as useful by evolution in several experiments is the combination of two transistors known to the electronic engineers as the Darlington pair. One should note, however, that this is only a part of the circuit and we have no direct information about its functionality—only the fitness of the overall circuit is evaluated during evolution. For example, assume that we do not know about the Darlington pair, and the combination appears while trying to obtain a patentable solution for a squaring circuit. In this case we would not know necessarily that it is useful to isolate the two transistors and associate them with a specific functionality (amplification) for which we would patent. Of course, such emerging subcircuits could be analyzed in isolation to see what their function is, and maybe even prove to be patentable solutions, but not to the problem we were trying to obtain a patent in the first place.

The evolvable hardware part is discussed solely in the context of field-programmable gate arrays (FPGA's), which are looked at as devices that can accelerate the computationally burdensome fitness evaluation task of evolutionary algorithms. I believe this is probably because the field started with FPGA's, they are the most widely used reconfigurable devices today, and the experiment in the book is performed on an FPGA. Evolvable hardware, however, is not limited to FPGA's, and evolution on other devices (such as field programmable analog arrays [6], [7] and programmable analog application specific integrated circuits (ASIC's) [8], [9]) has been demonstrated. Moreover, considering the arguments made earlier in the book for evolving analog circuits, evolution on flexible reconfigurable analog devices would appear to be very useful.

Some aspects that made the reconfigurable devices in the Xilinx XC6200 series suitable for evolvable hardware are highlighted. One should note that the reconfigurable devices market is moving at a very fast pace. In the short time since the writing of the chapter several changes occurred, including the fact that the Xilinx XC6200 series was discontinued. On the bright side, however, what appeared as unique characteristics of the XC6216 chip are now either matched by other chips, or the issues preventing evolution on other devices can be avoided. For example, the vulnerability of most FPGA's to damages caused by certain configurations can be avoided by software techniques producing only legal circuit configurations, as recently shown in [10]. The experiment on the Xilinx chip shows the use of the hardware in accelerating evolution. One should remark that the fitness evaluation is performed completely on the chip (in other approaches one measures the hardware response, whereas the actual fitness is calculated in software).

The number of evaluations used for most experiments in the book is large compared to other reported work, most runs being done with a population of 640 000 individuals for tens or hundreds of generations and required tens of hours for a run in the parallel implementation

(the time is primarily consumed by the SPICE simulator). The authors point out that today's computing power for equivalent platforms is already almost an order of magnitude higher than what it was at the time when the experiments were first run, and are expected to reduce to about 3 min by the year 2010. This would definitely make experiments of the complexity described in the book more accessible, however, designing more complex circuits may still remain hard, as SPICE does not scale well (time for SPICE simulators increases nonlinearly as a function of the number of nodes in the netlist, in an approximately subquadratic to quadratic way).

The book ends with the "Conclusion" chapter that states the main arguments justifying how GP possesses all the 16 attributes of a system for automatically creating computer programs. Most of these attributes are completely possessed by the GP, while for "wide applicability" there is considerable evidence, for "scalability" there is some evidence and for competitiveness with human-produced results there is moderate evidence (in the form of the 14 instances claimed by the authors).

This book, which is very clear and detailed overall, is accessible to a broad audience of computer scientists and engineers. The appealing subject of the book, the extremely interesting experiments described within, and the elegant reasoning make it a very attractive reading, while the importance of the points demonstrated in the book make it a fundamental reference material for evolutionary computation, as well as for automated design.

REFERENCES

[1] A. M. Turing, "Intelligent machines," in *Mechanical Intelligence: Collected Works of A. M. Turing*, D. C. Ince, Ed. Amsterdam, The Netherlands: North Holland, 1992, pp. 107–128.

[2] ____, "Computing machinery and intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[3] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.

[4] ____, *Genetic Programming II: Automated Discovery of Reusable Programs*. Cambridge, MA: MIT Press, 1994.

[5] J. R. Koza, F. H Bennett III, D. Andre, M. A. Keane, and S. Brave, *Genetic Programming III Videotape: Human-Competitive Machine Intelligence*. San Francisco, CA: Morgan Kaufmann, 1999.

[6] S. J. Flockton and K. Sheehan, "Intrinsic circuit evolution using programmable analog arrays," in *Evolvable Systems: From Biology to Hardware*, M. Sipper, D. Mange, and A. Perez-Uribe, Eds. (Lecture Notes on Computer Science, vol. 1478). New York: Springer, 1998, pp. 144–153.

[7] R. S. Zebulum, M. A. Pacheco, and M. Vellasco, "Analog circuits evolution in extrinsic and intrinsic modes," in *Evolvable Systems: From Biology to Hardware*, M. Sipper, D. Mange, and A. Perez-Uribe, Eds. (Lecture Notes on Computer Science, vol. 1478). New York: Springer, 1998, pp. 154–166.

[8] M. Murakawa, S. Yoshizawa, T. Adachi, S. Suzuki, K. Takasuda, M. Iwata, and T. Higuchi, "Analogue EHW chip for intermediate frequency filters," in *Evolvable Systems: From Biology to Hardware*, M. Sipper, D. Mange, and A. Perez-Uribe, Eds. (Lecture Notes on Computer Science, vol. 1478). New York: Springer, 1998, pp. 134–143.

[9] A. Stoica, A. Fukunaga, K. Hayworth, and C. Salazar-Lazaro, "Evolvable hardware for space applications," in *Evolvable Systems: From Biology to Hardware*, M. Sipper, D. Mange, and A. Perez-Uribe, Eds. (Lecture Notes on Computer Science, vol. 1478). New York: Springer, 1998, pp. 166–173.

[10] A. Stoica "Toward evolvable hardware chips: Experiments with a programmable transistor array," in *Proc. 7th Int. Conf. Microelectronics for Neural Networks, Evolutionary and Fuzzy Systems*, Granada, Spain, Apr. 7–9, 1999, pp. 156–162.

[11] D. Levi and S. A. Guccione, "Genetic FPGA: Evolving stable circuits on mainsteam FPGA devices," in *Proc. 1st NASA/DOD Workshop on Evolvable Hardware*, A. Stoica, D. Keymeulen, and J. Lohn, Eds. Los Alamitos, CA: IEEE Comput. Soc. Press, 1999, pp. 12–17.