České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra číslicového návrhu

Czech Technical University in Prague
Faculty of Information Technology
Department of Digital Design

Ing. Jan Schmidt Ph.D.

Experimentální algoritmika a postupy experimentálního vyhodnocení nástrojů automatizace návrhu číslicových zařízení

Experimental Algorithmics and Methods for Experimental Evaluation of EDA Tools

Summary

Algorithms used in Electronic Design Automation are complex, layered heuristics. Their performance projects directly into the performance of the chips designed with their support. In the past several years, several cases of provably poor performance of logic synthesis tools were reported. The investigation revealed that circuit descriptions deviating much from the standard human inputs cannot be processed efficiently, and that current evaluation methods do not help in understanding their operation.

Algorithms of similar complexity must be evaluated experimentally. Experimental techniques have been refined in the context of natural science for a long time. Experimental algorithmics applies the methods to study algorithms. Whereas measured values in physical systems usually exhibit errors with known (normal) distribution, problem instances have unknown sources of variance. This is overcome by generating instances with uniform probability which permits variance elimination.

The set of practical circuits cannot be characterized, despite the progress in the last 60 years. Hence, instances cannot be generated and practical examples must be collected instead. Strange sources of variance appear, and methods that interpret experimental data are restricted.

In such a limited environment, the measured examples must be well documented to prove that they are representative and thus relevant. This is difficult to achieve with older benchmark sets, but ways to derive such benchmarks from open-source designs do exist.

Souhrn

Algoritmy automatizace číslicového návrhu jsou složité, mnohavrstevné heuristiky. Jejich účinnost se přímo promítá do kvality čipů, které byly s jejich pomocí navrženy. V několika posledních letech se objevily případy, kdy nástroje logické syntézy poskytly prokazatelně nekvalitní výsledky. Bližší zkoumání ukázalo, že nástroje nedokáží efektivně zpracovat vstup, který se příliš odlišuje od běžného vstupu návrháře, a že současné metody experimentálního vyhodnocení příliš nepomáhají porozumět činnosti těchto nástrojů.

Algoritmy zmiňované složitosti je nutno vyhodnocovat experimentálně. Experimentální techniky mají svůj původ v přírodních vědách a vyvíjely se po dlouhou dobu. Experimentální algoritmika aplikuje tyto metody na hodnocení algoritmů. Zatímco měření fyzických systémů je zatíženo chybami, které vykazují normální rozložení, instance problémů na vstupu algoritmů mohou vykazovat neznámé zdroje variance. To se překonává náhodným generováním instancí tak, že každá možná instance má stejnou pravděpodobnost, což dovoluje užít statistických metod k eliminaci každé variance.

Množinu praktických obvodů není možno vymezit, bez ohledu na pokroky posledních 60 let. Instance proto nemohou být generovány a je třeba sbírat příklady z praxe. Objevují se nečekané zdroje variance, což značně omezuje metody použitelné pro interpretaci dat.

V tak omezeném kontextu, použité příklady musí být dobře dokumentovány, aby bylo zřejmé, že jsou reprezentativní a tudíž relevantní. Toho je těžké dosáhnout se staršími sadami příkladů. Nicméně, existují cesty, jak odvodit důvěryhodné příklady z otevřených návrhů.

Klíčová slova

Logická syntéza, standardní zkušební úloha, algoritmus, experimentální vyhodnocení, interpretace dat, kvalitativní analýza, zdroj variance, eliminace variance.

Keywords

Logic synthesis, benchmark circuit, algorithm, experimental evaluation, data interpretation, qualitative analysis, source of variance, variance elimination.

Contents

1 Introduction

Recent integrated circuits have high complexity and cannot be designed manually. Instead, the design relies on Electronic Design Automation (EDA) tools. The tools do automate the process to a large extent; but, because of the complexity, they are, in principle, heuristic optimizers.

The design of a device starts with specification *what* the device shall do, that is, by specifying its *behavior* (Gajski and Kuhn 1983). An abstract *structure* of sub-blocks is designed then, describing *how* the function will be implemented. This step is called *synthesis*. As the integrated devices are implemented as two- or three-dimensional structures, the *physical design* phase must determine *where* the processing and communication will take place.

The part of synthesis responsible for transforming behavior described by a Boolean function into a structure of logic gates is called *combinational* logic synthesis. This will be the topic of the lecture.

Logic synthesis came into research focus in the eighties. Methods of that era started with descriptions in a standardized or even canonical form (normal forms, truth tables) and gradually developed the necessary gate-level structure by its *decomposition* (Brayton et al. 1984; Hachtel and Somenzi 1996; Hassoun and Sasao 2002). Those methods scaled poorly with growing design complexity. Around 2005, an even older idea of design by gradual transformations was revived. Thanks to new approaches to data structures and the transformations themselves, this effort started a new epoch in logic synthesis and brought it to wide industrial acceptance. Since then, logic synthesis is considered a mature process.

Cong and Minkovich (2007) reported provably poor logic synthesis performance on artificial circuits. Kubalík, Fišer, and Kubátová (2006) and Fišer, Kubalík, and Kubátová (2008) found similarly poor performance in practical designs, but with a non-standard design procedure. These findings have both negative and positive interpretations. Recent logic synthesis may be flawed; on the other hand, finding what causes the observed failures may lead to new approaches to logic synthesis.

In the course of the research (Schmidt 2015), we were able to prove that neither the fears nor the hopes are well founded. We were able to offer and experimentally support the hypothesis that the existing tools are adapted to *human input*, and when the input deviates too much, the tools fail to compensate.

This is a *qualitative* result, unlike the more common *quantitative* evaluation found in most research contributions. Such a transition

is analyzed in core materials about scientific method (Fisher 1993), has been called for in the framework of Experimental Algorithmics (McGeoch 1996; Moret 2002; Hoos and Stützle 2007), but is rarely discussed in EDA research.

This lecture aims to apply existing standards of experimental work to EDA research. We outline Experimental Algorithmics and its (often unspoken) assumptions first. Then, we summarize answers what *human input* may actually mean, and that its characterization is so far unknown, if it is indeed possible. It then becomes apparent, that the assumptions of Experimental Algorithmics cannot hold. Although it greatly reduces the repertoire of usable methods, we argue that we can still achieve significant results by critical evaluation of experimental material.

2 Algorithm evaluation

Algorithms are evaluated for many purposes, theoretical and practical. The purpose of the algorithm is projected into a *question* about the outcome of the algorithm and its dependency on algorithm input (problem instance). To get answers that hold not only for a single run but in general, a kind of *instance characterization* and *algorithm characterization* must be devised, and the sought dependency be expressed in terms of the two.

We require any evaluation to be *relevant* in the sense that it brings information, and *reliable*, so that its statements hold under assumptions, explicit or implicit. To be accepted as such, *trust* is needed. The question may be purely quantitative (performance prediction), qualitative (explanation), or combined in nature (comparison). For some situations, e.g., making qualitative conclusions from quantitative data, some acceptance criteria do exist; elsewhere, the acceptance of research community decides.

As a famous example, Selman, Mitchell, and Levesque (1992) demonstrated that size is not the most relevant characterization of Satisfiability Problem (SAT) instances to estimate their hardness.

The methods used to answer a question naturally depend on the question itself, but are themselves limited in their abilities. The underlying mathematics, statistical methods, or technical issues in experiments are examples of such limitations. Examples of the most prominent cases are summarized in Table 1.

2.1 Analytical methods

The analytical approach has the advantage that it applies to all possible (not only the known) instances. It thus does not have problems with instance selection. It is indispensable in theoretical studies.

From the point of view of engineering application, however, it can only tell limited statements about limited class of algorithms. First of all, the algorithms analyzed so far had to be relatively simple. Second, they do not provide all the information one would eventually need. Classical analyses provide worst case information, often not of primary importance in applications. Expected (average) case characterization is obtained using strong assumptions (uniform instance distribution), and sometimes only asymptotically, as in the Randomized Quicksort analysis (Cormen et al. 2009, page 179). Further details of the characterization and its distribution are beyond the capabilities of analytical methods.

2.2 Experimental methods in science

Experimental methods originate in natural science (Fisher 1993). In that setting, the dependency of an measured variable on one independent variable is observed. All other variables that can influence the studied object are held constant. Often, the observed (quantitative) data are expressed as a (still quantitative) law. If the credibility criteria are met, the law can give us *understanding* of the object, therefore qualitative information.

Again, a well-known example of such a transition is the origin of quantum mechanics from the law of black body radiation (Kragh 2000), in this case even against the experimenter's judgment and opinion.

When experimenting with physical objects, all output measurements exhibit errors, and so do the set variables (Figure 1). Experience in that field tells that these errors have normal distribution. Using this (often unspoken) assumption, statistical methods can be used to remove the variance from measured data. Notice that even for the ubiquitous averaging to work, the distribution must be symmetric.

Table 1: Algorithm evaluation methods

Approach	Method	Instances	Instance char.	Algorithm char.	Statement
Classical complexity theory	analytical	all	size	time, memory	worst case
Approximative algorithms study	analytical	all	size	quality	worst case
Randomized algorithms study	analytical	all	size	optimality, time	expected case, success prob.
Experimental algorithmics	experimental	generated	any	optimality, time	observed cases
Engineering evaluation	experimental	collected	any	any metric	observed cases

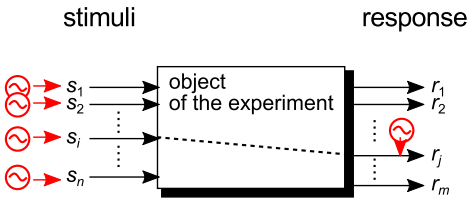

Figure 1: An experiment with variance sources

The importance of variance removal for qualitative conclusions can be documented by Tycho Brahe's astronomic observations (cca. 1597), with precision improved from 10' to 2'. Before Tycho, circular motion of planets was consistent with observation, while after the improvement, it was not (Rawlins 1993).

2.3 Experimental algorithmics

Algorithms that are too complex for analytical evaluation, can still be studied experimentally. Experimental algorithmics (Moret 2002; Hoos and Stützle 2007) aims to bring the standards of relevance and reliability from experimental science into such study of algorithms.

A difference from experiments in natural science, which is crucial from our point of view, lies in the sources of variance present. First, let us assume a deterministic algorithm. For a single value of instance characterization (say, size) we have multiple instances. They may have, as the case of SAT phase transition illustrates, properties which have been abstracted away by the instance characterization chosen but which still can influence the algorithm behavior. The standard remedy of experimental algorithmics is to assume that *all* instances are equally probable. Thus, any hidden source of variance cannot affect the median or average value. They, however, should be indicated by greater variance of the observed data, suggesting a loss of relevance.

The requirement of uniform distribution opens up the way to *generate* instances for experiments. Once the generator is proven that it produces any instance (or any instance in a chosen, characterized class) with equal probability, unlimited amount of experimental material is at the experimenter's disposal.

As it is standard in science, experiments must be designed to eliminate all sources of variance. Randomized algorithms have an additional source of variance, the internal (pseudo-)random number generator. Again, its statistical characteristics are known, and the variance can be eliminated from measured data by simple means. Usually, the first

stage is to average runs over a single instance, then over multiple instances.

With sources of variance in mind, algorithm comparison stops to be the matter of comparing two numbers. Experimental algorithmics brought the notion of *dominance*, where one algorithm is better on *each* instance. When neither the compared algorithm dominates the other, the situation can still be quantitatively characterized by giving the probability of one being better.

3 Practical circuit instances

The question, what a practical circuit is, dates back to Shannon (1949). By his earlier theorem, for "most" circuits of n inputs computing any function, the size of the circuit is asymptotically larger than $2^n/n$. Therefore, Shannon asks, why most real circuits belong to the small group outside the "most". The answer by Shannon (1949, page 89) is that the human designer thinks of functions that can be explained and that follow some design patterns.

Shannon then finds *group invariants*, such as symmetry (McCluskey 1956) as common properties. Further important feature is *decomposability*, that is, the property of having a concise description using subfunctions.

Ross, Noviskey, and Taylor (1991) take this conjecture farther. Using a general decomposition scheme, the authors decomposed Boolean functions not only from electronic design, but also from other fields such as pattern recognition. They found that human design indeed follows patterns, and that these patterns can be recognized and utilized.

In other fields, graphs coming from practical instances were characterized (cf. Albert, Jeong, and Barabasi 1999, for the WWW; and Ansótegui, Bonet, and Levy 2009, for SAT instances). Unfortunately, Boolean functions do not have a natural relationship to graphs (although their implementations have). As a result, we have *benchmark sets* as sources of evaluation instances (Brglez and Fujiwara 1985; Brglez, Bryan, and Kozminski 1989; Yang 1991a; Yang 1991b; McElvain 1993; Albrecht 2005).

From the experimental point of view, they are problematic. Industrial benchmarks are scarce, as the industrialists must observe strict non-disclosure rules. Therefore, the sets are hard to balance, and are not statistically representative for any class of practical circuits. Moreover, some of them have been altered in a nearly undocumented man-

ner, and their authenticity is dubious.

All those influences cause situation unlike that in other areas. We cannot characterize input instances, and do not have reliable, statistically representative material for experimental evaluation.

4 Evaluation methods for EDA

From the preceding sections, it follows that in practical evaluation, all sources of variance must be identified and eliminated. Although this may seem a trivial conclusion, its consequences are not trivial to perform.

Internal variance in randomized algorithms is the easiest one to eliminate. The series of random numbers produced by the generator have uniform distribution. Even if the algorithm modifies them internally to produce, e.g., normal distribution, such transformation is a part of the algorithm and does not affect evaluation.

4.1 Hidden sources of variance

As shown above, we cannot count on uniform instance probability. Moreover, EDA algorithms for combinational logic synthesis do not work on Boolean functions as abstract structures directly, but on their *descriptions* in Hardware Description Languages (HDLs), resembling (parallel) programs to a degree. Many descriptions may represent the same Boolean function. Some of them are *equivalent* not only because they describe the same Boolean function, but because they describe it the same way in the eyes of the designer. Few programmers care about the order of variable declaration, and many consider different loop constructs that go over the same values entirely equivalent.

Yet, Puggelli et al. (2011) discovered that for many tools, semantically equivalent control structures and a number of other constructs are not equivalent. Multiple authors (Puggelli et al. 2011; Fišer and Schmidt 2011; Fišer, Schmidt, and Balcárek 2014) reported that synthesis tools of both industrial and academic are sensitive to declaration ordering and even identifiers. The sensitivity can be attributed to algorithm implementation (as opposed to the abstract algorithm itself). In some cases, they were traced to container iteration and hash functions (Shum and Anderson 2012; Fišer and Schmidt 2011; Fišer, Schmidt, and Balcárek 2014).

Such sensitivities are sources of variance, because the exact form of the input description cannot be predicted and there is no prescription to

follow. They can be eliminated, however. When one input description is replaced by set of descriptions where each ordering and each alternative control structure is uniformly probable, statistical methods apply.

Already Puggelli et al. (2011) complained that hidden variance is often greater than improvements reported in research papers, and that those reports are in principle flawed. The variance found in Fišer and Schmidt (2011) and Fišer, Schmidt, and Balcárek (2014) are substantially greater, which further exaggerates the problem. So far, there are no systematic methods to discover such sources of variance; their characterization (e.g., declaration order) must be invented, and tested. Thus, even rudimentary checks on commonly known sources of variance would improve the reliability of any EDA algorithm comparison.

4.2 Instance transformation

The technique of transforming a single instance is useful, and in more situations than for variance elimination; however, there are limits to its usefulness.

Cong and Minkovich (2007), Schmidt (2015), and Fišer and Schmidt (2012a) use selected transformations to discard the original structure of the description and to test its rediscovery. Outside of the evaluation field, Fišer and Schmidt (2012c) and Fišer and Schmidt (2012b) (mis-)use the above described variance sources to randomize deterministic algorithms, and to get the advantages of randomized algorithms.

While transforming an input instance may reveal useful properties of the algorithm, the transformed instance is still an artificial object and not an authentic description obtained from a designer. This is a direct consequence of our inability to characterize what a practical instance is and to identify *all* hidden sources of variance.

There are numerous examples of experiments that suffer from this pitfall. Perhaps the most important ones are the conjectures in Cong and Minkovich (2007). The instances used there result from transformations that are never used in practice, and their structure is so far from the original that no designer is supposed to produce such input. Therefore, the results lack relevance. Moreover, the obtained descriptions cannot emulate designs by less competent designers, as the article suggests.

One idea of Harlow III and Brglez (2001) is to replace multiple circuits by transformed descriptions of their small subset. The sources of variance are not interchangeable; the transformation variance should have been eliminated independently, as its distribution is known. In

that case, however, the number of circuits would be insufficient.

Fujita et al. (1993) praise the "redundancy" of the *alu4* benchmark as a good opportunity to test their procedures. The version of the circuit cited in the text is one or two orders of magnitude bigger than the original solution (it is in fact the famous 74181 arithmetic circuit, for which the original gate-level solution is known, but the identity was never publicly admitted). The enlargement results from processing by poorly performing tools, which the authors probably knew about. The artificially enlarged circuit may still have been able to support their case; in general, its applicability is a dangerous assumption.

The same transformation as in Cong and Minkovich (2007) is used in Kubalík, Fišer, and Kubátová (2006) and Fišer, Kubalík, and Kubátová (2008), not for evaluation, but as a practical measure. In this situation, one can hardly advice "don't do it". If there is no alternative, a synthesis procedure which is not sensitive to such a transformation (the classical decomposition-based synthesis in this case) should be used. The belief that the transformed descriptions can be used as benchmarks, expressed in Fišer and Schmidt (2012a), is, of course, incorrect.

4.3 Applicable methods

The absence of uniform instance probability, together with the presence of hidden sources of variance, greatly limits applicable methods. Under the most stringent integrity requirements, all that remains is the existential quantifier: "there is a circuit description such that...". At this point, the question is how significant such a statement is.

There are no formal methods to quantify; it is the matter of *trust* that the circuit is representative for current design practice and not a damaged legacy circuit, as is the case of *alu4*. To develop trust, one has to document the reasons why the presented circuit is relevant, or why the given circuit set is balanced. This is not possible with older benchmark sets. A step towards trustworthy sets is Albrecht (2005), where at least a subset is based on open source circuits. Provided a documented and controllable HDL elaborator, such as ZamiaCAD (Bartsch et al. 2012), documented, comprehensible and therefore trusted combinational benchmarks can be produced.

5 Conclusions

To achieve scientific standards and to gain trust, experimental evaluation should adapt standard procedures from natural science. A notable

difference between a natural science experiment and algorithm evaluation are the sources of variance. While in experiments with physical objects, variance is caused by measurement errors and has normal distribution, instances of input data can exhibit variance of unknown probability distribution. Experimental algorithmics overcomes this difficulty using instance sets where each possible instance has the same probability to actually appear, and the variance is eliminated. The set of practical circuits cannot be characterized, and so this approach is not possible in the evaluation of EDA algorithms. Although some sources of variance (e.g., internal sources of randomized algorithms) do have known distributions and can be eliminated, the rest reduces applicable methods. The variance introduced by instance transformation is, in general, not interchangeable with variance caused by design style and thus, transformed instances cannot be used as benchmarks, contrary to multiple authors' claims. It also constraints applicable methods and hinder trustworthy generalization of the results. To obtain trusted benchmarks, their origin and derivation must be publicly known.

References

Albert, R., H. Jeong, and A.-L. Barabasi (1999). "Internet: Diameter of the World-Wide Web." In: *Nature* 401.6749, pp. 130–131.

Albrecht, C. (2005). *IWLS 2005 Benchmarks*. URL: `http://iwls.org/iwls2005/benchmarks.html`.

Ansótegui, C., M. Bonet, and J. Levy (2009). "On the Structure of Industrial SAT Instances." English. In: *Principles and Practice of Constraint Programming - CP 2009*. Ed. by I. P. Gent. Vol. 5732. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 127–141. ISBN: 978-3-642-04243-0. DOI: `10.1007/978-3-642-04244-7_13`. URL: `http://dx.doi.org/10.1007/978-3-642-04244-7_13`.

Bartsch, G. et al. (2012). *ZamiaCAD: Open Source Platform for Advanced Hardware Design*. URL: `http://zamiacad.sourceforge.net`.

Brayton, R. K. et al. (1984). *Logic Minimization Algorithms for VLSI Synthesis*. Boston, MA: Kluwer Academic Publishers, p. 192.

Brglez, F., D. Bryan, and K. Kozminski (May 1989). "Combinational profiles of sequential benchmark circuits." In: *Circuits and Systems, 1989., IEEE International Symposium on*, 1929–1934 vol.3. DOI: `10.1109/ISCAS.1989.100747`.

Brglez, F. and H. Fujiwara (1985). "A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortan." In: *Proceedings of the International Symposium on Circuits and Systems*, pp. 663–698.

Cong, J. and K. Minkovich (Feb. 2007). "Optimality Study of Logic Synthesis for LUT-Based FPGAs." In: *IEEE Trans. on Computer-Aided Design* 26.2, pp. 230–239.

Cormen, T. H. et al. (2009). *Introduction to Algorithms*. MIT Press.

Fišer, P. and J. Schmidt (2012a). "A Difficult Example Or a Badly Represented One?" In: *Proc. of 10th International Workshop on Boolean Problems*. Freiberg, DE: Technische Universität Bergakademie, pp. 115–122. ISBN: 978-3-86012-438-3.

Fišer, P., P. Kubalík, and H. Kubátová (2008). "An Efficient Multiple-Parity Generator Design for On-Line Testing on FPGA." In: *Proc. 11th Euromicro Conference on Digital Systems Design (DSD'08)*. Parma, Italy, pp. 94–99.

Fišer, P. and J. Schmidt (2011). "How Much Randomness Makes a Tool Randomized?" In: *Proc. of the 20th International Workshop on Logic and Synthesis (IWLS)*. San Diego, California, USA, pp. 136–143.

— (Apr. 2012b). "Improving the iterative power of resynthesis." In: *Design and Diagnostics of Electronic Circuits Systems (DDECS), 2012 IEEE 15th International Symposium on*, pp. 30–33. DOI: 10.1109/DDECS.2012.6219019.

— (2012c). "On Using Permutation of Variables to Improve the Iterative Power of Resynthesis." In: *Proc. of 10th Int. Workshop on Boolean Problems (IWSBP)*. Freiberg (Germany), pp. 107–114.

Fišer, P., J. Schmidt, and J. Balcárek (2014). "Sources of Bias in EDA Tools and Its Influence." In: *Proc. of 17th IEEE Symposium on Design and Diagnostics of Electronic Systems (DDECS)*. Warsaw, Poland, pp. 258–261.

Fisher, R. A. (1993). *Statistical methods, experimental design, and scientific inference*. Oxford University.

Fujita, M. et al. (1993). "Multi-level Minimization by Partitioning." In: *Logic Synthesis and Optimization*. Ed. by T. Sasao. Springer Science & Business Media, pp. 109–126.

Gajski, D. D. and R. H. Kuhn (Dec. 1983). "Guest Editors' Introduction: New VLSI Tools." In: *Computer* 16.12, pp. 11–14. ISSN: 0018-9162. DOI: 10.1109/MC.1983.1654264.

Hachtel, G. and F. Somenzi (1996). *Logic Synthesis and Verification Algorithms*. Kluwer Academic Publishers.

Harlow III, J. E. and F. Brglez (May 2001). "Design of experiments and evaluation of BDD ordering heuristics." In: *International Journal on Software Tools for Technology Transfer* 3.2, pp. 193–206. ISSN: ISSN 1433-2779.

Hassoun, S. and T. Sasao (2002). *Logic Synthesis and Verification*. Springer Science & Business Media, p. 454.

Hoos, H. H. and T. Stützle (2007). "Empirical analysis of randomized algorithms." In: *Handbook of Approximation algorithms and metaheuristics*. Ed. by T. F. Gonzalez. Boca Raton (FL): Chapman & Hall, pp. 14.1–14.17.

Kragh, H. (2000). "Max Planck: the reluctant revolutionary." In: *PhysicsWorld*.

Kubalík, P., P. Fišer, and H. Kubátová (2006). "Fault Tolerant System Design Method Based on Self-Checking Circuits." In: *Proc. 12th International On-Line Testing Symposium 2006 (IOLTS'06)*. Lake of Como, Italy, pp. 185–186.

McCluskey, E. (Nov. 1956). "Detection of group invariance or total symmetry of a Boolean function." In: *Bell System Technical Journal, The* 35.6, pp. 1445–1453. ISSN: 0005-8580. DOI: 10.1002/j.1538-7305.1956.tb03836.x.

McElvain, K. (1993). *LGSynth93 benchmark set: Version 4.0*.

McGeoch, C. (1996). "Toward an experimental method for algorithm simulation." In: *INFORMS J. Comput.* 1.1, pp. 1–15.

Moret, B. (2002). "Towards a discipline of experimental algorithmics." In: *Data Structures, Near Neighbor Searches, and Methodology: Fifth and sixth DIMACS implementation challenges*. Ed. by M. Goldwasser, D. Johnson, and C. McGeoch. AMS, p. 15.

Puggelli, A. et al. (2011). "Are Logic Synthesis Tools Robust?" In: *Proc. of the 48th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 633–638.

Rawlins, D. (1993). "Tycho's 1004-Star Catalog." In: *DIO, the International Journal of Scientific History* 1.3.

Ross, T. D., M. J. Noviskey, and D. A. Taylor Timothy N. ;and Gadd (1991). *Pattern Theory: An Engineering Paradigm for Algorithm Design*. Final Report ADA243214.

Schmidt, J. (2015). "Performance Problems in Logic Synthesis." Habilitation thesis.

Selman, B., D. Mitchell, and H. Levesque (1992). "Hard and Easy Distributions of SAT Problems." In: *Proceedings of Tenth National Conference on Artificial Intelligence (AAAI-92)*, pp. 459–465.

Shannon, C. E. (1949). "The synthesis of two-terminal switching circuits." In: *Bell Systems Technical Journal* 28, pp. 59–98.

Shum, W. and J. H. Anderson (2012). "Analyzing and predicting the impact of CAD algorithm noise on FPGA speed performance and power." In: *Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays (FPGA '12)*, pp. 107–110.

Yang, S. (Jan. 1991a). *Logic Synthesis and Optimization Benchmarks User Guide*. Technical Report 1991-IWLS-UG-Saeyang. Research Triangle Park, NC: MCNC.

— (Jan. 1991b). *Logic Synthesis and Optimization Benchmarks User Guide: Version 3.0*. Tech. rep. MCNC Technical Report.

Ing. Jan Schmidt, Ph.D., 9. 7. 1953

Qualifications

2001 CTU in Prague, Prague 2, PhD. in informatics

1976 CTU in Prague, Prague 2, Ing. in cybernetic technology

1971 SVVŠ W. Piecka, Prague 2

Employment History

1976–1989 CTU in Prague, Prague 2, Faculty of Electrical Engineering; *computer technician*. Research in electronic design automation, computer maintenance.

1989–2009 CTU in Prague, Prague 2, Faculty of Electrical Engineering; *assistant professor*. Research in electronic design automation and cryptographic hardware, teaching computer engineering courses; a 6-months stay at University of Loughborough, UK, in 1991; research in knowledge representation.

2009–recent CTU in Prague, Prague 6, Faculty of Information Technology; *assistant professor*. Research of design, verification and testing of digital devices, teaching courses in the area of digital design.

Publications

2014

Balcárek, J., P. Fišer, and J. Schmidt (Nov. 2014a). "On don't cares in test compression." In: *Microprocessors and Microsystems (MICPRO)* 38.8, pp. 754–765.

— (2014b). "PBO-Based Test Compression." In: *Proc. of 17th Euromicro Conference on Digital Systems Design (DSD)*. Verona (Italy).

Fišer, P. and J. Schmidt (Apr. 2014a). "Permuting Variables to Improve Iterative Resynthesis." In: *Recent Progress in the Boolean Domain*, pp. 213–230.

— (2014b). *The Logic Synthesis Homework Is Not Done Yet*. invited talk at 10th Italian Annual Seminar Day on Logic Synthesis.

Fišer, P., J. Schmidt, and J. Balcárek (2014a). "On Robustness of EDA Tools." In: *Proceedings 17th Euromicro Conference on Digital Systems Design (DSD)*. Verona (Italy), pp. 427–434.

Fišer, P., J. Schmidt, and J. Balcárek (2014b). "Sources of Bias in EDA Tools and Its Influence." In: *Proc. of 17th IEEE Symposium on Design and Diagnostics of Electronic Systems (DDECS)*. Warsaw, Poland, pp. 258–261.

Pospíšil, J., T. Vaňát, and J. Schmidt (2014). "Towards Trusted Devices in FPGA by Modeling." In: *Proceedings of the 2nd Prague Embedded Systems Workshop*, p. 16.

Schmidt, J., R. Blažek, and P. Fišer (2014). "On Probability Density Distribution of Randomized Algorithms Performance." In: *Proc. of 11th Int. Workshop on Boolean Problems (IWSBP)*. Freiberg (Germany), pp. 67–74.

2013

Balcárek, J., P. Fišer, and J. Schmidt (2013a). "Simulation and SAT Based ATPG for Compressed Test Generation." In: *Proc. of 16th Euromicro Conference on Digital Systems Design (DSD)*. Santander (Spain), pp. 445–452.

— (Mar. 2013b). "Techniques for SAT-based Constrained Test Pattern Generation." In: *in Microprocessors and Microsystems (MICPRO)* 37.2, pp. 185–195.

Pospíšil, J., J. Schmidt, and P. Fišer (2013). "New SEU Modeling by Architecture Analysis." In: *Proc. Work in Progress Session of 16th Euromicro Conference on Digital Systems Design (DSD)*. Santander (Spain).

Schmidt, J., P. Fišer, and J. Balcárek (2013). "The influence of implementation type on dependability parameters." In: *Microprocessors and Microsystems (MICPRO)* 37.6-7, pp. 641–648.

2012

Fišer, P. and J. Schmidt (Apr. 2012a). "Improving the iterative power of resynthesis." In: *Design and Diagnostics of Electronic Circuits Systems (DDECS), 2012 IEEE 15th International Symposium on*, pp. 30–33. DOI: 10.1109/DDECS.2012.6219019.

— (2012b). "On Using Permutation of Variables to Improve the Iterative Power of Resynthesis." In: *Proc. of 10th Int. Workshop on Boolean Problems (IWSBP)*. Freiberg (Germany), pp. 107–114.

Schmidt, J., P. Fišer, and J. Balcárek (2012a). "Generalized Miter and its Application in Hardware Design." In: *in Proc. of the 8th Doctoral Workshop on Mathematical and Engineering Methods in Computer Science (MEMICS)*. Znojmo (ČR), pp. 119–120.

Schmidt, J., P. Fišer, and J. Balcárek (2012b). "The Influence of Implementation Technology on Dependability Parameters." In: *Proc. of 15th Euromicro Conference on Digital Systems Design (DSD)*. Cesme (Turkey), pp. 368–373.

2011

Balcárek, J., P. Fišer, and J. Schmidt (2011a). "Implicit Techniques for Constrained Test Patterns Generation." In: *Proc. of Annual Doctoral Workshop on Mathematical and Engineering Methods in Computer Science (MEMICS)*. Lednice (ČR), p. 106.

— (2011b). "Techniques for SAT-based Constrained Test Pattern Generation." In: *Proc. of 14th Euromicro Conference on Digital Systems Design (DSD)*. Oulu (Finland), pp. 360–366.

Fišer, P. and J. Schmidt (2011). "How Much Randomness Makes a Tool Randomized?" In: *Proc. of the 20th International Workshop on Logic and Synthesis (IWLS)*. San Diego, California, USA, pp. 136–143.

2010

Balcárek, J., P. Fišer, and J. Schmidt (2010a). "Implicit Representations in Test Patterns Compression for Scan-Based Digital Circuits." In: *Proc. of Informal Proceedings of European Test Symposium (ETS)*. Prague (CR).

— (2010b). "Test Patterns Compression Technique Based on a Dedicated SAT-based ATPG." In: *Proc. of 13th Euromicro Conference on Digital Systems Design (DSD)*. Lille (France), pp. 805–808.

Fišer, P. and J. Schmidt (2010a). "It Is Better to Run Iterative Resynthesis on Parts of the Circuit." In: *Proceedings of 19th International Workshop on Logic and Synthesis*, pp. 17–24.

— (2010b). "New Ways of Generating Large Realistic Benchmarks for Testing Synthesis Tools." In: *Proc. of 9th Int. Workshop on Boolean Problems (IWSBP)*. Freiberg, Germany, pp. 157–164.