U.P.B. Sci. Bull., Series C, Vol. 85, Iss. 4, 2023 ISSN 2286-3540

BENCHMARKING A SMART FRAMEWORK FOR REDUCING THE COVERAGE CLOSURE TIME IN ASIC FUNCTIONAL VERIFICATION

Mihai-Corneliu CRISTESCU[1]

This paper outlines a solution that estimates and reduces the functional coverage closure time in integrated circuit verification by leveraging the power of artificial neural networks. This way, the process time cost is minimized by driving novel scenarios and discarding redundant test sequences. Practically, the article highlights a proof of concept by evaluating a smart framework that reduces stimuli redundancy. Some of the most common functional coverage models are considered for benchmarking the proposed framework and the results indicate a significant reduction in the total number of stimuli packets applied to the inputs of the integrated circuit.

Keywords: Integrated circuit, functional verification, artificial neural network, stimuli redundancy reduction, proof of concept, functional coverage feedback

1. Introduction

Since the creation of the first transistor in 1947, the semiconductor industry continuously faced ever-growing demands for faster, more power-efficient, and even smaller-sized integrated circuits. These improved product requirements have always brought new challenges when addressing the complexity of the resulting *systems-on-chip* (SoC). In addition, the needed functions of the integrated circuits have become even more complicated and interdependent [1]. These aspects keep increasing the chances of inserting unintentional errors when designing the circuits.

The pre-silicon functional verification phase of the SoC development process is required to point out the existing functional errors. These faults can emerge as either *register-transfer level* (RTL) implementation bugs or even architectural bugs that are traced back to the specification document [2]. As the functional verification tasks require a remarkably high degree of execution quality, the resources needed for identifying all design errors have increased dramatically. At one point, such tedious tasks cannot be further parallelized by involving more verification engineers. Thus, the closure time of the verification phase will continue to increase and push the product tape-out date even further. In many cases, this

[1] Eng., AMIQ Consulting, Bucharest, Romania,
Ph.D. Student, Doctoral School of ETTI-B, University POLITEHNICA of Bucharest, Romania,
e-mail: mihai.cristescu@amiq.com, mihai.cristescu2303@stud.etti.upb.ro

delivery delay determines a considerable cost to reach the market expectations, and, in some instances, the market opportunity is even lost.

Within the *artificial intelligence* (AI) family of process optimization methods, the subdomain of *machine learning* (ML) algorithms underwent outstanding research that provided breakthrough achievements across many industries during the past decade [3]. *Artificial neural networks* (ANN) form a set of supervised ML techniques that are suitable for minimizing and automating a large variety of processes [4]. Because the ANNs can model some complex or even unknown functions, they can be found at the center of many expert systems [5].

During the past decade, the research community devised several techniques to alleviate the scalability and reusability challenges brought by the latest *application-specific integrated circuit* (ASIC) functional verification tasks [6], [7]. However, for complex and heterogeneous SoC verification objectives, these improvements seem not to effectively reduce the overall verification effort, whereas the functional coverage closure time remains lengthy and resource costly. In consequence, this kind of challenge requires process minimization techniques that should optimally reduce the time cost with similar resource allocation efforts. Thus, different research groups tackled this problem and developed concepts to harness the synergy points between functional verification and AI. These are widely recognized as *intelligent verification* (IV) strategies [8]. Besides, a discrepancy is that the former requires exact computations that ensure design correctness, whilst the latter provides results with unideal accuracy and precision. However, synergism proves possible because both functional verification and ML algorithms do not rely on deterministic models, but rather on probabilistic methods that harvest the highlights of randomness in discrete computational mathematics.

2. Previous Work

The IV strategy that outlined the most promising research interest in article [8] is *Automated Directed Test Generation using Scenario Coverage Feedback*. The core idea to intelligently optimize the regression runtime has a significant and direct impact on reducing the time costs for many types of functional verification tasks. In this sense, Cristescu and Bob [9] propose an ML-assisted flexible framework that can perform *stimuli redundancy reduction* (SRR) by harnessing the power of ANNs. The main goal of this new paper is to benchmark the SRR-based *smart framework* (SF) by evaluating different coverage models and measuring the performance of the stimuli reduction engine.

Cristescu and Bob [9] describe in detail the concept of reducing stimuli redundancy for successfully optimizing the test regression runtime. The solution features a supervised-learning algorithm that comprehends the ASIC’s transfer functions between the input stimuli packets and the target coverage items. This way, the smart tool can model and use the ASIC's inverse sampling functions on the coverage feedback loop. Even so, the smart engine can interpret the current coverage result and then predict a novel stimuli sequence that covers a missing scenario [10]. From an implementation point of view, during both learning phases, the coverage data is used as input for the ANN, while stimuli data is collected from the output layer neurons. In terms of scenario modeling, the stimuli sequences contain a single item field that represents the 32-bit address data of the target ASIC.

Practically, feed-forward *multilayer perceptrons* (MLP) are considered for undergoing initial evaluations of the SF [9]. Depending on the number of bins that establish the coverage item, the input layer of the ANN is sized so that each input neuron corresponds to a separate coverage bin.

Moreover, regardless of the target verification task, the SF is configured to deploy the learning steps with some fixed hyperparameter values. Therefore, during the training phase, the SF uses 400 learning epochs that provide sufficient iterations to optimally adjust the synaptic weights. In addition, each batch is sized at 10 training examples/batch which provides the best tradeoff between the execution runtime and the final learning accuracy.

During the inference phase, all performance metrics are computed and carefully monitored. Once the coverage rate reaches 100%, the learning process is suspended, and the final performance results are logged.

A great advantage of the SF introduced in article [9] is the usage of the powerful Keras API within the TensorFlow library [11] that features many user-friendly hooks that enable design exploration. Another important advantage is the stand-alone learning process that is performed in an offline approach without the need of an ASIC logic simulator [12], [13]. This way, the learning exploration is run seamlessly without having to interrupt the simulator for costly data collection.

3. The Leveraged Artificial Neural Networks

For assessing the SRR proof of concept, the proposed SF is evaluated for reaching some of the most common functional coverage goals. For the target ASIC, which is a sequential de-multiplexer, the objective is to verify the addressing logic by collecting interesting addresses sent on the *design under test* (DUT) input.

An exhaustive verification approach is to cover all possible address values that can be driven on the data bus. Nevertheless, because the bus width is 32 bits, the number of possible address values is very large and is computed in (1). Reaching coverage closure for a *coverpoint* that has individual bins for each address value is unfeasible and is avoided in almost all verification processes. However, a common approach is to define a relatively small list of interesting address values that need

$$bin_i \leftrightarrow i \quad , i \in [0 : 2^{32} - 1] \cap \mathbb{N} \tag{1}$$

to be captured. Therefore, the following representative functional coverpoints are proposed and implemented using the SRR-based SF:

- Linear coverpoint with “N” equally distributed intervals.
- Nonlinear coverpoint with “power-of-two” distributed intervals.
- Nonlinear coverpoint with individual “min” and “max” bins.

The learning metrics obtained for each of these case studies are depicted and interpreted in the articles [9] and [14]. To obtain a learning configuration that optimally solves each of the aforementioned coverage tasks, the learning algorithm undergoes a consistent phase for design space exploration. This is achieved by deploying several hyperparameter combinations until the learning performance metrics indicate an optimal solution.

The hyperparameters and their value ranges are outlined in article [9] and after finishing the hyperparameter analysis step, the optimum results are manually identified and depicted in figure 1 for each of the three coverpoint distributions.

For the linear coverpoint, one can observe that only 20 epochs are needed for reaching coverage closure. However, for the nonlinear coverpoint with “power-of-two” distribution, after 312 learning epochs, the model’s accuracy reaches 100%,

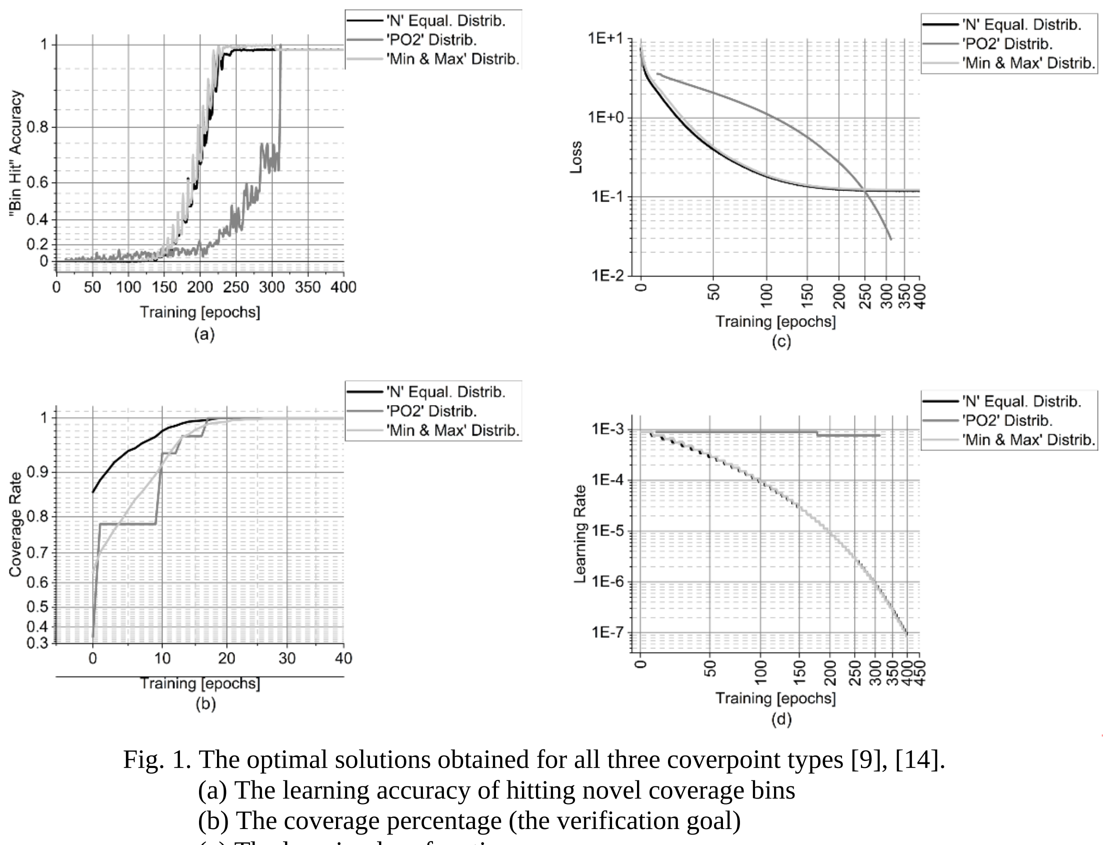

Fig. 1. The optimal solutions obtained for all three coverpoint types [9], [14].
(a) The learning accuracy of hitting novel coverage bins
(b) The coverage percentage (the verification goal)
(c) The learning loss function

but the coverage rate significantly increases from 34.38% to 100% in just 17 epochs. Compared to the linear coverpoint with 1000 distributed intervals, this nonlinear coverage item has only 32 bins, which requires a smaller number of learning epochs to reach coverage closure.

For the nonlinear coverpoint with “min” and “max” bins, the performance results are similar to those of the linear coverpoint with 1000 equally distributed intervals. Practically, the model’s accuracy reaches 99.8% after 251 learning epochs, but the coverage rate increases from 63.77% to 99.9% after only 26 epochs. This indicates that one of the two bins with low probability was not hit during any of the inference phase iterations.

In all cases, the intention is to use an ANN to learn the inverse sampling function of the respective coverage model. Since the structure of the ANN depends on the coverpoint size, each of these three use cases features singly different ANN layouts. Precisely, the input layer of the ANN has a number of neurons equal to the number of coverpoint bins. An ANN with an input layer size of 32 is depicted in figure 2. Before listing the learning performance results in articles [9] and [14], several ANN structures were evaluated, and the analysis pointed out that ANNs with a hidden layer size of at least 5 neurons can emulate the inverse sampling function with learning accuracies of at least 90%.

The ANNs used in these case studies are designed as sequential models using the Keras API. Practically, for leveraging best learning capabilities using the backpropagation methodology, an implementation decision is to have fully connected neurons [5]. To achieve this, the hidden and the output layers are implemented using the “dense layer” Keras model. Thus, as illustrated in figure 2,

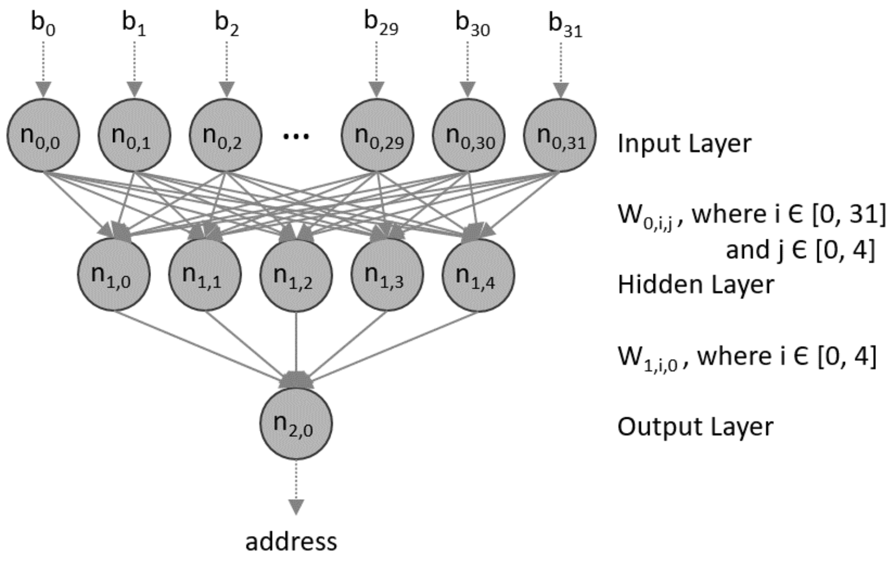

Fig. 2. The structure of the artificial neural network used for performing SRR within the SF

for a coverpoint with a size of 32 bins, the corresponding ANN features 160 synaptic weight, denoted $w_{0,i,j}$, between the input layer and the hidden layer and 5 synaptic weights, denoted $w_{1,i,j}$, between the hidden layer and the output layer.

Regarding the learning process, the objective is to reduce the complexity when designing the ANN for each case study without impacting the learning accuracy results. Practically, during the design exploration phase, I measured the learning metrics for different hidden layer configurations. The most stable activation function proved to be the *Rectified Linear Unit* (ReLU) compared to other options like the *SoftMax* or the *Sigmoid* functions. Moreover, the analysis showed that the most efficient learning optimizer is the *Adaptive Moment Estimation* (Adam) since it provides better accuracy and faster convergence compared to the *Stochastic Gradient Descent* (SGD) model.

4. Benchmarking Results

To demonstrate the SRR effect and how the coverage closure time is significantly reduced, the paper introduces a benchmarking model that compares the number of driven stimuli packets between the typical *constraint-driven verification* (CDV) approach and the novel SRR-based SF.

A. Benchmarking Model

The benchmarking model does not use any third-party component and it is fully designed as a feature of the Python-based SF. The goal is to leverage this tool to achieve at least the same functional coverage rate as in the typical CDV methodology, but with driving less stimuli packets on the DUT's inputs.

For benchmarking purposes, it is sufficient to emulate the DUT's sampling function using a dedicated Python module. Practically, the target functional coverage task is isolated from the rest of the complex verification environment and replicated inside the pure Python environment. Thus, no logic simulation is required as this would create a significant processing penalty during each learning epoch.

For emulating the CDV approach, as over-constraining stimuli is typically a bad practice, the SF model uses a pure random-number generator with a uniform probability distribution. The intention is to have a Python coarse model that emulates the randomization behavior within the *Universal Verification Methodology* (UVM).

For the proposed novel approach, the SF replaces the pure random-number generator with an ANN that is trained using an initial data set. This training set is also generated using a dedicated Python module that leverages the emulated DUT sampling function.

Regarding the testing scenario configuration, the benchmark runs many coverpoint types and sizes and it also compares cases with initial coverage rates of 33%, 50%, or 75%.

Because the learning process did not provide 100% coverage for the nonlinear coverpoint with individual *"min"* and *"max"* bins, this type of coverage distribution will not be included in the benchmarking process. For each coverpoint configuration, the benchmarking process averages out the results on 15 complete learning sessions because the learning processes can provide statistical variance.

For reducing the time cost of reaching coverage closure, it is important to have as few simulation cycles as possible. Therefore, an important metric used in the benchmarking process is the *Number of Stimuli packets that go Through the DUT* (NSTD). Moreover, the tool also measures the *Number of Stimuli packets Generated by the ANN* (NSGA). Because the main objective is to reduce the costly DUT simulation cycles, the main priority of this benchmarking process is to identify the coverpoint configurations that minimize the values for the NSTD performance metric.

B. Benchmarking the Linear Coverpoint with *"N"* Equally Distributed Intervals

For this type of coverage item, the SF benchmarking process covers many coverpoint configurations with different values for *"N",* ranging from 64 bins up to 8192 bins. After all the coverpoint configurations are deployed, the benchmarking results depicted in figure 3 point to a considerable reduction of the total NSTD for this use case. Focusing on the SF results, chart '(a)' indicates how the total NSTD values increase together with the size of the coverpoint. This is an expected behavior since a larger coverpoint size requires covering a larger number of scenarios. Also, the NSTD values are usually smaller when the initial coverage rate is at 33%. On the one hand, when the initial training set is obtained using the inefficient CDV technique for an initial 33% coverage, the majority of 66% bins that remain are addressed using the efficient SRR-based SF. On the other hand, when the initial coverage rate of 75% is inefficiently reached using the CDV approach, the minority of 25% bins that remain are covered using the SF.

As the configurations with an initial coverage rate of 33% generate the best NSTD results, a comparison between the CDV approach and the SF solution is captured in chart '(b)'. It can be observed that the NSTD values obtained with the SF are significantly smaller than the corresponding NSTD values obtained with the classic CDV approach. In addition, the difference increases together with the size of the coverpoint.

Chart '(c)' indicates how many times the SF NSTD is smaller compared to the CDV NSTD and the configurations with a higher coverpoint size provide better

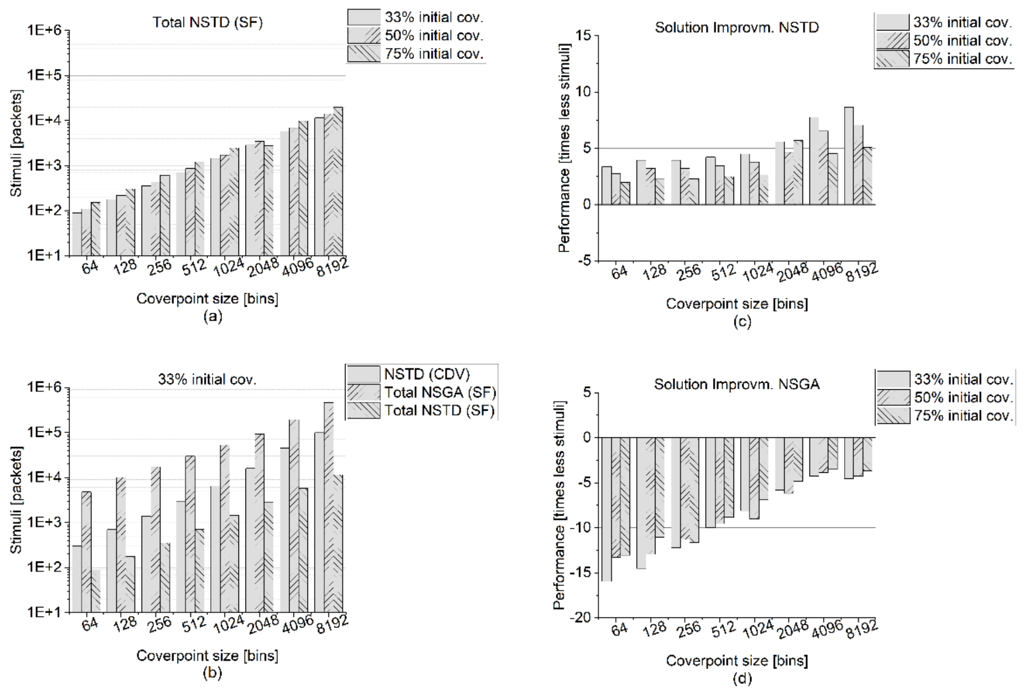

Fig. 3. A comparison of the benchmarking results for several coverpoint configurations with "N" equally distributed intervals.

(a) – The total NSTD values obtained using the SF

(b) – Comparison between CDV and SF with 33% initial coverage

(c) – NSTD performance results of the SF compared to CDV

(d) – NSGA performance results of the SF

NSTD improvements. Moreover, the performance variances across different initial coverage rates also increase together with the size of the coverage item.

Since the SF NSGA is larger compared to the CDV NSGA, chart '(d)' indicates only negative performance results. Like the NSTD metric, the NSGA ratio improves together with the size of the coverpoint. Practically, the NSGA overhead decreases, as the size of the coverage item increases. Still, configurations that have higher initial coverage ratios have better NSGA ratios.

Consequently, the best NSTD ratio is obtained when the initial coverage rate is at 33% because the SF is involved more in generating novel scenarios that reduce stimuli redundancy. Despite that, the best NSGA ratio is captured when the initial coverage rate is at 75% because the ANN gets involved only for the remaining 25% uncovered bins. Thus, this analysis uncovers a tradeoff between the main two performance metrics, and it can be observed that when the NSTD ratio is best, the NSGA metric is worst, and vice versa.

C. Benchmarking the Nonlinear Coverpoint with *“Power-of-Two”* Distributed Intervals

For this nonlinear coverpoint, the SF benchmarking process compares several configurations with coverage item sizes ranging from 8 bins to 32 bins. After measuring the performance metrics, the evaluation outcome also indicates a significant reduction of the total NSTD. The results are depicted in figure 4, which points at least 5 times and up to 15000 times less NSTD required for reaching the same verification goal using the proposed SRR-based SF compared to the typical CDV approach.

In chart ‘(a)’, the total NSTD values increase together with the size of the coverpoint, and this behavior is like the one observed in the previous use case. Another similarity is that the NSTD values decrease together with the initial coverage rate. However, the differences between the NSTD values across different initial coverage rates are significantly higher for this type of coverpoint. The differences vary under a nonlinear characteristic that is close to the *power-of-two*

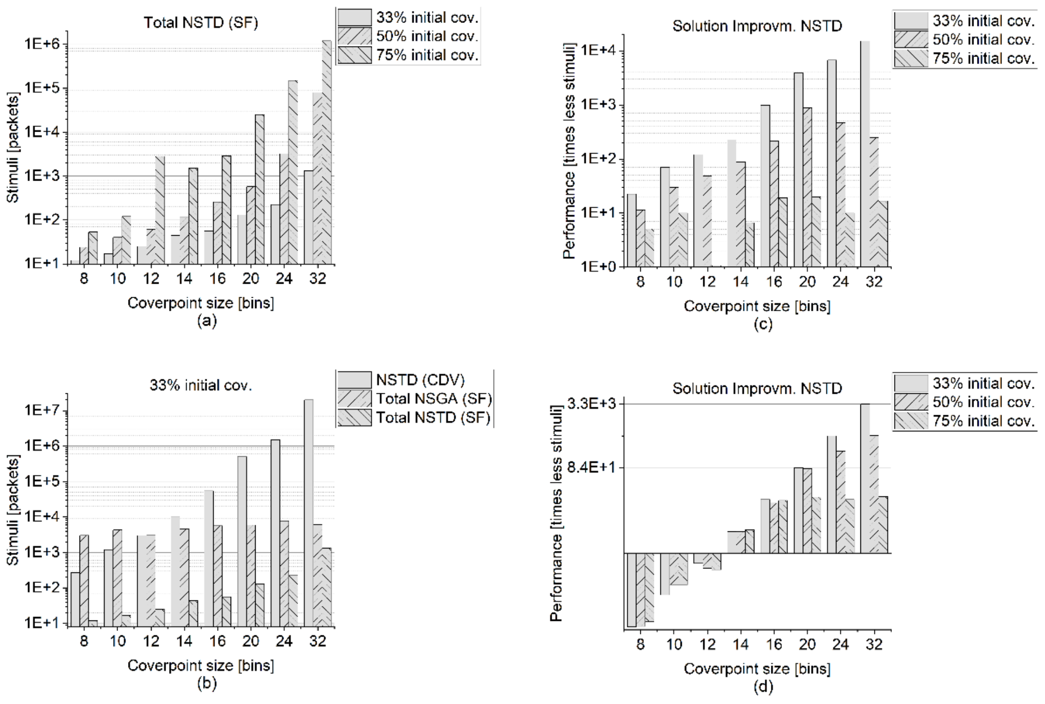

Fig. 4. A comparison of the benchmarking results for several coverpoint configurations with “power-of-two” equally distributed intervals.

(a) – The total NSTD values obtained using the SF

(b) – Comparison between CDV and SF with 33% initial coverage

(c) – NSTD performance results of the SF compared to CDV

(d) – NSGA performance results of the SF

distribution. Thus, for such a nonlinear coverpoint, it is important to carefully choose a small initial coverage rate. Similarly, the configurations with an initial coverage rate of 33% generate the best NSTD results. Therefore, chart ‘(b)’ outlines a comparison between the CDV approach and the SF solution. The great nonlinearity of the item distribution creates a heavy penalty for the CDV approach when analyzing larger coverpoints.
Like the previous use case, the NSTD differences increase together with the size of the coverpoint.

Chart ‘(c)’ depicts the NSTD performance results that increase together with the coverpoint size. In addition, the performance variances across different initial coverage rates are much larger compared to the ones obtained for the coverpoint with a linear distribution.

Chart ‘(d)’ indicates the NSGA performance results and positive results are obtained starting with a coverpoint size of 14 bins. For nonlinear coverpoints with small sizes, the ANNs have an implicit process overhead during the training epochs, which makes them less effective than the CDV approaches. Still, the variances of these NSGA results indicate a strong nonlinear behavior that increases together with the coverpoint size and decreases together with the initial coverage rate. Practically, the NSGA overhead decreases, as the size of the coverage item increases. Compared with the previous use case, a difference is that the *power-of-two* distributed coverpoint obtains better NSGA ratios for configurations that have lower initial coverage ratios.

Like the results obtained for the coverpoint with linear distribution, the best NSTD ratio is obtained when the initial coverage rate is at 33%. In contrast with the previous use case, the best NSGA ratio is also captured when the initial coverage rate is at 33%. Thus, for this type of coverage item, the recommendation is to deploy tasks on the SF that have large coverpoints sizes and low initial coverage rates.

5. Conclusions and Future Research

Within the CDV approach, functional coverage models with complex nonlinear bin distributions remain some of today’s major bottlenecks that drag the ASIC tape-out milestone. The proposed novel ML-based framework addresses this problem and significantly diminishes redundant scenario deployment. In other words, the test regression is optimized by dropping ineffective simulation cycles. This leads to faster functional coverage closure with a relatively smaller engineering overhead.

The most beneficial advantages of the smart framework are its support for process parallelization, which enables the execution of multiple ANNs at the same time, as well as its flexibility in deploying models for almost any user-defined verification task. The obtained performance results indicate great prospects for developing expert systems that can aid the *electronic design automation* (EDA) tools during future ASIC verification processes. This way, the demand for resources is reduced and the cost surge is mitigated.

As mentioned in chapter 4.A, the current implementation of the SF learning engine cannot provide full coverage rate for the use cases in which the coverpoint PMF has outliers. Thus, further investigation is planned for exploring different activation functions on the output perceptron of the ANN.

Another limitation is the implementation cost required for modeling the transfer function of the target DUT. This is an initial overhead that should be addressed before verifying an ASIC candidate using the proposed SF.

Future research directions are set on deploying the smart framework during the complete verification phase of an industry-level project. Before achieving this, further exploration of more complex ANN architectures, as well as wielding more sophisticated Keras optimizers, might be needed. Even more interesting benchmarking results could be uncovered for a larger set of coverpoints using data sets generated by industrial production settings.

Another interesting future work is to deploy nonlinear verification use cases on the AMIQ's ECTB framework [16]. Specifically, investigations are currently being held into integrating the ANN-based learning core of the SF tool into the AMIQ ECTB architecture.

One more investigation direction is to combine the strong points of both *genetic algorithms* (GA) [17] and *inductive logic programming* (ILP) [18] algorithms within a hybrid-like framework. The GA is suitable for improving the quality of the training set, while ILP could be involved in modeling complex coverage models [18]. Also, a *support-vector machine* (SVM) [19] engine could assist the SF with high-quality classification tasks.

Acknowledgement

The author wants to thank Daniel Ciupitu, Nicolae Ștefan Birman, and Cristian Bob from AMIQ Consulting, Bucharest, Romania for offering deep insights and for coordinating this research initiative.

R E F E R E N C E S

[1] *G. M. Stefan*, Let's consider Moore's law in its entirety, IEEE, 2021 International Semiconductor Conference (CAS), Oct. 2021, pp. 3-10, doi:10.1109/CAS52836.2021.9604143

[2] *A. Veneris, S. Venkataraman, I. N. Hajj, and W. K. Fuchs,* Multiple design error diagnosis and correction in digital VLSI circuits, Proceedings 17th IEEE VLSI Test Symposium, Apr. 1999, pp. 58-63, doi:10.1109/VTEST.1999.766647

[3] *V. Sze, Y.-H. Chen, T.-J. Yang, and J. Emer,* Efficient Processing of Deep Neural Networks: A Tutorial and Survey, IEEE, Mar. 2017, **vol. 12**, pp. 2295-2329, doi:10.1109/JPROC.2017.2761740

[4] *M. Mohri, A. Rostamizadeh, and A. Talwalkar*, Foundations of Machine Learning, MIT Press, 2012, ISBN:9780262018258, pp. 3-9

[5] *S. Ray*, A Quick Review of Machine Learning Algorithms, IEEE, 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Feb. 2019, pp. 35-39, doi:10.1109/COMITCon.2019.8862451

[6] *L. T. Wang, Y. W. Chang, and K. T. Cheng*, Electronic Design Automation: Synthesis, Verification, and Test (Systems on Silicon), Morgan Kaufmann, 1st edition, Feb. 2009, ISBN: 9780123743640, pp. 513-545

[7] *R. Drechsler, C. Chevallaz, F. Fummi, A. J. Hu, R. Morad, F. Schirrmeister, and A. Goryachev*, Panel: Future SoC verification methodology: UVM evolution or revolution?, IEEE, 2014 Design, Automation & Test in Europe Conference & Exhibition (DATE), Apr. 2014, p. 1-5, doi:10.7873/DATE.2014.385

[8] *M. C. Cristescu*, Machine Learning Techniques for Improving the Performance Metrics of Functional Verification, Romanian Journal of Information Science and Technology (ROMJIST), **vol. 24**, no. 1, 2021, pp. 99-116

[9] *M. C. Cristescu and C. Bob,* Flexible Framework for Stimuli Redundancy Reduction in Functional Verification Using Artificial Neural Networks, IEEE, 2021 International Symposium on Signals, Circuits and Systems (ISSCS), Jul. 2021, pp. 1-4, doi:10.1109/ISSCS52333.2021.9497443

[10] *H. W. Hsueh and K. Eder,* Test Directive Generation for Functional Coverage Closure Using Inductive Logic Programming, 2006 IEEE International High Level Design Validation and Test Workshop, Nov. 2006, pp. 11-18, doi:10.1109/HLDVT.2006.320005

[11] TensorFlow, Google, www.tensorflow.org. (URL link: 15 May 2023)

[12] *C. Stan,* CoverageLens 2.0 Release, AMIQ Cons., Dec. 2017, https://www.amiq.com/consulting/2017/12/06/coveragelens-2-0-release/. (URL link: 15th of May 2023)

[13] *C. Bob,* How to Connect SystemVerilog with Python, AMIQ Consulting, Mar. 2019, https://www.amiq.com/consulting/2019/03/22/how-to-connect-systemverilog-with-python/. (URL link: 15th of May 2023)

[14] *M. C. Cristescu and D. Ciupitu*, Stimuli Redundancy Reduction for Nonlinear Functional Verification Coverage Models Using Artificial Neural Networks, IEEE, 2021 International Semiconductor Conference (CAS), Oct. 2021, pp. 217-220, doi:10.1109/CAS52836.2021.9604141

[15] *N. S. Birman,* Functional Coverage Patterns: Bitwise Coverage, AMIQ Consulting, Sep. 2015, https://www.amiq.com/consulting/2015/09/18/functional-coverage-patterns-bitwise-coverage/. (URL link: 15th of May 2023)

[16] *A. Vintilă and S. Dudă*, AMIQ ECTB Framework, Available at: https://www.amiq.com/consulting/2023/01/30/amiq-ectb-externally-controlled-testbench-architecture-framework/. (URL link: 30th of Sep. 2023)

[17] *R. A. Zitar, A Review of the Genetic Algorithm and JAYA Algorithm Applications*, IEEE, International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, Nov. 2022, pp. 1-7, DOI:10.1109/CISP-BMEI56279.2022.9980332

[18] *Z. Zhang, L. Yilmaz, and B. Liu,* A Critical Review of Inductive Logic Programming Techniques for Explainable AI, IEEE Transactions on Neural Networks and Learning Systems, iss. 99, Apr. 2023, pp. 1-17, DOI: 10.1109/TNNLS.2023.3246980

[19] *A. Kazemi, R. Boostani, M. Odeh, and M. R. AL-Mousa*, Two-Layer SVM, Towards Deep Statistical Learning, IEEE 2022 International Engineering Conference on Electrical, Energy, and Artificial Intelligence (EICEEAI), Nov. 2022, pp. 1-6, DOI: 10.1109/EICEEAI56378.2022.10050469