NATIONAL UNIVERSITY
OF SCIENCE AND TECHNOLOGY
POLITEHNICA BUCHAREST

Doctoral School of Electronics, Telecommunications and Information Technology

Decision No. 127 from 09-11-2023

Ph.D. THESIS SUMMARY

Eng. Mihai-Corneliu CRISTESCU

ÎMBUNĂTĂȚIRI ALE METRICILOR DE PERFORMANȚĂ ÎN VERIFICAREA FUNCȚIONALĂ A CIRCUITELOR UTILIZÂND MACHINE LEARNING

PERFORMANCE METRIC IMPROVEMENTS IN CIRCUIT FUNCTIONAL VERIFICATION USING MACHINE LEARNING

THESIS COMMITTEE

Prof. Dr. Eng. Gheorghe BREZEANU National University of Science and Technology Politehnica of Bucharest	President
Prof. Dr. Eng. Gheorghe M. ȘTEFAN National University of Science and Technology Politehnica of Bucharest	Ph.D. Supervisor
Prof. Dr. Eng. Oana AMĂRICĂI-BONCALO Politehnica University of Timișoara	Referee
Prof. Dr. Eng. Petre OGRUȚAN Transilvania University of Brașov	Referee
Conf. Dr. Eng. Radu HOBINCU National University of Science and Technology Politehnica of Bucharest	Referee

BUCHAREST 2024

_____

Contents

Chapter 1

Introduction

1.1 Problem Description: High Resource Costs of ASIC Pre-Si Verification

In *application-specific integrated circuit* (ASIC) development, the *simulation-based pre-Silicon verification* process is popular due to its flexibility and features that help engineers achieve the scenario coverage goals. The desire is to reach all these goals with a minimal time cost, but this verification approach proves to be demanding and tedious. This is mostly caused by redundant scenarios generated by the randomization mechanisms of today's functional simulators.

1.2 Motivation

Major verification challenges are caused by iterative or manual procedures that require the verification engineers' input. Using today's most popular verification strategies and methodologies, such bottlenecks have been observed:

1. *Problem 1: Manual semantic extractions from the specification document are prone to unintentional error insertions.*
2. *Problem 2: Covering the remaining few percentages in the functional coverage metric can require unaffordable time and resource costs.*
3. *Problem 3: Manual simulation debugging for identifying the failure's root cause can require unacceptable time and resource costs.*
4. *Problem 4: Completion of complex function proving in formal verification can require unaffordable time costs.*

As data mining enables valuable process optimizations during the past decade [Don22], my main research interest is harnessing yet hidden synergies between functional verification and *artificial intelligence* (AI). These specific *synergy points* are known as *intelligent verification* (IV) approaches.

1.3 Scope of the Doctoral Thesis

The objective of this doctoral thesis is to research and identify potential solutions for alleviating the effects of the bottlenecks outlined in subchapter 1.2. After inquiring into different industrial reports on *integrated circuit* (IC) manufacturing challenges [LZC+14], my investigation indicated that the costliest bottleneck in the IC development industry is the one pointed out by *Problem 2*. Practically, the time cost should be reduced using a *state-of-the-art* strategy for performing process optimization. Therefore, the solutions proposed in this research thesis introduce a *methodology* and a *framework* that successfully filter out redundant stimuli packets.

1.4 Content of the Doctoral Thesis

Chapter 2 outlines a literature review, with a high emphasis on state-of-the-art synergies and smart approaches for reducing both resource allocations and time costs in ASIC functional verification. Specifically, AI-assisted engines that optimize key processes are analyzed. In the last few paragraphs, the associated highlights and lowlights are compared to identify the most promising strategy.

Chapter 3 presents the theoretical fundamentals of the prime *machine learning* (ML) approaches identified in Chapter 2. The smart optimization principle for reducing the coverage closure time using *stimuli redundancy reduction* (SRR) is introduced. I also introduce the principles of the novel *supervised learning for stimuli redundancy reduction methodology* (**SL4SRRM)** that I designed and proposed as the main implementation goal of this research.

Chapter 4 introduces the implementation details of an original *Coverage-Aware Stimuli Generation* (**CAStiG**) framework that uses the novel **SL4SRRM** methodology structure described in Chapter 3. The first subchapter specifies the target *design-under test* (DUT) involved in this research project. Key functional requirements of the DUT are selected for planning the verification process using the novel framework. The following subchapters outline the chosen technologies, other implementation approaches, and the strongpoints of the proposed AI-assisted framework. The chapter closes by outlining the structure of the *artificial neural network* (ANN) that powers the proposed **SL4SRRM**-based engine.

Chapter 5 presents the experimental results obtained after deploying a series of use cases on the novel **CAStiG** framework. Different nonlinear distribution coverage models are analyzed to measure the time reductions in reaching the coverage closure milestone. The framework is benchmarked by comparing the performance results with today's standard, the *coverage-driven verification* (CDV) approach.

Finally, yet importantly, Chapter 6 draws the conclusions and highlights the original research contributions presented in this thesis. Moreover, future perspectives and further research directions are underlined in the final few lines.

Chapter 2

Related Work

2.1 Comparison between the IV Strategies

After completing a literature review on four different IV strategies [Cri21], I list the main advantages, denoted *Adv*, and disadvantages, denoted *Dasv*, in Table 2.1. The comparison is done across four performance metrics: *the implementation effort*, *the process speeds*, *the process accuracy*, and *the solution's usability.*

Regarding the *implementation effort*, the *Scenario Coverage Feedback* features the most interesting strong points. Specifically, *data mining* (DM) methods do not require significant *domain knowledge,* while *inductive logic programming* (ILP) approaches generate data in convenient, *human-readable formats*. However, the latter requires a tedious amount of *domain knowledge,* and this lowlight is also affecting *Smart Root-Cause Analysis*.

Using the fourth IV strategy, heavy automation can be obtained, but the complex mathematical apparatus may bring modeling challenges. Meantime, for the first IV strategy, considerable effort is needed due to manual *sentence annotations*.

Concerning the *processing speed*, *genetic algorithms* (GA) are efficient in parallelizing process execution, whereas *support-vector machines* (SVM) provided much smaller improvements. Also, the formal methods reduce the execution time, but this advantage is canceled when searching for convenient hyperparameter values. Similarly, the third IV strategy reduces the debug time, but this strong point is lost when performing a dozen of data ordering steps. Lastly, the *Intelligent Requirement Extraction* approach automates constraint implementation, but the generation of the intermediate specification cancels the gain.

For *model accuracy*, the GAs are favorite because they proved to reach the global optimum. In contrast, *Bayesian networks* (BN) have low accuracies whereas DM provides inaccurate results when dealing with previously unseen examples.

The first IV strategy improves accuracy, the errors are reduced, but this is counterbalanced by manual labeling. Moreover, the *Smart Root Cause Analysis* approach can benefit from accurate SVM models only if *revision ranking* is possible.

In addition, the *solution usability metric* indicates which strategies are scalable. Specifically, for the second IV strategy, the ILP methods proves to be a *general-use* solution because it maps complex dependencies. Moreover, the SVMs

Table 2.1 *The Comparison of Different IV Strategies*

IV Strategy \ **Performance Metric**		**Implement. Effort**	**Process Speed**	**Process Accuracy**	**Solution Usability**
Intelligent Requirement Extraction *(Chapter 2.2)*	*Adv.*	-	Automatic constraints generation	Automatic semantic extraction	Avoid unintentional human errors
	Dadv.	Manual sentence labeling	Intermediate specification generation	Domain-specific labeling	IEEE documentation standard
Scenario Coverage Feedback *(Chapter 2.3)*	*Adv.*	DM: Low domain knowledge ILP: Human readable	GA: Parallelism	GA: Global optimum	SVM: Low training dependencies ILP: Complex mappings
	Dadv.	ILP: High domain knowledge	SVM: Low time reductions	BN: Initial assignments DM: Novel examples	GA: Complex coverpoints Less employed for functional coverage
Smart Root-Cause Analysis (*Chapter 2.4)*	*Adv.*	-	Narrower bug hunting space	SVM	-
	Dadv.	High domain knowledge	Data ordering Regressions (trace analysis)	Revision ranking	Unsuitable for small projects (low revision) Reference revision
Formal Intelligent Proving *(Chapter 2.5)*	*Adv.*	Heavy automation	Algorithm switch	N/A	-
	Dadv.	Complex mathematical apparatus	Hyper-parameter assignment	N/A	Not suitable for large projects

perform well when inferring completely new *mapping patterns*. Unfortunately, when deploying GAs on complex tasks, the research reports limited performance results.

The *Smart Root-Cause Analysis* strategy proves to have a quite small application scope, in which the revision system needs to be populated with hundreds of commits. Moreover, *formal methods* are known to be unusable across large projects with complex functionalities.

With respect to *Intelligent Requirement Extraction*, the highlight is that dozens of unintentional human error types can be minimized or totally avoided, but this application scope is diminished since a ridiculously small number of organizations write their specification documents according to the *IEEE 830-1998 standard* [Iee98].

Chapter 3

Theoretical Fundamentals

3.1 Supervised Learning for Stimuli Redundancy Reduction Methodology (SL4SRRM)

In the previous chapter, the IV strategy that outlined the most promising research interest for reducing the costs of IC functional verification is the *Automatic Novel Test Generation using Scenario Coverage Feedback*. The desire is to minimize the time cost by driving novel scenarios and discarding *redundant test sequences* that were previously deployed during the same test regression. This can be achieved by reducing the *stimuli redundancy rate*. To the best of our knowledge, the literature indicates two or three methods [XZB+16] for reducing the number of *test sequences*, but there is no terminology or focus on directly reducing the *stimuli redundancy rate*. Therefore, I introduced the SRR terminology in my work referenced in [CB21].

After analyzing the state of the art and harvesting the observations I made in the previous paragraphs, I continued with designing a methodology that can efficiently implement the SRR principle using various *supervised* ML models. I introduce the working principles and the implementation guidelines for the novel **SL4SRRM** methodology in my work referenced in [CB21]. From my point of view, the aim is to use an intelligent algorithm, preferably an ANN that can comprehend the DUT's *transfer functions* between the input *stimuli packets* and the target *coverage items*. This way, the novel, efficient, and scalable **SL4SRRM**-powered tool models and uses the DUT's *inverse transfer functions* on the *coverage feedback loop*.

3.1.1 Training Phase within the SL4SRRM Methodology

For the **SL4SRRM** methodology described in article [CB21], the sequence of steps for training the model is depicted in Figure 3.6 *(a)*. In this phase, the DUT is run under different scenarios generated by a usual *logic simulator*, as in a typical regression setup [Piz04]. A training example is created by pairing an input *stimuli sequence* with the associated *coverage result*. This way, the ML algorithm receives the training examples after they undergo basic preprocessing steps before they are fed into the ML engine [KKP06].

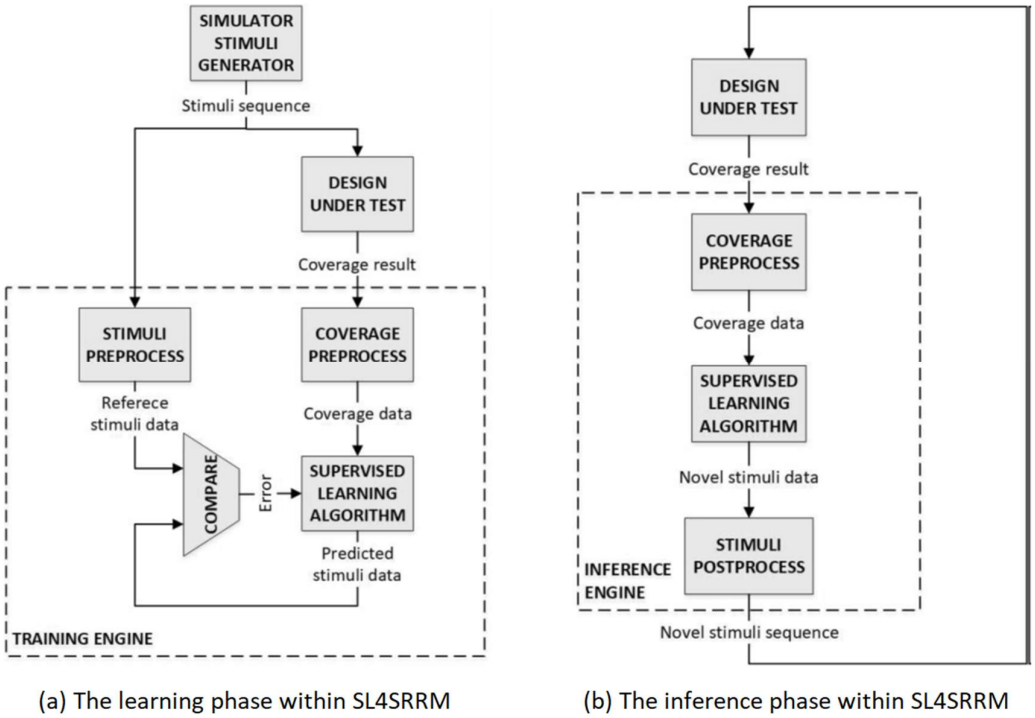

Figure 3.6 *The Learning Phases Proposed for the* ***SL4SRRM*** *Methodology [CB21]*

Depending on the implementation, the algorithm can be instructed to learn the inverse *sampling function* of the DUT's *coverage model*. This is done by selecting the coverage data as input features while the stimuli data is wielded as output labels. The algorithm predicts novel stimuli data that are compared with the reference stimuli data. For each batch [Hea15], the *neural engine* computes the prediction error that can efficiently adjust the model's *synaptic weights* in the next *training batch* [KKP06]. Moreover, the training set can be used across hundreds of *epochs* and with each new training *epoch*, the **SL4SRRM**-centred framework enriches the training set with previously unseen *corner cases*. Therefore, in this phase, the ML algorithm acts as a *training engine*.

3.1.2 Inference Phase within the SL4SRRM Methodology

The steps I propose for the **SL4SRRM** methodology during the inference phase are outlined in Figure 3.6 *(b)*. Throughout this phase, the same DUT is no longer exposed to scenarios generated by the *logic simulator*. Instead, the DUT receives the novel *stimuli sequence* output during the previous inference iteration by the trained ML model. Practically, the algorithm receives the current coverage data and based on its training experience, the *neural engine* can predict previously unseen stimuli data by modeling the *inverse sample function* of the DUT. The novel stimuli sequences have a high probability to improve the coverage results with each new inference iteration.

Before the DUT receives the *stimuli packet*, a dedicated block post processes the data and assembles a novel stimuli sequence [BF00]. Consequently, in this phase, the ML algorithm acts as an *inference engine*.

Chapter 4

SL4SRRM-based framework for *Coverage-Aware Stimuli Generation* (CAStiG)

For exploring the possibility of reducing *stimuli redundancy*, I implemented an ML-assisted *flexible framework* that can deploy functional verification use cases. In Chapter 4, I describe the techniques and the setup used for running the novel **SL4SRRM**-powered **CAStiG** framework.

4.1 Modeling the Design Under Test

For evaluating the proposed SRR *proof of concept*, the analysis focuses on verifying the selection mechanism of a *de-multiplexer*. Specifically, the user first runs test scenarios that deliver $S = 1000$ stimuli packets that target the 32-bit address values.

After this regression subset is run, the coverage database and the stimuli database are extracted from the simulation output data. The training set is obtained, and the initial results indicate a *functional coverage rate* of 85.8% for the *coverpoint* model that distributes the driven address values in N uniformly distributed intervals.

The goal of the learning engine is to obtain 100% functional coverage on the address values that are sent on the DUT input.

4.2 Common Base Technologies

Using *object-oriented* programming technologies, the *coverage items* and the input *stimuli sequences* are efficiently modeled using the highlights offered by the structural inheritance and the *polymorphism* paradigms.

When the verification engineer runs a test regression, today's *logic simulators* can store the *coverage results* in a database that complies with the UCIS format [UCI12]. This standard allows the users to merge different vendor coverage data into a unified coverage database. Therefore, the ML training set is extracted from the database with ease using dedicated utilities such as *CoverageLens* [Sta17].

4.3 The Strongpoints of the Flexible Framework

For developing complex neural engines, the research community created dedicated APIs that are *user-friendly*, offer pre-trained models, and enable fast model deployment. One such powerful library preferred for modeling deep neural networks is the *Keras API* of the *open-source TensorFlow* library [Goo15]. By using this development platform, the proposed **SL4SRRM**-powered **CAStiG** framework [CB21] avoids implementing the lowlights of the *online learning* strategy and harnesses an intelligent solution using a pure *Python* programming environment.

For seamless integration in the *learning engine*, the DUT's *coverage model* is designed using a dedicated *Python* module and another important advantage is that the *Numpy random generator* module is used for creating the initial *stimuli packets*. This means that other *Keras* modules can easily process the stimuli dataset since it is represented in a standard *Numpy binary format*.

4.4 Artificial Neural Network Architecture

According to the **SL4SRRM** methodology described in subchapter 3.1, the ANN should be part of the *coverage feedback loop* and learn the inverse *sampling function* of the DUT's *coverage model.*

As underlined in subchapter 3.1, during both learning phases, the coverage data is used as input for the ANN, while stimuli data is collected from the output layer neuron. In terms of scenario modeling, the *stimuli sequences* are designed to contain a single item field that represents a 32-bit address value for the DUT.

Multilayer feedforward perceptrons are considered for undergoing the evaluations of the **CAStiG** framework. Since the structure of the ANN depends on the target *coverpoint* size, the ANN layout is slightly different from one verification use case to another. Precisely, the input layer of the ANN has a number of neurons equal to the number of *coverpoint bins*. Thus, for a *coverpoint* with a size of $C = 8$ bins, the structure of the corresponding *multilayer perceptron* has $C = 8$ neurons on the input layer.

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 the best accuracy and the fastest convergence time.

Regardless of the target verification task, the **CAStiG** framework is configured to deploy the learning steps with fixed *hyperparameter* values. Therefore, during the training phase, the **CAStiG** framework uses $E = 400$ learning epochs that provide sufficient iterations to optimally adjust the *synaptic weights*. Moreover, each *learning batch* is sized at $B = 10$ training examples/batch which provides the best tradeoff between execution runtime and the final learning accuracy.

Chapter 5

Use-Cases and Experimental Results

For assessing the proof of concept of the novel **SL4SRRM** methodology described in subchapter 0, the proposed **CAStiG** framework introduced in Chapter 4 is evaluated for reaching some of the most common functional coverage goals of typical ASIC verification processes.

The next subchapters underline the details of how two functional *coverpoint* use cases are implemented and deployed using the **CAStiG** framework. Also, the use cases 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 DUT.

5.1 Benchmarking Model

To better indicate the SRR effect within the **SL4SRRM** methodology and how the coverage closure time is significantly reduced when deploying the **CAStiG** framework, I introduce a *benchmarking model* in paper [Cri23] that counts the number of *stimuli packets* that are driven through the DUT.

The *benchmarking model* does not use any *third-party* component and it is fully designed as a feature of the **CAStiG** framework introduced in Chapter 4. For the CDV approach, the benchmarking model uses a *pure random-number generator* with a *uniform probability distribution*. Also, the benchmark process performs a design exploration by deploying and collecting data for a dozen of *coverpoint* types and sizes. In addition, the analysis is extended by comparing cases with initial *functional coverage rates* of 33%, 50%, or 75%.

An important metric used in the benchmarking process is the *Number of Stimuli packets that go Through the DUT* (**NSTD**). In addition, since the **CAStiG** framework adds processing penalty for generating the stimuli packets, it is also interesting to measure the *Number of Stimuli packets Generated by the ANN* (**NSGA**). Thus, the main objective of the benchmarking process is to identify the *coverpoint* configurations that minimize the values for the **NSTD** metric. Since the **NSGA** indicates the processing workload that falls on the ANN, it is important to reduce these numbers as well, but with lower priority compared to the **NSTD** values.

5.2 Linear Coverpoint with *N* Equally Distributed Intervals

5.2.1 Coverpoint Model

One of the most common coverage items is the *coverpoint* that divides the entire range of address values in *N* equally distributed intervals. Considering that the addressing bus is $W = 32$ bits wide, instead of individually covering all $M = 2^W = 2^{32}$ address values, from a functional perspective, it is sufficient to consider a coverpoint with *N* equally distributed intervals across the range of integer values [0: *M*-1]. The range limits for each of these $C = N = 1000$ bins are outlined in expression (5.3). By analyzing (5.3) and because the ANN is designed to replicate the DUT's *inverse*

$$bin_i \leftrightarrow \left[i\frac{2^{32}}{1000} : (i+1)\frac{2^{32}}{1000} - 1 \right] \quad , i \in [0, 999] \cap \mathbb{N} \tag{5.3}$$

sampling behavior, the function modeled by the *supervised* learning algorithm is mentioned in equation (5.4).

$$predicted_{address} = target_{bin_{index}} \cdot \frac{2^{32}}{1000} \tag{5.4}$$

In case the bus values are generated from a *uniform distribution*, a function that indicates the probabilities of covering each bin of the *coverpoint* is the *Probability Mass Function* (PMF). It points to a uniform probability distribution across all bins and is computed in equation (5.5).

$$P(b) = \frac{M-2}{N \cdot M} = \frac{2^{32}-2}{1000 \cdot 2^{32}} \quad , i \in [0{:}\,999] \cap \mathbb{N} \tag{5.5}$$

5.2.2 Model Fitting Results

To obtain a learning configuration that optimally solves this coverage task, I deploy dozens of ANN models using different *hyperparameter* combinations until the learning performance metrics indicate an optimal solution. The *hyperparameters* and their value ranges are outlined in Table 5.1. After finishing the *hyperparameter*

Table 5.1 *Hyperparameter values used during the machine learning exploration processes for model fitting*

Hyperparameter	**Notation**	**Value Range**	
		Min.	**Max.**
Initial learning rate	*initial_lr*	1.00E-06	1.00E-03
Learning rate decay steps	*decay*	1.00E+03	1.00E-05
Decay rate	*decay_lr*	0.5	0.99

analysis step, the optimum results are manually identified and depicted in Figure 5.2.

One can observe that the model's *accuracy* of hitting a novel coverage *bin* increases rapidly after performing 150 learning *epochs*. After interpreting this optimal solution, the observations are listed in Table 5.2. One can see that only 20 *epochs* are needed to reach coverage closure. However, the other learning metrics obtain optimal values well beyond the 20 learning epochs threshold.

5.2.3 Benchmarking Results

For this type o coverage item, the **CAStiG** framework benchmarking process covers a dozen of coverpoint configurations with different values for *N*, ranging from *C* = 64 bins up to *C* = 8192 bins. Compared to the classic CDV approach, the objective is to

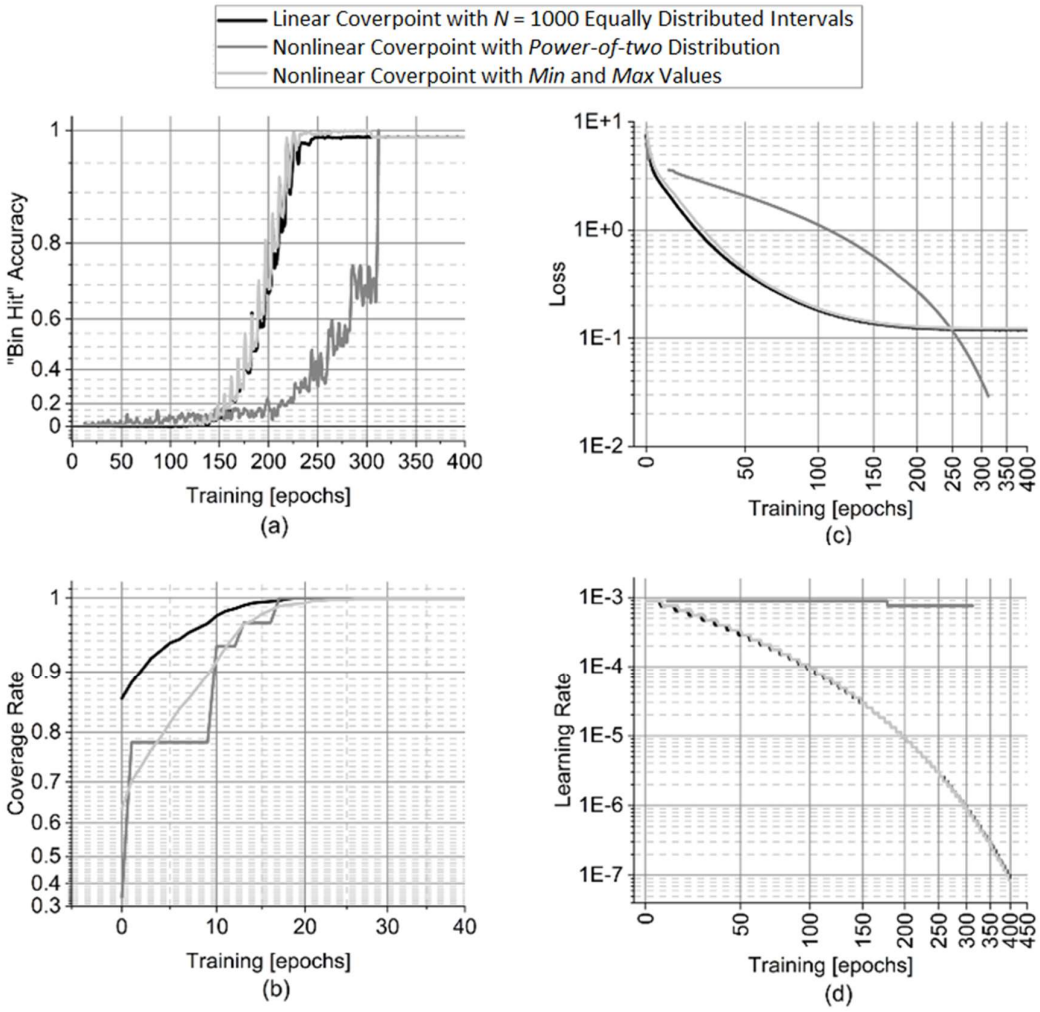

Figure 5.2 *The optimal learning configurations obtained for all three coverpoint use cases*
(a) The learning accuracy of hitting novel coverage bins
(b) The coverage percentage (the verification goal)
(c) The learning loss function
(d) The learning rate function

Table 5.2 *Learning performance results for the linear coverpoint with N = 1000 equally distributed intervals*

Metric	Value	Number of Epochs[a]
Accuracy	98.51%	240
Coverage	100%	20
Loss	0.14	250
Learning rate	8.50E-08	400

a. The number of learning epochs after which the presented value is reached

see if the **CAStiG** framework manages to reduce the total number of stimuli packets required for reaching coverage closure.

After all the coverpoint configurations are deployed and after all the associated performance metrics are measured, the benchmarking results point to a considerable reduction of the total **NSTD** for this use case. The most relevant benchmarking statistics are mentioned in Table 5.3.

For indicating the performance improvements, the tenth column outlines the reduction of the **NSTD** using the **CAStiG** framework compared to the CDV strategy. Precisely, at least two times and up to 8.6 times less **NSTD** are sufficient for achieving the same verification goal using the proposed **SL4SRRM**-based **CAStiG** framework compared to the classic CDV approach.

For better visualizing the benchmarking results, the most important data outlined in Table 5.3 is used for creating the charts depicted in Figure 5.3. Focusing on the **CAStiG** framework results, chart *(a)* indicates how the total **NSTD** values increase together with the size of the *coverpoint*.

As the configurations with an *initial coverage rate* of 33% generate the best **NSTD** results, a comparison between the CDV approach and the **CAStiG** framework solution is captured in Figure 5.4. It can be observed that the **NSTD** values obtained with the **CAStiG** framework 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)* in Figure 5.3 depicts the results outlined in the tenth column of Table 5.3 and it can be seen that the configurations with a higher coverpoint size provide better **NSTD** improvements. Moreover, the performance variances across different *initial coverage rates* also increase together with the size of the coverage item.

The eleventh column of Table 5.3 is presented in chart *(b)* of Figure 5.3. Since the **CAStiG** framework's **NSGA** is larger compared to the CDV **NSGA**, the chart indicates only negative performance results. Like the **NSTD** ratio, 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 **CAStiG** framework is involved more in generating novel

Table 5.3 *Benchmarking results for the linear coverpoint with N equally distributed intervals*

Initial Coverage Rate [%]	Number of bins in coverpoint (*N*)	NSTD (for CDV)	NSTD (Training phase for CAStiG)	NSTD (Both phases for CAStiG)	Total NSTD (for CAStiG)	NSGA (Training phase for CAStiG)	NSGA (Inference phase for CAStiG)	Total NSGA (for CAStiG)	CAStiG Improvement NSTD [times less stimuli]	CAStiG improvement NSGA [times less stimuli]
33%	64	300	25	64	89	2099	2667	4791	**3.37**	**-15.97**
	128	700	50	128	178	4610	5504	10164	**3.93**	**-14.52**
	256	1400	100	256	356	7746	9301	17147	**3.93**	**-12.25**
	512	3000	200	512	712	12990	16718	29908	**4.21**	**-9.97**
	1024	6500	420	1024	1444	23728	28535	52683	**4.50**	**-8.11**
	2048	16000	820	2048	2868	41760	50517	93097	**5.58**	**-5.82**
	4096	45000	1700	4096	5796	87772	103219	192691	**7.76**	**-4.28**
	8192	100000	3400	8192	11592	213066	241664	458130	**8.63**	**-4.58**
50%	64	300	45	64	109	1789	2159	3993	**2.75**	**-13.31**
	128	700	90	128	218	4214	4762	9066	**3.21**	**-12.95**
	256	1400	180	256	436	7268	8380	15828	**3.21**	**-11.31**
	512	3000	360	512	872	13150	15104	28614	**3.44**	**-9.54**
	1024	6500	700	1024	1724	27241	30720	58661	**3.77**	**-9.02**
	2048	16000	1400	2048	3448	45555	51883	98838	**4.64**	**-6.18**
	4096	45000	2800	4096	6896	81080	91477	175357	**6.53**	**-3.90**
	8192	100000	6000	8192	14192	200253	219546	425799	**7.05**	**-4.26**
75%	64	300	90	64	154	1823	2005	3918	**1.95**	**-13.06**
	128	700	177	128	305	3638	3917	7732	**2.30**	**-11.05**
	256	1400	360	256	616	7680	8226	16266	**2.27**	**-11.62**
	512	3000	700	512	1212	12540	13312	26552	**2.48**	**-8.85**
	1024	6500	1450	1024	2474	20926	22528	44904	**2.63**	**-6.91**
	2048	16000	2900	2048	2800	35853	38571	77324	**5.71**	**-4.83**
	4096	45000	5800	4096	9896	73436	78507	157743	**4.55**	**-3.51**
	8192	100000	11500	8192	19692	174324	183501	369325	**5.08**	**-3.69**

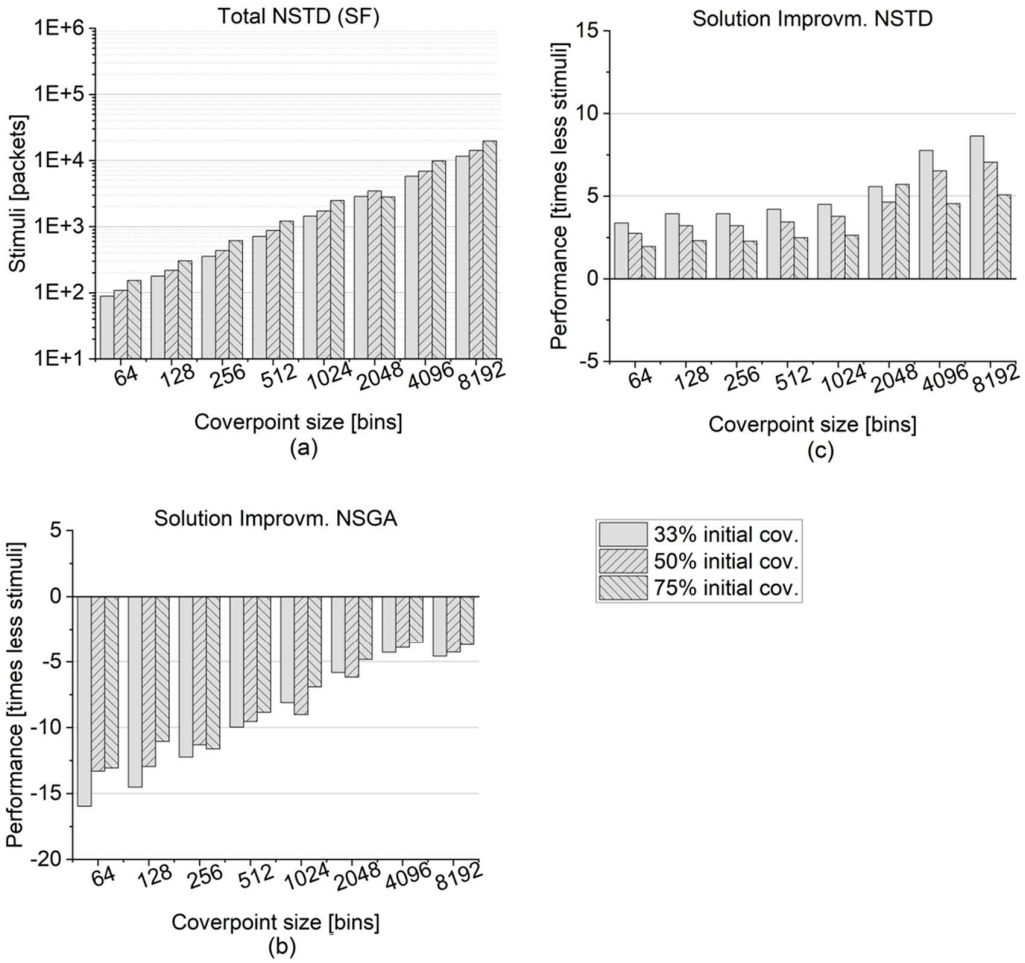

Figure 5.3 *A comparison of the obtained benchmarking results for several linear coverpoint configurations with N equally distributed intervals*
*(a) The total **NSTD** values obtained using the **CAStiG** framework*
*(b) The **NSGA** performance results of the **CAStiG** framework*
*(c) The **NSTD** performance results of the **CAStiG** framework compared to the CDV approach*

scenarios that reduce stimuli redundancy. This analysis uncovers a tradeoff between these two performance metrics and it can be seen that when the **NSTD** ratio is best, the **NSGA** ratio is worst, and vice versa.

5.3 Nonlinear Coverpoint with *Power-of-Two* Distribution

5.3.1 Coverpoint Model

A typical coverage item that uses a nonlinear grouping of the address values is the *coverpoint* with *power-of-two* distributed intervals [Bir15]. The mappings between each *bin* index and the associated range of captured values are suggested in (5.6). One

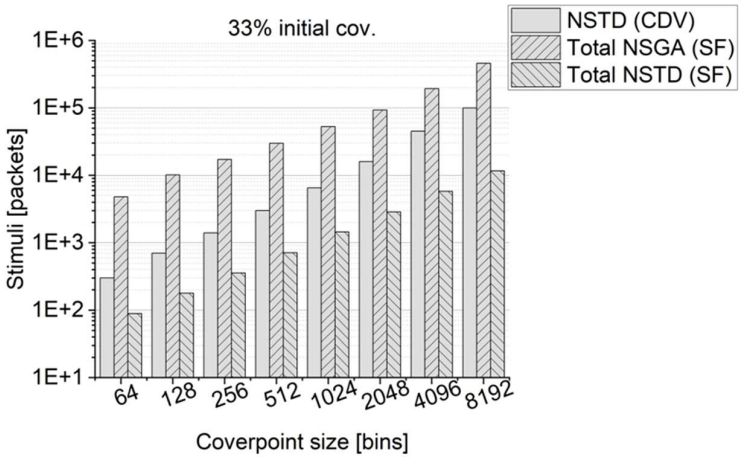

Figure 5.4 *A comparison between the CDV and the* ***CAStiG*** *framework with 33% initial coverage rate for the linear coverpoint configurations with N equally distributed intervals*

$$b_i \leftrightarrow \left[2^i \;:\; 2^{i+1}-1\right] \quad , i \in [0: N-1] \cap \mathbb{N} \tag{5.6}$$

can observe that the higher the *bin* index, the more bus values are included in that *bin*. The nonlinear PMF for this type of coverage item is defined in (5.7)

$$P(b) = P(b_i) = \frac{2^i}{2^N - 1} \quad , i \in [0: N-1] \cap \mathbb{N} \tag{5.7}$$

In this case, the probability increases together with the number of values captured in the respective *bin*.

5.3.2 Model Fitting Results

As in the previous case study, the same set of deep learning *hyperparameters* are used. Similarly, the tuning process uses the same *hyperparameter* value ranges, as presented inTable 5.1. The analysis proceeds under several iterations and Figure 5.2 shows that the learning metrics indicate interesting performance results.

After analyzing the optimal solution, the highlights are mentioned in Table 5.4. Precisely, after 312 learning *epochs*, the accuracy reaches 100%, but the coverage rate significantly increases from 34.38% to 100% in just 17 *epochs*. Compared to the

Table 5.4 *Learning performance results for the nonlinear coverpoint with "power of two" distributed intervals*

Metric	Value	Number of Epochs[a]
Accuracy	100%	312
Coverage	100%	17
Loss	0.029	312
Learning rate	7.65E-04	312

a. The number of learning epochs after which the presented value is reached

linear coverpoint with N = 1000 equally distributed intervals, this coverage item has only C = 32 bins, which requires a smaller number of learning epochs to reach coverage closure. Moreover, the learning process ends at *epoch* number 312 since the accuracy rate reaches 100% by that point.

5.3.3 Benchmarking Results

The full list of benchmarking results is outlined in Table 5.5. Since the outline of this table is similar to the profile of Table 5.3, the meaning of each column remains the same as already described in subchapter 5.2.3.

Fortunately, Table 5.5 points at least 5 times and up to 15000 times less **NSTD** required for reaching the same verification goal using the proposed **SL4SRRM**-based **CAStiG** framework compared to the typical CDV approach. Figure 5.6 has a similar

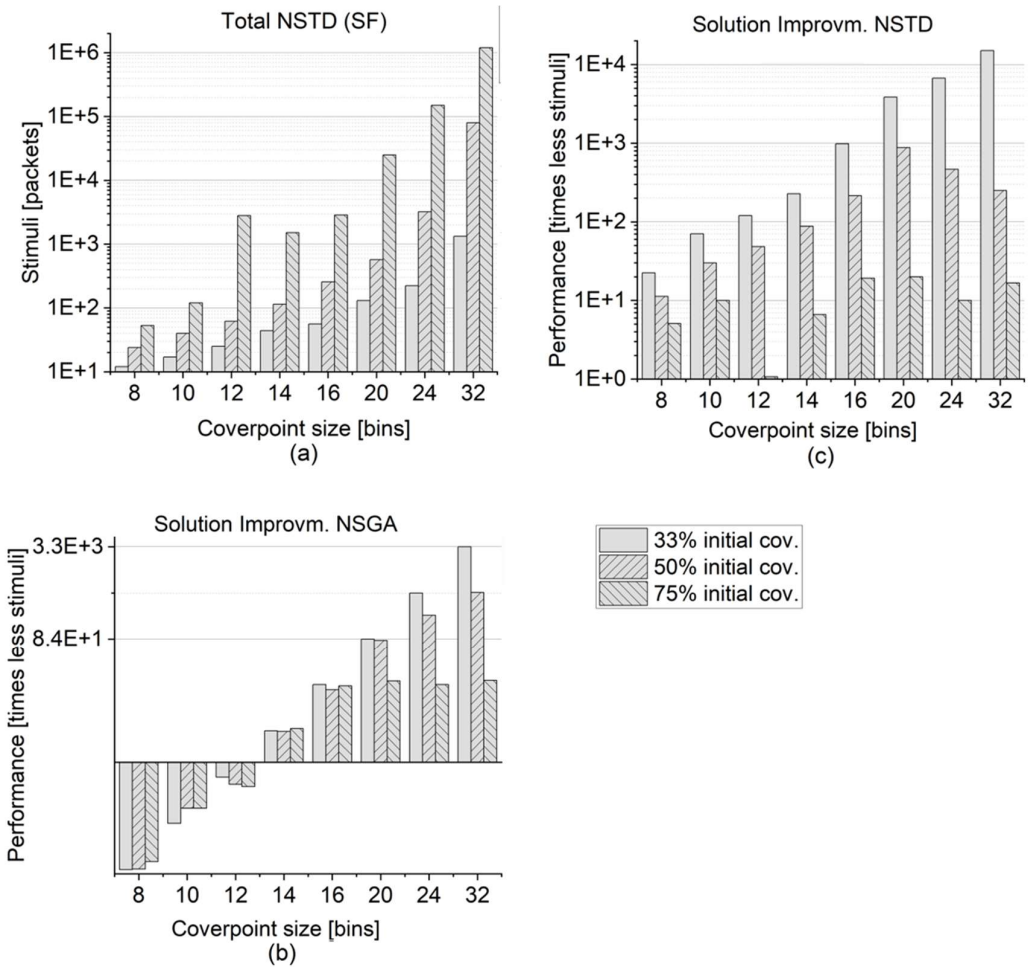

Figure 5.6 *A comparison of the obtained benchmarking results for several nonlinear coverpoint configurations with "power of two" distributed intervals*
*(a) The total **NSTD** values obtained using the **CAStiG** framework*
*(b) The **NSGA** performance results of the **CAStiG** framework*
*(c) The **NSTD** performance results of the **CAStiG** framework compared to the CDV approach*

Table 5.5 *Benchmarking results for the nonlinear coverpoint with power of two distributed intervals*

Initial Coverage Rate [%]	Number of bins in coverpoint (*N*)	NSTD (for CDV)	NSTD (Training phase for CAStiG)	NSTD (Both phases for CAStiG)	Total NSTD (for CAStiG)	NSGA (Training phase for CAStiG)	NSGA (Inference phase for CAStiG)	Total NSGA (for CAStiG)	CAStiG Improvement NSTD [times less stimuli]	CAStiG improvement NSGA [times less stimuli]
33%	32	2.0E+7	1300	32	1332	2297	2441	6038	**15,015**	**3,312**
	24	1.5E+6	200	24	224	3622	3955	7777	**6,696**	**192**
	20	5.0E+5	110	20	130	2717	3093	5920	**3,846**	**84**
	16	5.5E+4	40	16	56	2552	3067	5659	**982**	**9.72**
	14	10000	30	14	44	2112	2538	4680	**227**	**2.14**
	12	3000	13	12	25	1853	1188	3054	**120**	**-1.02**
	10	1200	8	9	17	1678	2661	4347	**70**	**-3.62**
	8	270	5	7	12	1170	1879	3054	**22**	**-11.31**
50%	32	2.0E+7	80000	32	80032	2917	3066	85983	**249**	**232**
	24	1.5E+6	3200	24	3224	3594	3874	10668	**465**	**140**
	20	5.0E+5	550	20	570	2620	2888	6058	**877**	**82**
	16	5.5E+4	240	16	256	2774	3172	6186	**214**	**8.89**
	14	10000	100	14	114	2104	2554	4758	**87**	**2.10**
	12	3000	50	12	62	1791	2430	4271	**48**	**-1.42**
	10	1200	30	10	40	1429	1867	3326	**30**	**-2.77**
	8	270	17	7	24	1257	1749	3023	**11.25**	**-11.20**
75%	32	2.0E+7	1.2E+6	32	1.2E+6	3679	3865	1.2E+6	**16.67**	**16.56**
	24	1.5E+6	1.5E+5	24	1.5E+5	1940	2032	1.5E+5	**10.00**	**9.74**
	20	5.0E+5	2.5E+4	20	2.5E+4	3460	3803	3.2E+4	**19.98**	**15.50**
	16	5.5E+4	2850	16	2866	1406	1522	5778	**19.19**	**9.52**
	14	10000	1500	14	1514	1197	1301	3998	**6.61**	**2.50**
	12	3000	300	12	2800	2015	2324	4639	**1.07**	**-1.55**
	10	1200	110	10	120	1501	1711	3322	**10.00**	**-2.77**
	8	270	45	8	53	1200	1449	2694	**5.09**	**-9.98**

layout as Figure 5.3 and it depicts benchmarking results outlined in Table 5.5.

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*.

Similarly, the configurations with an *initial coverage rate* of 33% generate the best **NSTD** results. Therefore, Figure 5.7 outlines a comparison between the CDV approach and the **CAStiG** framework solution. Like the previous use case, the **NSTD** differences increase together with the size of the *coverpoint*.

In Figure 5.6, chart *(c),* the **NSTD** improvements increase together with the *coverpoint* size, as for the linear *coverpoint*. However, the performance variances across different *initial coverage rates* are much larger for this nonlinear *coverpoint*.

Chart *(b)* indicates the **NSGA** performance outcome and positive results are obtained starting with a *coverpoint* size of $C = 14$ 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. Moreover, The **NSGA** overhead decreases, as the size of the coverage item increases.

Like the results obtained for the *coverpoint with N equally distributed intervals*, i.e., 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%.

In contrast with the previous use case, for which the benchmarking model generated negative results, the coverage closure goal for this *coverpoint* type with *power-of-two* distribution was achieved more conveniently using the **CAStiG** tool.

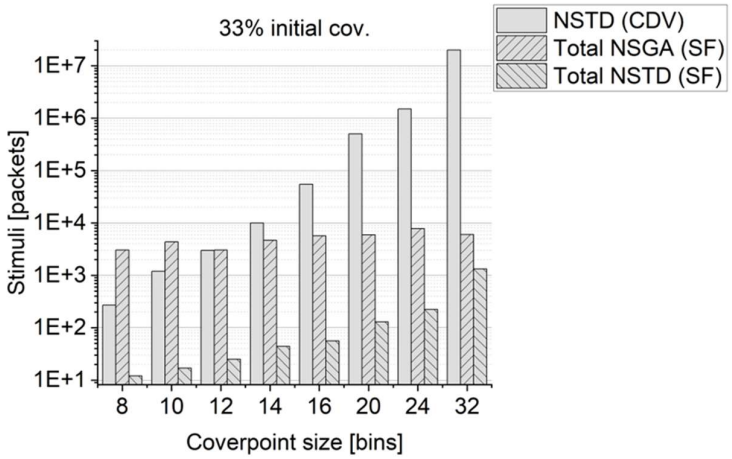

Figure 5.7 *A comparison between the CDV and the* ***CAStiG*** *framework with 33% initial coverage rate for the nonlinear coverpoint configurations with "power of two" distributed intervals*

Chapter 6

Conclusions

6.1 Obtained Results

In Chapter 2, after presenting the state of the art for the IV approaches, I compared the main strategies in Table 2.1, and I underlined the potential of automating *novel test generation using scenario coverage feedback.* This is the foundation stone for my **SL4SRRM**-powered **CAStiG** framework implementation strategy.

In Chapter 3, I highlighted the theoretical considerations for architecting the novel **SL4SRRM** methodology where I designed the basic working models for the *training* and the *inference* phases.

In Chapter 4, I describe the implementation details for my novel **SL4SRRM**-powered **CAStiG** framework, which is a tool based on the *Keras* API [Goo15]. I identified the most efficient *artificial neural network* architecture, and I uncovered that the *Adam* learning model provides the best fitting accuracy and faster convergence.

In Chapter 5, I selected two of the most common *coverpoint* distributions, and I performed a benchmarking process that indicates significant reductions in the total number of simulation cycles. I introduce the **NSTD** and the **NSGA** performance metrics, and the most promising results indicate a reduction of the **NSTD** score of more than 15000 times while the **NSGA** improvement ratio is more than 3300.

6.2 Original Contributions

As the industry pinpoints growing needs for novel and faster methods to reduce the costs allocated for completing IC functional verification, my research outlines the following original contributions:

1. In my journal article [Cri21], I review the state of the art in automating functional verification and I contribute with outlining a comparative table, i.e., Table 2.1. I describe the highlights and lowlights between the identified IV solutions that indicate promising results. Moreover, this contribution received

credit from the academic community since my journal article [Cri21] is cited at least 21 times by other research groups between January 2021 and June 2023.

2. In my conference papers [CB21, CC21], I address the *novel test generation using scenario coverage feedback* strategy by deploying ANNs as the core *learning engines*. As I highlighted in my literature review [Cri21], there were no previous attempts to enable this automation using ANNs. Thus, I took advantage of the *open-source TensorFlow* library [Goo15] and I implemented the **CAStiG** framework described in Chapter 4. Thus,

    a. With article [CB21] I contribute by introducing the novel **SL4SRRM** methodology enabled using ANNs and by proving that the *sampling function* of a *linear coverpoint with N = 1000 equally distributed intervals* can be successfully modeled using a *multilayer feedforward perceptron*. The presented **SL4SRRM**-based **CAStiG** framework successfully reaches the coverage closure goal for the target case study with a small engineering effort. It also indicates the flexibility in deploying models for any user-defined verification task. Practically, the learning process can be easily parallelized using an ANN for each *coverpoint* of the coverage model.
    b. In article [CC21] I extend my **CAStiG** framework analysis started in [CB21] and I contribute to demonstrating that nonlinear *sampling functions* for coverage bin distributions like *power-of-two* or *min & max bins* are successfully modeled using resolute ANNs. This paper alleviates the problem with novel ML-assisted process optimization techniques that enable faster functional coverage closure on nonlinear *coverpoint* models. The obtained performance results indicate great prospects for functional verification processes aided by ANN-enabled EDA tools.

3. In my journal article [Cri23], I do a design exploration on a linear and two nonlinear *coverpoint* configurations, I introduce the novel performance metrics, i.e., the **NSTD** and the **NSGA** improvement ratios, and I contribute with listing the benchmarking results that demonstrate the **SL4SRRM** proof-of-concept using ANNs.

4. In the conference article [VDC23], my co-authors and I integrate the **SL4SRRM**-based engine of the **CAStiG** framework [CB21] in the AMIQ ECTB Framework [VD23]. Thus, I contribute by exploring and listing more experimental results on how my **SL4SRRM**-enabled core performs on a real and more complex DUT model.

6.3 List of Original Publications

During this doctoral research program, I published five original papers that target the chosen research subject and my initiatives described in this thesis.

I published the two articles in ISI-indexed international conferences, namely [CB21, CC21]. In addition, also submitted article [VDC23] to a non-ISI-indexed international conference, namely *Design and Verification Conference* (DVCON) *Europe 2023*. The article has been accepted by the conference and it will be presented on during November 2023. *DVCON Europe* is a yearly conference that focuses on the industry's state of the art IC design and verification topics and is sponsored by the *Accellera System Initiative* [Acc00]. Moreover, I also published two papers in ISI-indexed international journals, which are [Cri21, Cri23].

In addition to the aforementioned publications, I also filed four research reports as part of this research program, namely [Rep1, Rep2, Rep3, Rep4]. Specifically, the report referenced in [Rep1] is equated using the manuscript before publishing article [Cri21]. Similarly, [Rep2] is equated using the work in [CB21]. [Rep3] is equated using the work in [CC21], while [Rep4] is equated using the article in [Cri23].

The full list of works that target the doctoral thesis subject filed during my research program is outlined below.

[CB21] 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

[CC21] 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

[VDC23] A. Vintilă, S. Dudă, and M.C. Cristescu, *An analysis on the impact of AI on digital IC verification coverage closure*, Design and Verification Conference (DVCon) Europe 2023, contribution accepted on the 23rd of June 2023, to be presented at the conference on the 14th of November 2023.

[Cri21] 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, Apr. 2021, pp. 99-116, ISSN:1453-8245

[Cri23] M.C. Cristescu, *Benchmarking a Smart Framework for Reducing the Coverage Closure Time in ASIC Functional Verification*, University

POLITEHNICA of Bucharest, Scientific Bulletin Series C - Electrical Engineering and Computer Science, 2023, ISSN: 2286-3540 (in review)

[Rep1] M.C. Cristescu, "*Tehnici de Machine Learning pentru îmbunătățirea metricilor de performanță în verificarea funcțională – O analiză a literaturii*", Scientific Research Report no. 1, University POLITEHNICA of Bucharest, June 2020, equated using the manuscript of article [Cri21].

[Rep2] M.C. Cristescu, *Flexible Framework for Stimuli Redundancy Reduction in Functional Verification Using Artificial Neural Networks*, Scientific Research Report no. 2, University POLITEHNICA of Bucharest, June 2021 equated using the manuscript of article [CB21].

[Rep3] M.C. Cristescu, *Stimuli Redundancy Reduction for Nonlinear Functional Verification Coverage Models Using Artificial Neural Networks*, Scientific Research Report no. 3, University POLITEHNICA of Bucharest, June 2021, equated using the manuscript of article [CC21].

[Rep4] M.C. Cristescu, *Benchmarking a Smart Framework for Reducing Coverage Closure Time in Functional Verification*, Scientific Research Report no. 4, University POLITEHNICA of Bucharest, June 2022, equated using the manuscript of article [Cri23].

6.4 Perspectives for Further Developments

6.4.1 Integrating the CAStiG Learning Core into the ECTB Framework

A first future development is to deploy the use cases described in subchapters 5.2 and 5.3 on the AMIQ's ECTB framework [VD21]. Specifically, I identified an interesting research opportunity brought by the aforementioned initiative and I decided to integrate the ANN-based *learning core* of my **CAStiG** tool into the AMIQ ECTB architecture. Therefore, the selection of new constraint definitions can now be automated using the ANN model introduced with the **CAStiG** tool.

After completing the implementation underlined in the aforementioned paragraph, together with my co-authors, we described the process in the article [VDC23] which was accepted by *DVCon Europe 2023* conference and will be officially presented on the 14th of November 2023. Still, experiments for generating even newer performance results are currently in development and we plan to provide preliminary results when the work in [VDC23] will be officially presented.

6.4.2 Other Research Directions

An interesting future development is to improve **CAStiG**'s ML model of the *Nonlinear Coverpoint with Min and Max Distribution* coverpoint so that coverage closure can be reached with a reasonable amount of engineering effort.

Further analysis pointed out new prospects to extend the **CAStiG** framework capabilities by researching more complex ANN architectures that can fit new verification use cases.

Another future research opportunity is to leverage the **SL4SRRM**-powered **CAStiG** framework for optimizing more complex functional coverage tasks available within the verification phases of industry-level projects.

One more investigation direction is to combine the strong points of both GA and ILP 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*.

Bibliography

[Don22] Z. Dong, *Research of Big Data Information Mining and Analysis: Technology Based on Hadoop Technology*, IEEE, 2022 International Conference on Big Data, Information and Computer Network (BDICN), Jan 2022, pp. 173-176, DOI: 10.1109/BDICN55575.2022.00041

[LZC+14] Y.Q. Lv, Q. Zhou, Y.C. Cai, and G. Qu, Trusted Integrated Circuits: The Problems and Challenges, Journal of Computer Science and Technology, vol. 29, no. 5, Sep. 2014, pp. 918-928, DOI:10.1007/s11390-014-1479-9

[Cri21] 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, Apr. 2021, pp. 99-116, ISSN:1453-8245

[Iee98] Standard 830-1998 - *IEEE Recommended Practice for Software Requirements Specifications*, IEEE, Oct. 1998, DOI:10.1109/IEEESTD.1998.88286

[XZB+16] B. Xue, M. Zhang, W.N. Browne, and X. Yao, *A Survey on Evolutionary Computation Approaches to Feature Selection*, IEEE Transactions on Evolutionary Computation, vol. 20, no. 4, Aug. 2016, pp. 606-626, DOI:10.1109/TEVC.2015.2504420

[CB21] 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

[Piz04] A. Piziali, Information Technology: Transmission, Processing and Storage. In: *Functional Verification Coverage Measurement and Analysis*, vol. 1, Springer New York, Jun. 2004, pp. 28, ISBN: 978-1-4020-8025-8

[KKP06] S.B. Kotsiantis, D. Kanellopoulos, and P.E. Pintelas, *Data Preprocessing for Supervised Learning*, World Academy of Science, Engineering and Technology, International Journal of Computer and Information Engineering, vol. 1, no. 1, Jan. 2006, pp. 111-117, ISSN: 1306-4428

[Hea15] J. Heaton, *Artificial Intelligence for Humans, Volume 3: Deep Learning*

and Neural Networks, Heaton Research Inc, CreateSpace Independent Publishing Platform, Dec. 2015, pp. 263, ISBN: 978-1505714340

[BF00] I. Bruha and A. Famili, *Postprocessing in Machine Learning and Data Mining*, ACM SIGKDD Explorations Newsletter, vol. 2, no. 2, Dec. 2000, pp. 110-114, DOI:10.1145/380995.381059

[Cri23] M.C. Cristescu, *Benchmarking a Smart Framework for Reducing the Coverage Closure Time in ASIC Functional Verification*, University POLITEHNICA of Bucharest, Scientific Bulletin Series C - Electrical Engineering and Computer Science, 2023, ISSN: 2286-3540 (in review)

[UCI12] *Unified Coverage Interoperability Standard (UCIS)*, Version 1.0, 2012, Accelera Systems Initiative, Available at: https://www.accellera.org/images/downloads/standards/ucis/UCIS_Version_1.0_Final_June-2012.pdf (URL link Jun. 2023)

[Sta17] C. Stan, *CoverageLens 2.0 Release*, AMIQ Consulting, Dec. 2017, Available at: https://github.com/amiq-consulting/CoverageLens/ (URL link Jun. 2023)

[Goo15] *TensorFlow*, Google Brain, 2015, Available at: https://github.com/tensorflow/tensorflow (URL link Jun. 2023)

[CC21] 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

[Bir15] N.Ș. Birman, *Functional Coverage Patterns: Bitwise Coverage*, AMIQ Consulting, Sep. 2015, Available at: https://www.amiq.com/consulting/2015/09/18/functional-coverage-patterns-bitwise-coverage/ (URL link Jun. 2023)

[Acc00] Accellera Systems Initiative, Available at: www.accellera.org (URL link Jun. 2023)

[VDC23] A. Vintilă, S. Dudă, and M.C. Cristescu, *An analysis on the impact of AI on digital IC verification coverage closure*, Design and Verification Conference (DVCon) Europe 2023, contribution accepted on the 23rd of June 2023, to be presented at the conference on the 14th of November 2023.

[VD23] 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 Jun. 2023)

[Rep1] M.C. Cristescu, *Tehnici de Machine Learning pentru îmbunătățirea metricilor de performanță în verificarea funcțională – O analiză a literaturii*, Scientific Research Report no. 1, University POLITEHNICA

of Bucharest, June 2020, equated using the manuscript of article [Cri21].

[Rep2] M.C. Cristescu, *Flexible Framework for Stimuli Redundancy Reduction in Functional Verification Using Artificial Neural Networks*, Scientific Research Report no. 2, University POLITEHNICA of Bucharest, June 2021 equated using the manuscript of article [CB21].

[Rep3] M.C. Cristescu, *Stimuli Redundancy Reduction for Nonlinear Functional Verification Coverage Models Using Artificial Neural Networks*, Scientific Research Report no. 3, University POLITEHNICA of Bucharest, June 2021, equated using the manuscript of article [CC21].

[Rep4] M.C. Cristescu, *Benchmarking a Smart Framework for Reducing Coverage Closure Time in Functional Verification*, Scientific Research Report no. 4, University POLITEHNICA of Bucharest, June 2022, equated using the manuscript of article [Cri23].