MAPLE: A Machine Learning based Aging-Aware FPGA Architecture Exploration Framework

Behnam Ghavami, Milad Ibrahimipour, Zhenman Fang, Lesley Shannon
Simon Fraser University, Burnaby, BC, Canada
Emails: {behnam_ghavami, zhenman, lesley_shannon}@sfu.ca

***Abstract*—In this paper, we develop a framework called MAPLE to enable the aging-aware FPGA architecture exploration. The core idea is to efficiently model the aging-induced delay degradation at the coarse-grained FPGA basic block level using deep neural networks (DNNs). For each type of the FPGA basic block such as LUT and DSP, we first characterize its accurate delay degradation via transistor-level SPICE simulation under a versatile set of aging factors from the FPGA fabric and in-field operation. Then we train one DNN model for each block type to quickly and accurately predict the complex relation between its delay degradation and comprehensive aging factors. Moreover, we integrate our DNN models into the widely used Verilog-to-Routing toolflow (VTR 8) to support analyzing the impact of aging-induced delay degradation on the entire large-scale FPGA architecture. Experimental results demonstrate that MAPLE can predict the delay degradation of FPGA blocks 10^4 to 10^7 times faster than transistor-level SPICE simulation, with a prediction error less than 0.7%. Our case study demonstrates that FPGA architects can effectively leverage MAPLE to explore better aging-aware FPGA architectures.**

I. Introduction

The continuous technology scaling has increased the transistor density of modern FPGAs to provide even higher performance with billions of transistors. However, it has also exacerbated the lifetime reliability issue for FPGA circuits under accelerated transistor aging in the nanoscale era, which would lead to performance degradation and even timing failure [1]. This creates a big concern for critical application areas such as aerospace, medical, and automotive industries [1].

Unfortunately, research on aging-induced delay degradation analysis for FPGAs is still in its early stage. Most existing studies have modeled the impact of aging on FPGA architectures with transistor-level SPICE simulation [2], [3]. Due to the long simulation time with traditional transistor-level approaches, it is impractical to comprehensively explore aging-aware architecture choices for modern FPGAs with billions of transistors under various aging factors.

In this paper, we present an accurate yet fast aging-aware FPGA architecture exploration framework, named as MAPLE, considering a comprehensive set of aging factors (summarized in Section II). The core idea behind MAPLE is to raise the modeling abstraction to the coarse-grained FPGA basic block level to achieve faster speed and leverage deep neural networks (DNNs) to learn an accurate delay degradation model that can achieve a similar accuracy to transistor-level simulation. We build our technique on top of the widely used FPGA CAD (Computer-Aided Design) flow VTR 8 [4] and enhance it with the fast and accurate aging-aware static timing analysis feature. Experimental results demonstrate that, compared to traditional transistor-level SPICE simulation, MAPLE can predict the aging-induced delay degradation for each FPGA soft and hard block at least 10^4x and 10^7x faster, while the prediction error rate is less than 0.7%.

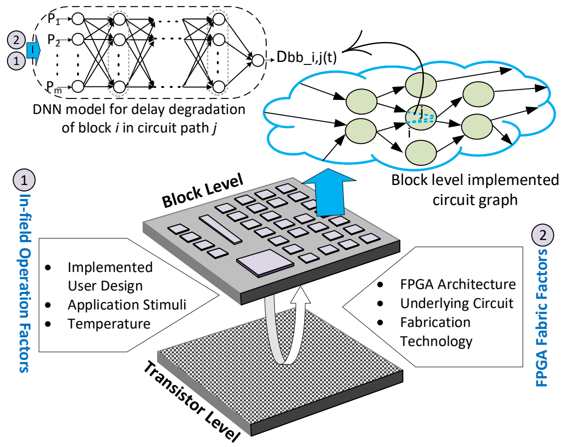

Fig. 1: Overview of FPGA fabric and in-field factors affecting FPGA aging at transistor and basic block levels. We use DNNs to model FPGA delay degradation at basic block level.

II. Our Proposal: Degradation Modeling of FPGAs at Basic Block Level Using DNN

A. Aging Factors at Transistor Level

During the operation of a transistor, due to the defects caused by the formation of *interface traps* and *oxide traps*, the mobility of the transistor degrades and its threshold voltage (V_{th}) increases. This reduces the drain current in the transistor over the projected lifetime t, which ultimately increases the transistor latency. As shown in Figure 1, delay degradation of a transistor, i.e., $D_{tr}(t)$, depends on two kinds of factors [5]. One is its environmental variations, including the temperature (T), input signal probability (λ)[1], and output load capacitance (C). The second is the technology parameters, including the power supply voltage (V_{dd}), initial threshold voltage $(V_{th}(0))$, oxide thickness (th_{ox}), transistor's length (L) and width (W). Therefore, we have:

$$\mathrm{D_{tr}(t) = f\left(t, T, \lambda, C, V_{dd}, V_{th}(0), th_{ox}, L, W\right)} \qquad (1)$$

[1] The signal probability of a wire in a circuit is defined as the probability that a randomly generated input vector will produce a one value at the wire.

where f is a model function.

To accurately model the aging effects of an FPGA design, one has to consider the delay degradation of all individual transistors deployed in the mapped design. It is worth to mention that each transistor across the circuit may be aged in a different rate based on its physical and working conditions. Considering the time-to-market of the FPGA designs, it is impractical to model the aging behavior of designs mapped onto state-of-the-art FPGAs that have billions of transistors using the very time-consuming transistor-level simulation, despite of its high accuracy.

B. Delay Degradation Model at FPGA Basic Block Level

An FPGA typically has a regular architecture, which is comprised of reconfigurable coarse-grained Basic Blocks (BBs), such as LUT, adder, MUX, flip-flop, DSP, BRAM, and switch and connection blocks. Therefore, we can leverage this regular FPGA architecture to model aging-induced delay degradation of benchmark circuits at the block level more efficiently.

To do this, we abstract a weighted circuit graph from a mapped design on an FPGA. As shown in Figure 1, each node represents an BB and each edge represents the wires connecting two BBs. Note that the switch and connection blocks are also modeled as BBs. The weights of a node represent the delay degradation of the corresponding BB in different paths, and the weights on an edge represent the wire delays. Each BB inside the FPGA design has specific transistor parameters and is also exposed to different operating conditions. Since each BB has multiple inputs and outputs, and multiple potential paths between each combination, we model the lifetime delay degradation ($\mathrm{D_{bb\_i,j}(t)}$) for each BB (i) in each specific path (j) as:

$$\mathrm{D_{bb\_i,j}\left(t\right) = f\left(t, T_i, \lambda_{\mathbf{i,j}}, \mathbf{C_{i,j}}, V_{dd\_i}, V_{th\_i}(0), th_{ox\_i}, L_i, \mathbf{W_i}\right)} \tag{2}$$

where $\lambda_{i,j}$ is a vector of signal probability and $C_{i,j}$ is a vector of load capacitance for BB i in path j; and W_i is a vector of transistor widths of BB i. Without loss of generality, $T_i, V_{dd\_i}, V_{th\_i}(0), th_{ox\_i}, L_i$ denote the working temperature, power supply voltage, initial threshold voltage, oxide thickness, and transistor length in BB i. In this paper, we do not consider the process variation, temperature variation and voltage drop across the chip. Hence, T_i, $V_{dd\_i}, V_{th\_i}(0), th_{ox\_i}, L_i$ are the same across BBs for a given fabrication technology.

C. DNN Modeling for Delay Degradation

Estimating the aging-induced delay degradation function (i.e., f in Equation 2) for an FPGA basic block under a given set of aging parameters could be considered as a regression problem. DNNs are very promising in learning models for regression problems by approximating complex and non-linear functions. Therefore, we propose to estimate the complex delay degradation function f using DNN models.

1) DNN Model: Considering the regression problem complexity associated with the delay degradation model, we use a fully-connected DNN model in this paper to provide an accurate prediction. As shown in Figure 1, the input feature vector P, which feeds to the neurons in the input layer, corresponds to the input aging parameters of a BB (Equation 2). There is only one neuron in the output layer, which corresponds to the estimated aging-induced delay degradation for the BB in the corresponding circuit path, i.e., $D_{bb\_i,j}\left(t\right)$. There are n layers of hidden layers in the DNN. In general, the relation between the input feature vector P and the output $D_{d\_i,j}\left(t\right)$ can be expressed as:

$$\begin{aligned} D_{d\_i,j}\left(t\right) &= \mathbf{w_n} \cdot g^{(n-1)} + \mathbf{b_n};\ g^{(0)} = \mathbf{P} \\ \forall k \in [1, n-1],\ g^{(k)} &= f^{(k)}\left(\mathbf{w_k} \cdot g^{(k-1)} + \mathbf{b_k}\right) \end{aligned} \tag{3}$$

where w_k and b_k are respectively the weight matrix and bias vector of the k^{th} hidden layer, which need to be trained for the DNN. For the output layer, b_n equals to zero. $f^{(k)}$ is a nonlinear activation function for the k^{th} hidden layer and we use the widely used ReLU (Rectified Linear Units) function.

2) Progressive DNN Training: We propose to design a different DNN model for each block type that can predict for this type of BB in different circuit paths: Each DNN model has different number of hidden layers and different number of neurons inside a hidden layer. Considering the importance of both the prediction accuracy and prediction time, our goal is to find the smallest DNN structure that could predict accurate enough delay degradation. Therefore, we propose a progressive training process to find the best DNN structure for each block type. Considering the training time, we limit the search of the number of hidden layers ranging from 1 to 20, and the number of neurons in each hidden layer as 256, 384, or 512. Note that each DNN model for a different block type could be trained in parallel.

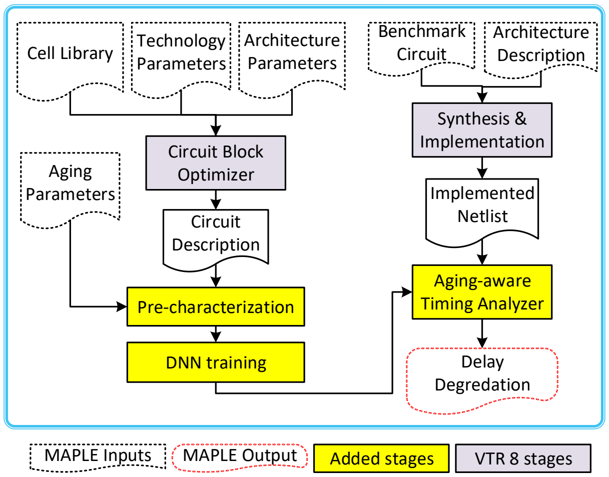

Fig. 2: Overview of our MAPLE framework.

III. MAPLE Framework

In this section, we introduce our MAPLE framework, the first CAD tool that supports aging-aware timing analysis for designs mapped onto large-scale FPGAs. An overview of MAPLE is presented in Figure 2. Next, we explain each step of MAPLE in more detail.

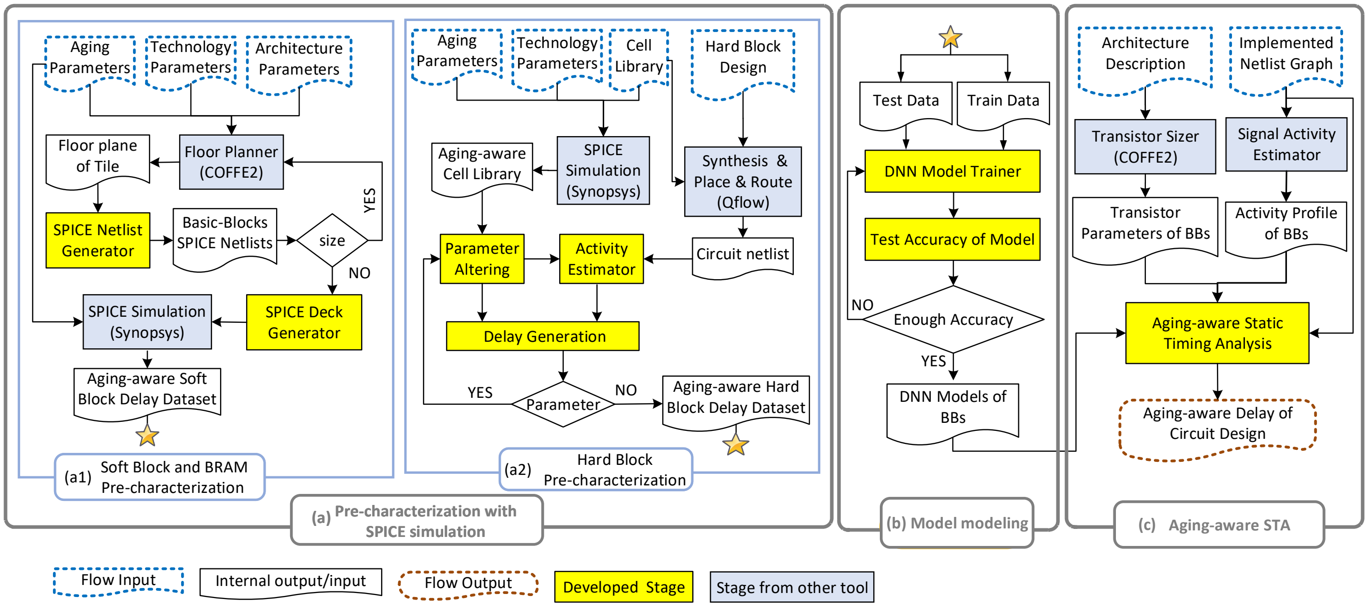

Fig. 3: Detailed procedure of analyzing the aging-induced delay degradation of a benchmark circuit in MAPLE.

A. *Pre-Characterization of Delay Degradation*

The first step is to characterize the true delay degradation values for each type of the BBs in an FPGA architecture. These characterized results will be used to train the DNN models in the next step. Figure 3 describes the detailed flow to automatically characterize the delay degradation dataset for each type of the FPGA basic block via accurate transistor-level SPICE simulation. In a modern FPGA architecture, typically there are two categories of blocks: 1) soft blocks such as the LUT, which have relatively high reconfigurability and a small number of transistors, and 2) hard blocks such as DSPs, which have relatively low reconfigurability and a large number of transistors. In the following, we explain how our toolflow characterizes the soft blocks and hard blocks.

1) Pre-Characterization of Soft Blocks (and BRAM): To explore the impact of different transistor parameters, shown in Figure 3(a1), we integrate our method with state-of-the-art automatic FPGA transistor design tool called *COFFE2* [4] that can automatically generate the transistor-level SPICE netlists of FPGA's blocks, which are required for delay measurement for each block under the input technology parameters.

In order to generate meaningful training datasets for our DNN model for a specific FPGA architecture under a set of fabrication technology parameters, its floorplan is created using the COFFE2 [4] tool. Note the FPGA floorplan varies for different parameters, and affects the wire loads and eventually the lifetime delay of each block inside the FPGA. For each type of the basic block inside the floorplan, we generate the corresponding SPICE netlists and perform transistor sizing using COFFE2 [4]. At this step, the transistor sizing operation could change the floorplan of the block. Thus, we calculate the average area of the block and use it to update the FPGA floorplan. Finally, we generate the SPICE netlist and perform SPICE simulation for each block type under a wide range of values for the aging parameters summarized in Section II to generate the corresponding delay degradation values.

2) Pre-Characterization of Large Hard Blocks: For large hard blocks such as DSP, simulating the entire block using SPICE simulation is too time consuming, as will be illustrated in Table II in Section IV-B. Therefore, we propose a different flow shown in Figure 3(a2). The main idea is to create an aging-aware cell library under different aging parameters, and then use it to create a delay dataset for the entire hard block using static timing analysis. For a given fabrication technology and cell library, we use the lookup table based method introduced for ASIC aging modeling [6] to create the aging-aware cell library. For each library cell, we perform the accurate SPICE simulation by changing a range of values for the aging parameters summarized in Section II and then create a lookup table where the input is the aging parameter values and the output is the delay degradation value. Since we use DNNs to learn the relation between aging parameters and delay degradation, this library does not need to consider every single aging condition and a lookup table approach turns to be accurate enough for hard blocks (for training purpose).

Once the aging-aware cell library is built, the hard block design in HDL (hardware description language) is synthesized, placed and routed by a synthesis tool to generate the circuit netlist that is composed of standard cells from the cell library. By altering the aging parameters summarized in Section II, an aging-aware delay dataset of the intended hard block is generated. To pass the stimuli from the hard block to each cell inside the block, we develop an activity estimator to extract the stimuli of all cells in the hard block. The delay degradation of each cell in the circuit netlist can be retrieved from the lookup tables in the aging-aware cell library. Finally, a static timing analysis tool is used to estimate the delay degradation of the entire hard block.

B. DNN Modeling for Delay Degradation

As shown in Figure 3(b), using the characterized aging-aware block delay degradation dataset, we train one DNN model for each block type to learn the relation between its input aging parameters and output delay degradation, where multiple conditions of the block in different circuit paths are considered. At the end, by using the proposed progressive DNN training method explained in Section II-C2, we get a set of trained DNN models that can estimate the delay degradation for each FPGA block under different aging conditions in different circuit paths with fast runtime and high accuracy.

C. Aging-Aware Static Timing Analysis

As shown in Figure 3(c), we estimate the lifetime delay of the entire benchmark circuit by extending the built-in static timing analysis (STA) engine of VTR 8, called Tatum [7], to perform aging-aware STA. We use COFFE2 [4] to generate the optimized circuit-level description of BBs. Then, we extract transistor sizes and output loads parameters for each BB and feed them into our DNN models. Moreover, the delay degradation estimation also requires a profile detailing the signal probability information for the entire circuit netlist. To generate this information, we pass the circuit netlist to the signal probability generator tool [8] to generate an activity profile that contains signal probability and switching activity information for every single node inside the circuit netlist graph including the inputs of all blocks. Using the circuit netlist and the activity profile, our DNN models can accurately predict the lifetime delay of all deployed BBs of the FPGA. Note that we have considered both the propagation delay and slope. We have modified the VPR (Versatile Place and Route in VTR 8) data structures so that the STA engine can exploit our DNN models for delay degradation prediction. At last, the computed lifetime delay of each graph node is passed to the STA engine to calculate the lifetime delay of the entire circuit.

IV. Experimental Results and Analysis

A. Experimental Setup

For the FPGA architecture, we use the default Altera Stratix IV like architecture supported by VTR 8 [4]. At the transistor-level, in line with COFFE2, we use the 22nm predictive technology model. The corresponding V_{dd} and oxide thickness (th_{ox}) value are 0.7V, 9.5Å. The HSPICE MOSFET Model Reliability Analysis (MOSRA) [9], [10] is used under different aging conditions as summarized in Section II. The ranges for other aging parameters are: T: [10:120]$°C$, λ: [0:1], C: [2:30] fF, $V_{th}(0)$: [-0.5:-0.1] V for PMOS transistors and [0.1:0.5] V for NMOS transistors, and t: [0:10] years. We also consider 6 different W/L ratios of each transistor: 1X, 2X, 4X, 8X, 16X, and 32X to have enough variety of transistor sizes per block by their combinations.

We build our DNN models in MAPLE based on Google's Tensorflow framework. The training of the DNN models runs on a Nvidia Titan V GPU. All the rest of our tools, including the DNN model prediction, run on a server with an Intel Core i7 (4 GHz) processor and 16GB of DRAM. The whole process of generating the DNN model for each block type, including preparing the true delays and the DNN training process, takes about 2 hours and 40 minutes. A typical FPGA architecture has less than 20 block types, which can be trained in parallel using multiple CPU and GPU servers.

TABLE I: Model accuracy and prediction runtime comparison with SPICE simulation for major FPGA soft blocks

Major Blocks	Inaccuracy in RMSPE	Runtime	
		Our Model	SPICE Simulation
LUT	0.46%	9.55 $\times 10^{-4}$s	16.05s
MUX	0.33%	9.10 $\times 10^{-4}$s	12.02s
FF	0.26%	7.38 $\times 10^{-4}$s	8.69s
Switch block	0.37%	9.14 $\times 10^{-4}$s	10.31s
Connection block	0.48%	9.61 $\times 10^{-4}$s	14.70s
BRAM	0.53%	9.16 $\times 10^{-4}$s	21.57s

TABLE II: Model accuracy and prediction runtime comparison with SPICE-based STA for major FPGA hard blocks

Major Blocks	Inaccuracy in RMSPE	Runtime		
		Our Model	LUT + STA	SPICE + STA
DSP	0.61%	9.56 $\times 10^{-4}$s	0.83s	69,225.74s
Multiplier	0.54%	9.16 $\times 10^{-4}$s	0.61s	24,462.13s

B. Accuracy and Runtime at Block Level

Model accuracy. Table I lists the accuracy validation results for some major FPGA soft blocks and BRAM; due to space constraint, we omit the full list. As shown in the second column, we calculate the Root Mean Squared Percentage Error (RMSPE) between our predicted delays (D_i) and true delays ($\hat{D}_i$). RMSPE is a robust indicator of the model accuracy [11] : the lower RMSPE is, the more accurate the model is. The very low RSMPE (less than 0.7%) validates that our DNN models are very accurate in predicting the block level delays.

Table II presents the accuracy results for major hard blocks such as the Stratix IV like DSP and multiplier. For the accuracy verification purpose, we first perform SPICE simulation for each cell inside the hard block, and then run STA to get the delay degradation for the entire hard block. We call this method as SPICE-based STA, shown as *SPICE + STA* in Table II. Compared to the SPICE-based STA approach, the error rate of our DNN model prediction is around 0.6%.

Model prediction time. As shown in Table I, on average, our DNN model prediction for soft blocks is about 10^4x faster than the transistor-level SPICE simulation. Shown in Table II, on average, our DNN model prediction for hard blocks is about 10^7x faster than the SPICE-based STA approach. The *LUT + STA* (i.e., table lookup + STA) approach is about 10^4x slower than our DNN prediction, since STA is much slower.

C. Accuracy and Runtime at Circuit Level

Model accuracy. To further verify the accuracy of our proposed model for predicting the lifetime delay of an FPGA

TABLE III: Model accuracy and prediction runtime comparison with SPICE simulation for some synthetic FPGA circuits

Circuit	#blocks	Inaccuracy in cRMSPE	Runtime	
			Our Model	SPICE Simulation
C1	24	1.01%	87.51×10^{-4}s	18,800.89s
C2	54	1.07%	140.37×10^{-4}s	54,253.81s
C3	96	1.1%	195.62×10^{-4}s	111,526.89s
C4	109	1.08%	218.59×10^{-4}s	156,856.49s

design, we calculate its *cumulative RMSPE (cRMSPE)* that includes the accumulation and propagation of all errors for estimating the delay of each individual block inside an FPGA design. The *cRMSPE* is calculated by comparing our predicted FPGA lifetime delay with the true one from SPICE simulation. Since the SPICE simulation is very time consuming, we only demonstrate the prediction accuracy for a few small synthetic FPGA designs in Table III, which mimic critical paths in large circuits. For all these synthetic FPGA circuits, their cRMSPE is around 1%, which validates that our models are accurate enough to estimate the lifetime delay of FPGA circuits.

Model prediction time. As shown in Table III, at the FPGA circuit level, our DNN model prediction is about $10^6 - 10^7$x faster than the transistor-level SPICE simulation, since the SPICE simulation time grows significantly with more blocks. We have also used MAPLE to perform aging-aware STA for large circuits, which takes around 6.1 seconds for a benchmark with around 100,000 LUTs.

D. Case Study: Aging-Aware FPGA Architecture Exploration

We present one case study with MAPLE to illustrate the impact of aging on FPGA architecture design choices. We explore 12 different parameters of an Altera Stratix IV like architecture in which the LUT input size (K) and cluster size (N) change. We choose the MCNC benchmark circuits as they are useful in architectural exploration [12]. We run benchmark circuits on the these various architectures to get the initial delay without aging (blue dots in Figure 4) and the aging-induced lifetime delay (red dots in Figure 4) after 10 years under a temperature of $50°C$.

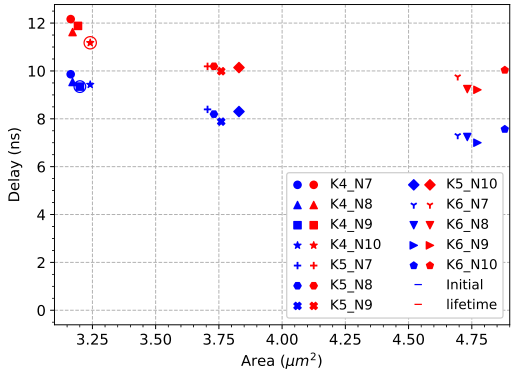

Fig. 4: Impact of aging on FPGA architecture exploration. Blue and red dots denote initial delay and lifetime delay.

Figure 4 presents the geometric mean of delays and areas across all benchmark circuits for each architecture choice. Typically, the architecture that has the lowest area-delay product is chosen as the best FPGA architecture. As shown in Figure 4, if the aging impact is not considered, the architecture K4_N9 will be chosen since it has the lowest area-delay product among all blue dots (circled), which is in line with prior studies [12]. However, if the aging impact is considered, the architecture K4_N10 will be chosen as it has the lowest area-delay product among all red dots (circled).

V. Conclusion

In this paper, we have presented MAPLE, the first CAD framework that enables fast and accurate aging-aware FPGA architecture exploration. MAPLE trains deep neural networks (DNNs) to learn the FPGA delay degradation at the basic block level under a comprehensive set of aging factors from both the FPGA fabric and in-field operation. As a result, it achieves 10^4 to 10^7 times faster speed over traditional transistor-level SPICE simulation, while achieving almost the same accuracy. Moreover, we have enhanced VTR 8 with aging-aware static timing analysis to enable the aging analysis for the entire circuit that runs on top of the FPGA architecture.

Acknowledgements

We acknowledge the support from Government of Canada Technology Demonstration Program and MDA Systems Ltd; NSERC Discovery Grant RGPIN-2019-04613 and DGECR-2019-00120; Canada Foundation for Innovation John R. Evans Leaders Fund; Simon Fraser University New Faculty Start-up Grant; Xilinx, Huawei and Nvidia.

References

[1] S. Srinivasan, P. Mangalagiri, Y. Xie, N. Viiaykrishnan, and K. Sarpatwari, "Flaw: Fpga lifetime awareness," in *DAC 2016*, pp. 630–635.

[2] E. Stott, P. Sedcole, and P. Y. Cheung, "Modelling degradation in fpga lookup tables," in *FPT 2009*, pp. 443–446.

[3] M. Naouss and F. Marc, "Modelling delay degradation due to nbti in fpga look-up tables," in *FPL 2016*, pp. 1–4.

[4] K. E. Murray, O. Petelin, S. Zhong, J. M. Wang, M. Eldafrawy, J.-P. Legault, E. Sha, A. G. Graham, J. Wu, M. J. P. Walker, H. Zeng, P. Patros, J. Luu, K. B. Kent, and V. Betz, "Vtr 8: High-performance cad and customizable fpga architecture modelling," *TRETS 2020*, vol. 13, no. 2, pp. 1–55, May 2020.

[5] K. Kang, S. Gangwal, S. P. Park, and R. Kaushik, "Nbti induced performance degradation in logic and memory circuits: How effectively can we approach a reliability solution?" in *ASP-DAC 2008*, pp. 726–731.

[6] H. Amrouch, B. Khaleghi, A. Gerstlauer, and J. Henkel, "Reliability-aware design to suppress aging," in *DAC 2016*, pp. 1–6.

[7] K. E. Murray and V. Betz, "Tatum: Parallel timing analysis for faster design cycles and improved optimization," in *FPT 2018*, pp. 110–117.

[8] J. Lamoureux and S. J. Wilton, "Activity estimation for field-programmable gate arrays," in *FPL 2006*, pp. 1–8.

[9] B. Tudor, J. Wang, C. Sun, Z. Chen, Z. Liao, R. Tan, W. Liu, and F. Lee, "Mosra: An efficient and versatile mos aging modeling and reliability analysis solution for 45nm and below," in *ICSICT 2010*, pp. 1645–1647.

[10] B. Tudor, J. Wang, W. Liu, and H. Elhak, "Mos device aging analysis with hspice and customsim," *Synopsys, White Paper*, 2011. [Online]. Available: https://www.synopsys.com/content/dam/synopsys/verification/white-papers/mosra-wp.pdf

[11] K. Huang, X. Zhang, and N. Karimi, "Real-time prediction for ic aging based on machine learning," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 12, pp. 4756–4764, 2019.

[12] G. Zgheib and P. Ienne, "Evaluating fpga clusters under wide ranges of design parameters," in *FPL 2017*, pp. 1–8.