Fast and Accurate Aging-Aware Cell Timing Model via Graph Learning

Yuyang Ye, Tinghuan Chen, Zicheng Wang, Hao Yan, Bei Yu, *Senior Member, IEEE*, and Longxing Shi

***Abstract*—With transistors scaling down, aging effects become increasingly significant in circuit design. Thus, the aging-aware cell timing model is necessary for evaluating the aging-induced delay degradation and their impact on circuit performance. However, the tradeoff between accuracy and efficiency becomes a bottleneck in traditional methods. In this brief, we propose a fast and accurate aging-aware cell timing model via graph learning. The information of multi-typed devices on different arcs can be embedded by heterogeneous graph attention networks (H-GAT) and the embedded results help improve the accuracy of our aging-aware timing model. The experimental results indicate the proposed timing model can achieve high accuracy efficiently.**

***Index Terms*—Timing analysis, graph learning, design for reliability.**

I. Introduction

With the continuous shrinking of CMOS technology nodes, transistor aging effects such as negative bias temperature instability (NBTI) become more and more significant, making timing closure and signoff increasingly challenging [1]. To guarantee the parametric yield and circuit lifetime, designers resort to adding performance-degrading timing guardbands (GBs) on the operational clock period [2], [3], [4]. The accuracy of the aging-aware cell timing model is of great importance for evaluating the amount of GBs efficiently.

Existing methods for aging-aware cell timing modeling can be divided into two categories: (1) Look-up-table-based (LUT-based) modeling methods [5], [6], [7]. In [6], a LUT-based model is proposed to estimate NBTI-induced delay degradation while the PBTI aging effects are totally ignored. Kiamehr et al. [7] introduce an aging-aware timing model considering the impacts of input signal probabilities. In [5], a more accurate LUT-based method is introduced. (2) Machine learning-based modelingmethods [4], [8], [9]. In [8], the support vector machine (SVM) model is used to capture the relationship between signal probabilities and delay degradation of cells. However, the impractical assumptions about constant supply voltages and temperatures cause an obvious loss of accuracy. Klemme and Amrouch [4] present a machine learning (ML) approach that accurately predicts aging-induced delay degradation for a continuous range of supply voltages and threshold voltages. In [9], feed-forward neural networks (FFNNs) are used.

For LUT-based models, the accuracy is limited to the operating conditions under which the LUT is built. For ML-based models, the bottleneck of accuracy is caused by two main challenges: (1) They ignore the information of circuit structures on different timing arcs in cells. Due to significant parasitic effects, the timing arc structures of standard cells become more and more complex. In addition, there is an increasing necessity for modeling large cells with complicated structures, e.g., full adder (FA). Thus, the traditional ML methods always induce delay errors due to losing some essential structure information on timing arcs. (2) They ignore the impact of capacitors and resistors in cells. The cell circuits have heterogeneity since being composed of multi-typed devices, including transistors, capacitors and resistors. Thus, the impact of capacitors and resistors on transistor aging should be considered while modeling aging-induced timing degradation, especially under advanced technology

To overcome the two challenges, graph learning methods can be used to collect information of all transistors, capacitors and resistors on different timing arcs in the whole cells. For challenge (1), the cell structures can be learned through aggregating cell circuit information among different timing arcs; For challenge (2), the impacts of capacitors and resistors on timing degradation caused by transistor aging can be mined by combing the information from multi-typed devices. Finally, the cells can be embedded accurately based on collected information and the embedded results helps to improve the aging-aware timing model accuracy.

In this brief, we propose a fast and accurate aging-aware cell timing model via graph learning. It can estimate aging-induced delay degradation of cells based on information from transistors, capacitors and resistors on different timing arcs of cells under different operational conditions. For collecting information and embedding cells efficiently, we present heterogeneous graph attention networks (H-GAT). According to experimental results, the aging-aware cell timing model based on H-GAT achieves higher accuracy and efficiency benefitting from more cell information.

Manuscript received 26 June 2023; accepted 19 July 2023. Date of publication 26 July 2023; date of current version 8 January 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 61874152 and Grant 61904030, and in part by the Research Grants Council of Hong Kong, SAR, under Project CUHK24209017. This brief was recommended by Associate Editor L. Qian. *(Corresponding authors: Hao Yan; Tinghuan Chen.)*

Yuyang Ye, Zicheng Wang, Hao Yan, and Longxing Shi are with the National ASIC Research Center, Southeast University, Nanjing 210096, China (e-mail: yeyuyang@seu.edu.cn; yanhao@seu.edu.cn).

Tinghuan Chen is with the School of Science and Engineering, The Chinese University of Hong Kong (Shenzhen), Shenzhen 518172, China (e-mail: chentinghuan@cuhk.edu.cn).

Bei Yu is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, SAR.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSII.2023.3298917.

Digital Object Identifier 10.1109/TCSII.2023.3298917

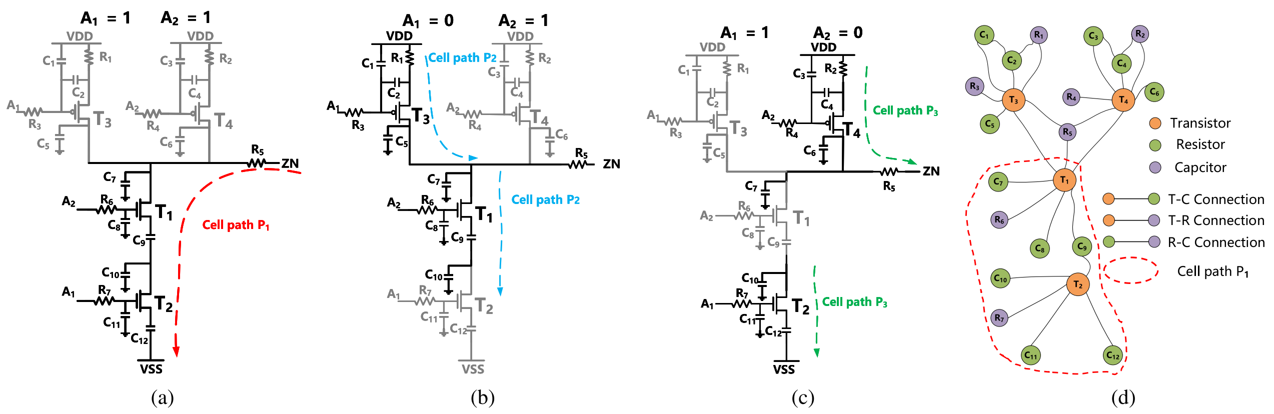

Fig. 1. The examples of cell paths and cell path-based neighbors in NAND2 are shown in (a), (b) and (c). (d) is the heterogeneous graph representation for a NAND2 circuit with multi-typed nodes (devices), edges (connections) and an example of cell path $\mathcal{P}_1$ (a timing arc).

- We develop a fast and accurate aging-aware cell timing model based on H-GAT, which can achieve a trade-off between model accuracy and efficiency.
- We propose heterogeneous graph attention networks (H-GAT) to collect information from multi-typed devices on different timing arcs and to achieve cell embedding.
- The experimental results indicate the proposed timing model outperforms conventional ML-based models and the LUT-based model in terms of accuracy.

II. Preliminaries

A. Aging-Aware Cell Library Characterization Using SPICE

In this brief, an aging-aware timing model method will be achieved to estimate aging-induced cell delay degradation for different kinds of cells in the cell library. For each kind of cell under different cell operational conditions, the commercial SPICE simulation tool (HSPICE in this brief [10]) with the MOSRA aging model [11] is adopted to perform the aging simulation to calculate the ground truths of cell delay. MOSRA in HSPICE offers an accurate solution for analyzing the aging-aware timing degradation. Because it accurately models circuit performance with the MOS device aging and circuit operation. Finally, we can get accurate delay results.

For accurate aging-aware timing analysis for timing paths, PrimeTime [12] helps generate SPICE deck templates from the design database for particular paths of interest. The SPICE deck templates are transistor-level netlists of timing paths with the parasitic information. Based on the SPICE deck templates and different testbenches, HSPICE [10] with the MOSRA aging model [11] is adopted to perform the aging simulation on each extracted timing path. Then it can generate the aging-aware timing baseline results of timing paths.

B. Problem Formulation

We focus on getting fast and accurate aging-aware timing models for standard cells. To clearly define our problem, we first provide some definitions as follows.

Definition 1 (Cell Path): The timing arcs controlled by different input vectors, which contain charging paths when PMOS is on and discharging paths when NMOS is on.

Definition 2 (Cell Path-Based Neighbors): The same type of devices located on the same cell path.

Examples: As shown in Fig. 1, the timing arcs caused by three different input vectors in NAND2 (including $[A_1 = 0, A_2 = 1]$, $[A_1 = 1, A_2 = 0]$ and $[A_1 = 1, A_2 = 1]$) are regarded as cell paths (including $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$). The input vector $[A_1 = 0, A_2 = 0]$ is considered when analyzing multi-input-switching effect, which will be discussed in our future work. In Fig. 1(a), the NMOS transistor T_1 and T_2 are on when $[A_1 = 1, A_2 = 1]$. There is a discharging path from the output pin ZN to the power rail VSS, which is regarded as cell path $\mathcal{P}_1$; In Fig. 1(b), the NMOS transistor T_1 and PMOS transistor T_3 are on when $[A_1 = 0, A_2 = 1]$. There is a charging path from the power rail VDD to the output pin ZN, the capacitor C_7 and the capacitor C_{10}, which is regarded as cell path $\mathcal{P}_2$; In Fig. 1(c), the NMOS transistor T_2 and PMOS transistor T_4 are on when $[A_1 = 1, A_2 = 0]$. There is a charging path from the power rail VDD to the output pin ZN and a discharging path from the capacitor C_{10} to the power rail VSS, which are combined and then regarded as cell path $\mathcal{P}_3$. Note that the behavior of discharging path from the capacitor C_{10} influence the body voltage and threshold voltage of transistor T_4, which cause different aging-aware cell delay degradation. For the transistor node T_1 on cell path $\mathcal{P}_1$, the transistor node T_2 is regarded as a cell path-based neighbor of T_1. For the capacitor node C_1 on cell path $\mathcal{P}_2$, the capacitor node C_2 is regarded as a cell path-based neighbor of C_1. There is a close relationship between timing arcs (cell paths) and cell delays [1]. Obviously, multi-typed devices on different cell paths push totally different impacts on aging-induced cell delay degradations. Thus, the information on multi-typed devices and cell path structures should be considered jointly while estimating aging-aware cell delay.

Our problem formulation is formally defined as follows.

Problem 1 (Aging-Aware Timing Cell Modeling): Given a cell circuit with multi-typed devices (transistors, capacitors and resistors), cell structure, cell paths and operational conditions, estimate the aging-aware cell delay.

C. Cell Circuit Vs. Heterogeneous Graph

As shown in Fig. 1(d), we model a cell circuit as a heterogeneous graph $\mathbb{G}_h = (\mathcal{V}_h, \mathcal{E}_h, \mathcal{P})$. In the heterogeneous

TABLE I
ORIGINAL FEATURES OF MULTI-TYPED DEVICES AND CELL OPERATIONAL CONDITIONS USED IN OUR H-GAT

Type	Feature	Range or Options
Transistor	input slew	[5 : 1250] ps
	threshold voltage	[-0.5 : -0.1] V for PMOS; [0.1 : 0.5] V for NMOS
	W/L ration	$[1\times, 2\times, 4\times, 8\times, 16\times, 32\times]$
	input signal probability	[0 : 1]
Capacitor	value of capacitance	[0.001 : 40] fF
Resistor	value of resistance	[0.00001 : 10] kΩ
Ope. Con.	output load capacitance	[0.25 : 25] fF
	temperature	[20 : 120] °C
	supply voltage	[0.7 : 1.1] V
	lifetime	[0 : 9] years

graph, multi-typed nodes $\mathcal{V}_h = \mathcal{T} \cup \mathcal{C} \cup \mathcal{R}$, including red, green and blue nodes represent multi-typed devices, including transistors, capacitors and resistors. Multi-typed edges $\mathcal{E}_h = \mathcal{E}_{\mathcal{T},\mathcal{C}} \cup \mathcal{E}_{\mathcal{T},\mathcal{R}} \cup \mathcal{E}_{\mathcal{C},\mathcal{R}}$ represent multi-typed connections between multi-typed devices. Cell paths $\mathcal{P}$ represent timing arcs consisting of all nodes and edges visited, which can be regarded as a sub-graph.

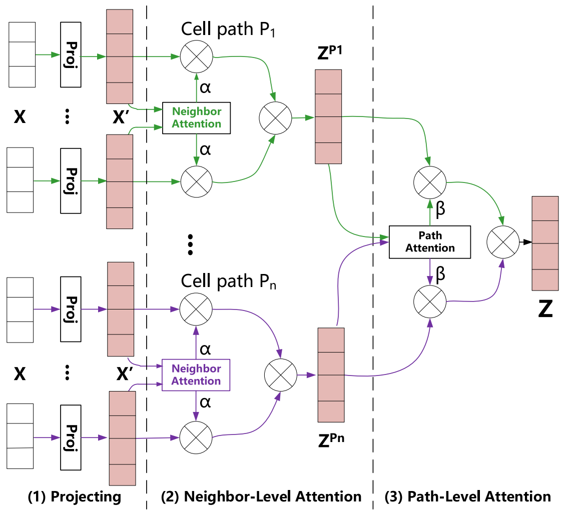

Fig. 2. The flow of generating node representations for multi-typed devices based on aggregating information from cell path-based neighbors and cell paths.

III. PROPOSED METHOD

In this brief, we develop a fast and accurate aging-aware cell timing model via graph learning. The overall flow contains: Step 1 performs cell modeling with a heterogeneous graph consisting of multi-typed nodes, multi-typed edges and cell paths; Step 2 embeds cells to aggregate the information of multi-typed devices on different cell paths through multi-typed connections based on trained H-GAT. Then it generates cell embedding results, which helps improve the accuracy of timing models efficiently; Step 3 estimates aging-aware cell delays based on embedding results.

A. Modeling Cells

The heterogeneous graph used to model cell $\mathbb{G}_h = (\mathcal{V}_h, \mathcal{E}_h, \mathcal{P})$ is represented by multi-typed node feature matrices $\boldsymbol{X}$: $\{\boldsymbol{X}_{\mathcal{T}}, \boldsymbol{X}_{\mathcal{C}}, \boldsymbol{X}_{\mathcal{R}}\}$, multi-typed adjacency matrices $\boldsymbol{A}$: $\{\boldsymbol{A}_{\mathcal{T},\mathcal{C}}, \boldsymbol{A}_{\mathcal{T},\mathcal{R}}, \boldsymbol{A}_{\mathcal{R},\mathcal{C}}\}$ and a cell operational condition matrix $\boldsymbol{H}$. The multi-typed node feature matrices $\boldsymbol{X}_{\mathcal{T}}$: $\{\boldsymbol{x}_T, \forall T \in \mathcal{T}\}$, $\boldsymbol{X}_{\mathcal{C}}$: $\{\boldsymbol{x}_C, \forall C \in \mathcal{C}\}$ and $\boldsymbol{X}_{\mathcal{R}}$: $\{\boldsymbol{x}_R, \forall R \in \mathcal{R}\}$ collect information from multi-typed devices, including transistors, capacitors and resistors. The multi-typed adjacency matrices $\boldsymbol{A}$ indicate multi-typed connections in the given cell. The cell operational condition matrix $\boldsymbol{H}$ collects different operational conditions at the cell level. For all cells, the aging effects under various operating conditions cause different delay degradations due to trap generation [1]. Thus, we select some circuit parameters based on domain knowledge and parameter-sweeping experiments and use them in the multi-typed node feature matrices. They are shown in Table I.

B. Embedding Cell Using H-GAT

There are four important modules in H-GAT, including the projecting module, neighbor-level attention module, path-level attention module and pooling module. In the projecting module, the multi-typed device feature matrices $\{\boldsymbol{x}_T, \boldsymbol{x}_C, \boldsymbol{x}_R\}$ are translated to $\{\boldsymbol{x}'_T, \boldsymbol{x}'_C, \boldsymbol{x}'_R\}$ with the same space. In the neighbor-level attention module, the target nodes aggregate the features from their cell-path-based neighbors on different cell paths. It generates cell-path specific aggregated node representations $\{z_T^{\mathcal{P}}, z_C^{\mathcal{P}}, z_R^{\mathcal{P}}\}$. In the path-level attention module, the cell-path specific aggregated node representations are fused jointly to generate final node representations $\{z_T, z_C, z_R\}$. The flow of generating node representations is shown in Fig. 2. In the pooling module, the final node representations and cell operational condition matrix $\boldsymbol{H}$ are concatenated to generate the cell embeddings $\boldsymbol{E}$.

Projecting module: The multi-typed node matrices $\{\boldsymbol{x}_T, \boldsymbol{x}_C, \boldsymbol{x}_R\}$ have different spaces. Thus, we design the type-specific transformation matrices to project multi-typed node feature matrices into the same space:

$$\boldsymbol{x}'_T = \boldsymbol{P}_{\mathcal{T}} \cdot \boldsymbol{x}_T, \boldsymbol{x}'_C = \boldsymbol{P}_{\mathcal{C}} \cdot \boldsymbol{x}_C, \boldsymbol{x}'_R = \boldsymbol{P}_{\mathcal{R}} \cdot \boldsymbol{x}_R, \qquad (1)$$

where $\boldsymbol{x}_T$, $\boldsymbol{x}_C$ and $\boldsymbol{x}_R$ are original feature matrices. Then, $\boldsymbol{x}'_T$, $\boldsymbol{x}'_C$ and $\boldsymbol{x}'_R$ are projected feature matrices. $\boldsymbol{P}_{\mathcal{T}}$, $\boldsymbol{P}_{\mathcal{C}}$ and $\boldsymbol{P}_{\mathcal{C}}$ are the transistor-specific, capacitor-specific and resistor-specific transformation matrix, respectively. Finally, $\boldsymbol{x}'_T$, $\boldsymbol{x}'_C$ and $\boldsymbol{x}'_R$ have a same space.

Neighbor-level attention module: This module takes the transformed multi-typed feature matrices $\boldsymbol{x}'_T$, $\boldsymbol{x}'_C$ and $\boldsymbol{x}'_R$ as inputs, and produces cell-path specific node representations $\{z_T^{\mathcal{P}}, z_C^{\mathcal{P}}, z_R^{\mathcal{P}}\}$. The information of cell path-based neighbors can be aggregated based on neighbor-level attention coefficients. Equation (2) is an example of how to calculate these coefficients. The neighbor attention coefficient $\alpha_{T_1,T_2}^{\mathcal{P}_1}$ means how important transistor node T_2 will be for T_1 on cell path $\mathcal{P}_1$. And all neighbor-level attention coefficients of node T_1 should be normalized with softmax function.

$$\alpha_{T_1,T_2}^{\mathcal{P}_1} = \frac{\exp\Big(\text{LeakyReLU}\Big((\boldsymbol{a}^{\mathcal{P}_1})^\top \Big[\boldsymbol{x}'_{T_1} \| \boldsymbol{x}'_{T_2}\Big]\Big)\Big)}{\sum_{T_k \in \mathcal{N}_{T_1}^{\mathcal{P}_1}} \exp\Big(\text{LeakyReLU}\Big((\boldsymbol{a}^{\mathcal{P}_1})^\top \Big[\boldsymbol{x}'_{T_1} \| \boldsymbol{x}'_{T_k}\Big]\Big)\Big)}, \qquad (2)$$

where T_1 is the target node and T_2 is its cell path-based neighbor belonging to neighborhood set $N_{T_1}^{\mathcal{P}_1}$. $\cdot^\top$ represents transposition and $\|$ is the concatenation operation. The neighbor

TABLE II
THE MODEL ACCURACY AND RUNTIME RESULTS OF THE AGING-AWARE CELL DELAY ESTIMATION USING DIFFERENT APPROACHES, INCLUDING RE (%), MAE (PS) AND RUN.(MS). "RUN." REPRESENTS RUNTIME OF PREDICTING CELL DELAY USING DIFFERENT APPROACHES. "OURS W/O $\mathcal{R}$" AND "OURS W/O $\mathcal{C}$" REPRESENT OUR WORK WITHOUT CONSIDERING RESISTOR ELEMENTS AND CAPACITOR ELEMENTS

Gate	SPICE [10]			LUT [5]			SVM [8]			Aadam [9]			GAT [13]			Ours w/o $\mathcal{C}$			Ours w/o $\mathcal{R}$			Ours		
	RE	MAE	Run.	RE	MAE	Run.	RE	MAE	Run.	RE	MAE	Run.	RE	MAE	Run.	RE	MAE	Run.	RE	MAE	Run.	RE	MAE	Run.
INV	0.00	0.00	2.84k	4.08	3.54	0.139	1.48	3.12	0.117	0.45	2.99	0.324	0.32	2.18	0.332	0.23	1.54	0.347	0.15	1.21	0.349	**0.03**	**0.12**	0.351
NAND	0.00	0.00	3.56k	4.38	5.02	0.156	1.54	3.68	0.135	0.42	2.04	0.329	0.34	1.79	0.386	0.25	1.42	0.426	0.19	1.09	0.431	**0.02**	**0.19**	0.458
NOR	0.00	0.00	3.76k	4.25	4.68	0.158	1.69	3.54	0.141	0.51	2.31	0.335	0.43	2.07	0.391	0.35	1.79	0.406	0.24	1.37	0.414	**0.03**	**0.23**	0.442
XNOR	0.00	0.00	9.84k	3.89	5.23	0.154	1.31	4.05	0.139	0.42	3.61	0.338	0.35	3.22	0.465	0.24	2.79	0.492	0.21	2.14	0.502	**0.03**	**0.25**	0.547
AOI	0.00	0.00	9.82k	5.62	6.67	0.156	2.95	6.53	0.124	0.79	5.92	0.352	0.65	4.67	0.458	0.43	2.94	0.491	0.34	2.31	0.503	**0.04**	**0.34**	0.558
BUFF	0.00	0.00	2.66k	4.12	3.04	0.142	1.39	2.57	0.126	0.47	2.43	0.126	0.29	2.07	0.319	0.21	1.23	0.322	0.15	1.02	0.324	**0.02**	**0.09**	0.327
AND	0.00	0.00	3.07k	4.55	4.36	0.147	1.12	2.41	0.131	0.39	2.79	0.317	0.30	1.84	0.371	0.23	1.52	0.408	0.17	1.12	0.412	**0.03**	**0.14**	0.435
OR	0.00	0.00	3.18k	4.26	4.82	0.143	1.52	3.17	0.129	0.47	2.09	0.322	0.35	1.89	0.382	0.29	1.61	0.412	0.18	1.24	0.423	**0.02**	**0.11**	0.439
XOR	0.00	0.00	8.67k	5.34	4.98	0.155	1.93	4.05	0.151	0.25	4.16	0.346	0.18	3.79	0.492	0.14	2.59	0.531	0.11	2.09	0.534	**0.04**	**0.21**	0.552
OAI	0.00	0.00	9.71k	5.08	7.63	0.141	2.41	6.08	0.132	0.81	6.34	0.327	0.74	5.69	0.498	0.27	3.08	0.522	0.22	2.57	0.531	**0.04**	**0.31**	0.541
FA	0.00	0.00	15.24k	9.06	13.65	0.137	5.78	12.41	0.129	4.92	10.05	0.331	3.14	7.23	0.581	1.34	4.21	0.621	0.84	3.41	0.637	**0.05**	**0.48**	0.653
Ave.	0.00	0.00	6.58k	4.97	5.78	0.148	2.10	4.69	0.132	0.90	4.07	0.331	0.65	3.32	0.427	0.36	2.25	0.453	0.26	1.78	0.462	**0.03**	**0.22**	0.482

attention coefficient is parametrized by a weight vector $\boldsymbol{a}^{\mathcal{P}_1}$ with the LeakyReLU nonlinear function.

Based on the normalized coefficients for each neighbor in $\mathcal{N}_{T_1}^{\mathcal{P}_1}$, we can perform a weighted sum to get the cell-path specific node representation ($\boldsymbol{z}_{T_1}^{\mathcal{P}_1}$) of T_1 on cell path $\mathcal{P}_1$:

$$\boldsymbol{z}_{T_1}^{\mathcal{P}_1} = \sum_{T_k \in \mathcal{N}_{T_1}^{\mathcal{P}_1}} \alpha_{T_1,T_k}^{\mathcal{P}_1} \boldsymbol{x}'_{T_k}. \tag{3}$$

Given the cell-path set $\{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_n\}$ through node T_1, we can obtain cell path-specific node representations $\{\boldsymbol{z}_{T_1}^{\mathcal{P}_1}, \boldsymbol{z}_{T_1}^{\mathcal{P}_2}, \ldots, \boldsymbol{z}_{T_1}^{\mathcal{P}_n}\}$.

Path-level attention module: Every node in a heterogeneous graph contains multiple cell path-specific node representations on different cell paths. To learn a more comprehensive node representation, they will be fused. The information of different cell path-specific node representations can be aggregated based on path-level attention coefficients. These coefficients can be calculated as follows:

$$\beta_{T_1}^{\mathcal{P}_1} = \frac{\exp\Big(\Big((\boldsymbol{q}^{\mathcal{P}_1})^\top \tanh\Big[\boldsymbol{W}^{\mathcal{P}_1}\boldsymbol{z}_{T_1}^{\mathcal{P}_1} + \boldsymbol{b}^{\mathcal{P}_1}\Big]\Big)\Big)}{\sum_{i=1}^{n} \exp\Big(\Big((\boldsymbol{q}^{\mathcal{P}_i})^\top \tanh\Big[\boldsymbol{W}^{\mathcal{P}_i}\boldsymbol{z}_{T_1}^{\mathcal{P}_i} + \boldsymbol{b}^{\mathcal{P}_i}\Big]\Big)\Big)}, \tag{4}$$

where $\boldsymbol{W}^{\mathcal{P}_i}$ is the weight matrix, $\boldsymbol{b}^{\mathcal{P}_i}$ is the bias vector and $\boldsymbol{q}^{\mathcal{P}_i}$ is the path-level attention vector. Based on the normalized path-level attention coefficients, we can fuse cell-path specific node representations $\boldsymbol{z}_{T_1}^{\mathcal{P}_i}$ to obtain the final embedding $\boldsymbol{z}_{T_1}$:

$$\boldsymbol{z}_{T_1} = \sum_{i=1}^{n} \beta_{t_1}^{\mathcal{P}_i} \boldsymbol{z}_{T_1}^{\mathcal{P}_i}. \tag{5}$$

Following the progress, we can get the final node representations for each device in the cell, including transistors ($\boldsymbol{Z}_{\mathcal{T}}$: $\{\boldsymbol{z}_T, \forall T \in \mathcal{T}\}$), capacitors ($\boldsymbol{Z}_{\mathcal{C}}$: $\{\boldsymbol{z}_C, \forall C \in \mathcal{C}\}$) and resistors ($\boldsymbol{Z}_{\mathcal{R}}$: $\{\boldsymbol{z}_R, \forall R \in \mathcal{R}\}$). Compared with the original features, the new representations contain more information from cell path-based neighbors and different cell paths.

Pooling module: In the pooling module, all the node representations are combined with the operational condition feature $\boldsymbol{H}$ to generate the cell embedding $\boldsymbol{E}$. As shown in Equation (6), the node representations of transistors, capacitors and resistors in the cell are summed up, averaged and concatenated with the operational condition feature:

$$\boldsymbol{E} = \left(\frac{1}{N_T}\sum_{T\in\mathcal{T}} \boldsymbol{z}_T\right) \Big\| \left(\frac{1}{N_C}\sum_{C\in\mathcal{C}} \boldsymbol{z}_C\right) \Big\| \left(\frac{1}{N_R}\sum_{R\in\mathcal{R}} \boldsymbol{z}_R\right) \Big\| \boldsymbol{H}, \tag{6}$$

TABLE III
THE BENCHMARK INFORMATION

Circuit	#Cells	#Nets	#FFs	#Paths	#Input	#Output
PCI	32015	52347	3519	25961	160	201
VGA	163741	198674	20168	67259	85	99
DIST	129812	174395	5839	48236	2562	12
DES	143542	185932	9216	53198	234	140
MULT	176295	206537	3591	65929	3202	1600
NET	986347	1035482	97831	213687	1836	10

TABLE IV
THE COMPARISONS OF ACCURACY FOR THE AGING-AWARE PATH DELAY CALCULATION, INCLUDING RE (%) AND MAE (PS)

Circuit	LUT [5]		SVM [8]		Aadam [9]		GNN4REL [15]		Ours	
	RE	MAE	RE	MAE	RE	MAE	RE	MAE	RE	MAE
PCI	8.92	75.28	5.92	68.35	3.69	63.27	2.58	10.46	**0.17**	**4.31**
VGA	9.21	81.69	6.26	64.66	4.92	57.52	2.69	14.33	**0.43**	**5.22**
DIST	6.34	68.31	4.04	56.02	2.91	48.37	2.49	9.25	**0.21**	**4.28**
DES	8.25	88.17	5.52	71.56	3.52	65.29	2.55	11.48	**0.34**	**5.12**
MULT	10.57	92.64	7.62	81.37	5.71	72.51	3.06	17.54	**0.59**	**6.09**
NET	14.32	125.45	9.23	92.54	8.69	83.23	5.89	27.27	**0.62**	**7.52**
Ave.	9.60	88.59	6.43	72.42	4.91	65.03	3.21	15.06	**0.39**	**5.42**

where N_T, N_C and N_R are the numbers of transistors, capacitors and resistors in the cell, respectively.

C. Predicting Aging-Aware Cell Delay

Based on the cell embedding result $\boldsymbol{E}$, we use a multilayer perceptron layer (*MLP*) to predict the aging-aware delay. A trainable parameter $\boldsymbol{\theta}$ in the *MLP* is introduced.

$$D = MLP(\boldsymbol{\theta} \mid \boldsymbol{E}), \tag{7}$$

where D is the aging-aware delay estimation result. *MLP* is a multilayer perception.

IV. EXPERIMENTAL RESULTS

For evaluating the model accuracy while estimating aging-aware cell delay, the types of cells we consider include 65 different cells from the TSMC 16nm standard cell library. For each type of cell, we achieve one specific aging-aware delay model using our method. The ranges and options of used features are shown in Table I, and we generate 100000 samples for each kind of cell based on the ranges and options. Table II shows all aging-aware cell delay estimation accuracies. And for aging-aware path delay calculation, the timing paths are exacted from ISPD benchmarks [14] and details of designs in benchmarks used in our work are listed in Table III. Table IV illustrates the path delay calculation delay accuracy of aging-aware STA based on different models and GNN4REL [15].

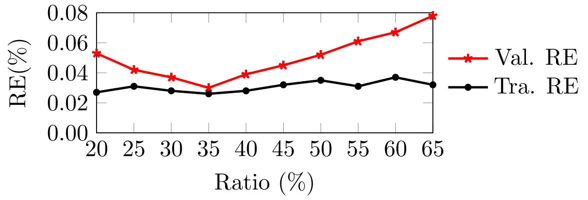

Fig. 3. Average training RE (Tri. RE) and average validation RE (Val. RE) under different ratios of training samples ranging from 20% to 65%.

The ground truths of aging-aware cell delays and path delays are generated via HSPICE [10] (described in Section II-A). We use PyTorch to implement our models. For predicting path delay, we achieve aging-aware STA through implementing our model into Opentimer [16]. Our models are trained on a Linux machine with 32 cores and 4 NVIDIA Tesla V100 GPUs. The total memory used is 128GB. The parallel training progress on multiple GPUs consumes about 25 minutes on our servers for each cell on average. We use the root mean squared percentage errors (RE) and maximum absolute errors (MAE) between the estimated delay by our work (D) and the ground truths ($\hat{D}$) to evaluate the accuracy of the aging-aware cell delay and path delay. The smaller RE score means higher accuracy.

A. Generalization Ability

For ensuring the generalization ability, we achieve our model under different ratios of training samples ranging from 20% to 65%. Fig. 3 shows the average training RE and validation RE results of different cells from TSMC 16nm standard cell library. The generalization ability of H-GAT is the most powerful with the smallest validation RE while the ratio setting to 35%, which we used in our work. Thus, it is unnecessary to include training time while predicting cell delay using our model with powerful generalization ability.

B. Performance of Predicting Cell Delay

We compare our work with the LUT-based model [5], the SVM-based model [8], Aadam [9], basic graph attention network-based (GAT) [13] and H-GAT without considering parasitics. For fair comparisons, the training and testing samples are the same. As shown in Table II, our model can achieve the highest estimation accuracy. In addition, the average RE of our work is smaller than 0.03% when the SVM-based model and Aadam reach 2.10% and 0.90%. The primary reason is that they ignore the structural information in cell circuits. Compared to other graph learning methods, considering parasitics in cell circuits improves the accuracy of our delay model. Table II shows the average runtime of cell delay prediction. Compared to SPICE, our delay model based on H-GAT can achieve obvious speedup. The average runtime of our model is 0.482 ms.

C. Performance of Calculating Path Delay

For evaluating the performance of different aging-aware timing models on path level, Table IV compares the path delay accuracy. Our model achieves better accuracy than others. Compared with the original OpenTimer, and our aging-aware STA only takes around 21 to 89 seconds for a circuit with 32K to 986K cells, which is manageable for large-scale circuit analysis with high efficiency.

V. Conclusion

In this brief, we propose an aging-aware cell timing model via graph learning, which helps optimize circuits after aging. A heterogeneous graph attention network, called H-GAT, is proposed to embed the cell by learning cell information from transistors, capacitors, resistors, timing arcs and cell structures. Experimental results on standard cells show that the aging-aware timing model based on H-GAT and the aging-aware STA using our model are accurate and fast. In future work, our method can be applied to processes below 16nm. For accuracy, we will consider the multi-input-switching effect and aging effect jointly; For efficiency, we will use transfer learning to accelerate the retraining process.

References

[1] R. Huang et al., "Variability-and reliability-aware design for 16/14nm and beyond technology," in *Proc. IEDM*, 2017, p. 12.
[2] K. Balaskas, G. Zervakis, H. Amrouch, J. Henkel, and K. Siozios, "Automated design approximation to overcome circuit aging," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 11, pp. 4710–4721, Nov. 2021.
[3] X. Zhang, Z. Zhang, Y. Lin, Z. Ji, R. Wang, and R. Huang, "Efficient aging-aware standard cell library Characterization based on sensitivity analysis," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 70, no. 2, pp. 721–725, Feb. 2023.
[4] F. Klemme and H. Amrouch, "Machine learning for on-the-fly reliability-aware cell library characterization," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 6, pp. 2569–2579, Jun. 2021.
[5] H. Amrouch, B. Khaleghi, A. Gerstlauer, and J. Henkel, "Reliability-aware design to suppress aging," in *Proc. DAC*, 2016, pp. 1–6.
[6] J. B. Velamala, V. Ravi, and Y. Cao, "Failure diagnosis of asymmetric aging under NBTI," in *Proc. ICCAD*, 2011, pp. 428–433.
[7] S. Kiamehr, F. Firouzi, M. Ebrahimi, and M. B. Tahoori, "Aging-aware standard cell library design," in *Proc. DATE*, 2014, pp. 1–4.
[8] A. Vijayan, A. Koneru, S. Kiamehr, K. Chakrabarty, and M. B. Tahoori, "Fine-grained aging-induced delay prediction based on the monitoring of run-time stress," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 5, pp. 1064–1075, May 2018.
[9] S. M. Ebrahimipour, B. Ghavami, H. Mousavi, M. Raji, Z. Fang, and L. Shannon, "Aadam: A fast, accurate, and versatile aging-aware cell library delay model using feed-forward neural network," in *Proc. ICCAD*, 2020, pp. 1–9.
[10] Synopsys. "HSPICE user guide." Accessed: 2023. [Online]. Available: https://www.synopsys.com/implementation-and-signoff/ams-simulation/primesim-hspice.html
[11] B. Tudor et al., "MOSRA: An efficient and versatile MOS aging modeling and reliability analysis solution for 45nm and below," in *Proc. ICSICT*, 2010, pp. 1645–1647.
[12] Synopsys. "PrimeTime user guide." 2023. [Online]. Available: https://www.synopsys.com/cgi-bin/imp/pdfdla/pdfr1.cgi?file=primetime-wp.pdf
[13] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.
[14] M. M. Ozdal, C. Amin, A. Ayupov, S. M. Burns, G. R. Wilke, and C. Zhuo, "An improved benchmark suite for the ISPD-2013 discrete cell sizing contest," in *Proc. ISPD*, 2013, pp. 168–170.
[15] L. Alrahis, J. Knechtel, F. Klemme, H. Amrouch, and O. Sinanoglu, "GNN4REL: Graph neural networks for predicting circuit reliability degradation," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 11, pp. 3826–3837, Nov. 2022.
[16] T.-W. Huang and M. D. Wong, "OpenTimer: A high-performance timing analysis tool," in *Proc. ICCAD*, 2015, pp. 895–902.