CNN-Assisted Low-Power Clock Tree Synthesis for 3D ICs

Chenbo Xi, Jindong Zhou, Pingqiang Zhou
School of Information Science and Technology, ShanghaiTech University, Shanghai, China
{xichb2023, zhoujd, zhoupq}@shanghaitech.edu.cn

2026 31st Asia and South Pacific Design Automation Conference (ASP-DAC) | 979-8-3315-9123-6/26/$31.00 ©2026 IEEE | DOI: 10.1109/ASP-DAC66049.2026.11420521

***Abstract*—In this work, a convolutional neural network (CNN)-assisted low-power clock topology generation method for 3D clock tree synthesis (CTS) is proposed. Our approach considers both local and global costs in each merging step to prevent getting stuck in local optima. To explore the trade-off between local and global costs, we use CNN to set two important weighting factors to obtain an optimized clock tree that reduces power consumption. Compared with the conventional NNG-based method, experimental results on ISPD09 benchmarks show that our approach can reduce wirelength by 5.83% and power consumption by 4.73% on average. We also demonstrate the transferability of our method on larger-scale ISPD10 benchmarks.**

***Index Terms*—clock tree synthesis, 3D ICs, CNN, low power.**

I. Introduction

Clock tree synthesis (CTS) plays a crucial role in the physical design process, whose primary goal is generating a clock distribution network that balances the clock arrival times among all clock sinks across the chip. The CTS process is typically divided into two steps: topology generation and clock network embedding. Method of Means and Medians (MMM) [1] is a classical method for topology generation that aims at minimizing clock skew. Since clock power is a major contributor to the chip's total power (up to 70% [2]), the nearest neighbor graph (NNG)-based method [3] is proposed to reduce clock power by minimizing the total clock wirelength. For clock embedding, deferred-merge embedding (DME) [4] is a classical method that guarantees building a clock tree with minimum wirelength and zero skew.

With the emergence of three-dimensional integrated circuits (3D ICs), researchers have looked into the problem of CTS in 3D ICs, in which the clock tree is distributed on multiple tiers. For 3D-CTS problem, the latest commercial tools [5] first partition the circuit into separate 2D tiers, and then run 2D CTS on each tier, resulting in a pseudo 3D-CTS flow. To construct a real 3D-CTS process, several works extend the 2D CTS algorithms to 3D domain [6]–[10]. For example, [9], [10] extend the 2D MMM algorithm to generate the 3D clock topology with low time complexity. To further reduce clock power, Kim et al. [7] follow the idea of NNG [3] and propose NN-3D to generate topology at the cost of larger time complexity. On top of these 3D design flows, a few works explore other topics, such as thermal [11], [12], pre-bond testability [13], [14], robustness [15], [16] and clock-gating [2], [17], [18].

This work addresses the problem of topology generation in 3D-CTS aiming at minimizing clock power. We focus on low-power 3D-CTS because power optimization is critical to alleviating the thermal and reliability challenges in 3D ICs [2], [7], [19]. In the past decade, 3D-MMM or NNG-based methods are widely adopted in 3D-CTS works [11], [20]–[22]. Our work is motivated by the observation that although NNG-based methods are superior over MMM-based methods in reducing clock power, they are essentially greedy methods that tend to get stuck in local optima. In NNG-based method, the algorithm merges the node pair with the minimum merging cost at each step. Consider the simple example shown in Fig. 1, NNG-based method chooses the node pair (a,b) to merge first, due to their closer proximity, and eventually obtains *Topology 1* in Fig. 1 (a). Applying DME to this topology yields a clock tree with a total wirelength of 13.5. However, if we merge b and c first instead, we obtain *Topology 2*, shown in Fig. 1 (b), resulting in a solution with reduced wirelength and, consequently, lower power consumption. The key reason is that although the merging cost between b and c may be initially higher, the resultant node m_1' is closer to the remaining sink a, which potentially benefits the next merge. However, NNG-based method fails to evaluate whether a local decision can reduce the overall cost in future steps.

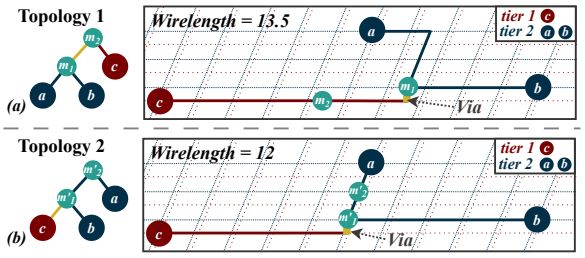

Fig. 1. (a) 3D Clock topology generated by NNG-based method. (b) Optimal topology. *Wirelength* is the clock wirelength, and the yellow lines are vias.

Inspired by the routing algorithm for signal nets [23], in which both the current cost and a predicted cost (for the future process) are considered at each step, we propose to address the local optima bottleneck of the NNG-based method by defining the merging cost function as

$$Cost = (1-\beta)\cdot Cost_{local} + \beta \cdot Cost_{global}, \qquad (1)$$

where *local cost* denotes the cost of merging two nodes a, b, and *global cost* denotes the average cost to merge the newly generated node m and the remaining nodes. We use

a weighting factor β to explore the trade-off between local and global costs. We leverage Convolutional Neural Networks (CNNs) to assist in capturing the spatial features of the clock sinks and finding the appropriate weighting factor β (together with the weighting factor α used in the local cost calculation presented in Section II) for a given circuit in terms of clock power reduction. In the literature, several works [24]–[28] have explored deep learning in the prediction of the performance metrics of clock trees in 2D IC design. In the field of 3D ICs, recent attempts to leverage deep learning techniques have focused on issues such as tier partitioning [29], TSV assignment [30] and floorplanning [31], [32]. However, to the best of our knowledge, our work is the first to explore deep learning in 3D-CTS flow, by utilizing CNNs to assist in the generation of clock tree topology.

The contributions of this work are as follows:

- We propose a low-power clock topology generation method for 3D-CTS, which considers the local merging cost alongside the global cost in each step, thereby preventing the obtained solution from getting trapped in local optima. The topology generated by our method effectively reduces power consumption, and benefits subsequent steps in the CTS process.
- We demonstrate that deep learning techniques such as CNNs can assist in determining the two important weighting factors α and β in the topology generation process. We also show that the trained models for determining weighting factors can be applied across different circuit benchmarks through transfer learning.
- We first evaluate the performance of our proposed algorithm on the ISPD09 benchmarks. Compared to the NNG-based method, our method can reduce the wirelength by an average of 5.83% and power consumption by 4.73%, with a maximum reduction of 7.29% in wirelength and 5.86% in power consumption. We further demonstrate the transferability on larger-scale ISPD10 benchmarks.

II. Preliminary

The problem of topology generation in 3D-CTS is defined as follows: given a set of clock sinks S, together with the x/y position, tier id, and load capacitance of each sink, the task of topology generation is to generate a binary tree topology, where the leaf nodes are clock sinks, and non-leaf nodes are virtual nodes, and the root of the binary tree is directly connected to the clock source.

As the first step in CTS process, a well-designed topology can benefit the subsequent network embedding step, and significantly reduce the power consumption of the clock network. 3D-MMM [9] and MMM-3D [10] are the extensions of 2D MMM [1], but with relatively high power consumption.

To reduce the power consumption, NNG-based method [7] is proposed, following a bottom-up strategy. In each step, the algorithm iterates over all the nodes that have not yet been merged and selects the pair with the minimal merging cost. Given two nodes a and b, the merging cost is:

$$Cost(a,b) = c_w \cdot (|e_a| + |e_b|) + c_v \cdot |l_a - l_b| + \alpha \cdot (C_a + C_b), \quad (2)$$

where $|e_a|$, $|e_b|$ denotes the Manhattan distance from the merging node to node a and b, l_a, l_b denote the tier ids of the nodes, and C_a, C_b are the load capacitance of a and b. c_w denotes the unit wire capacitance, and c_v denotes the unit via capacitance. α is a weighting factor for the load capacitance.

Several works also adopt the ideas of the NNG-based method, generate topologies by adjusting the merging cost function (e.g., introducing considerations for switching activity [18]). In recent work [11], the weighting factors α in Eq. (2) was directly set to 0.1 for all benchmark circuits, without accounting for the distinct characteristics of different circuits. Zhou et al. [22] have shown that the clock power can be reduced by tuning α manually, but there was limited discussion about how to find the proper value of α.

In our work, we propose to augment the definition of cost function in Eq. (2) with a global cost term, which can potentially avoid getting trapped in local optima and achieve lower power consumption. We also propose to use CNNs to find the appropriate α for each circuit.

III. Methodology

Fig. 2 illustrates the complete process of our CNN-assisted low-power CTS flow for 3D ICs, where the topology generation step can be broadly divided into two parts: a topology generation algorithm that simultaneously considers local cost and global cost, and a CNN-assisted approach to find the appropriate weighting factors. DLE-3D and DME-3D [7] are applied in subsequent clock embedding step, ensuring the minimum number of TSV and wirelength, and also guarantees to build a clock tree with minimum wirelength and zero skew.

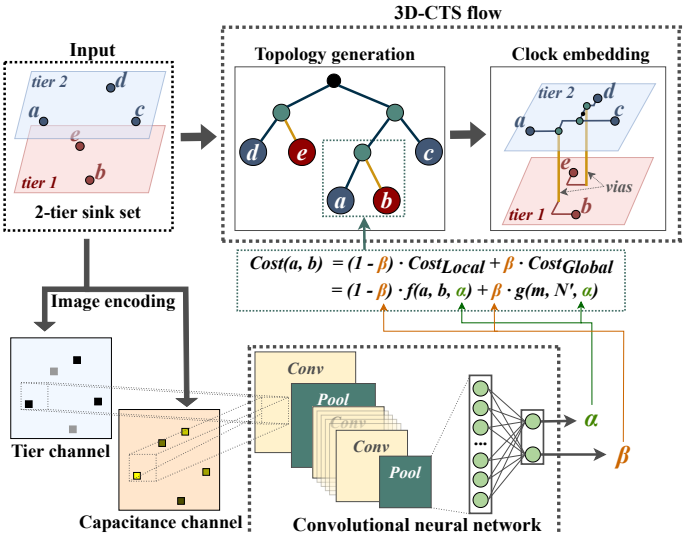

Fig. 2. The complete process of our approach. The features of the sink set are encoded into the 2-channels image. The CNN is then used to provide appropriate α and β for the subsequent topology generation in 3D-CTS flow.

A. Global Cost Aware Topology Generation

Our topology generation method is an improvement upon the NNG-based approach. We consider both local and global

costs at each step. Assume that the set N contains all the nodes that have not been merged in topology generation. Initially, N is equal to S, which is the set containing all sinks. At each step, our algorithm selects the node pair with the minimal merging cost to merge. The selected nodes are removed from N, while the newly generated node is added to the set. The process continues until there is only one element remaining in the set, at which point the algorithm terminates.

An example with $N = \{a, b, c, d\}$ is shown in Fig. 3. Node a and b are merged to form m, resulting in the updated set $N' = \{m, c, d\}$. The local merging cost evaluates the dynamic clock power consumption associated with merging the node pair (a, b), and can be defined as

$$f(a, b, \alpha) = c_w \cdot (|e_a| + |e_b|) + c_v \cdot |l_a - l_b| + \alpha \cdot (C_a + C_b), \quad (3)$$

which is the same as the original cost used in NNG-based method [7] (see Eq. (2)). Eq. (3) considers the capacitances of interconnects, vias and sub-tree, and has been widely used in the past decades [7], [11], [22]. The factor α reflects to what extent the load capacitance is considered during merging.

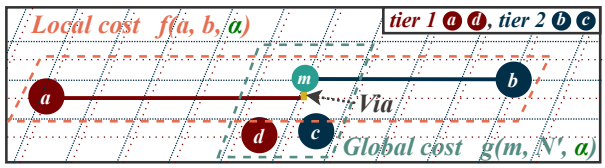

Fig. 3. Local cost and global cost for merging a and b, given $N = \{a, b, c, d\}$.

We then use a global cost term to evaluate whether a merge can benefit subsequent merging steps. The global cost of merging a and b is defined as

$$g(m, N', \alpha) = \frac{1}{|N'| - 1} \sum_{n \in N', n \neq m} f(m, n, \alpha), \quad (4)$$

For the example in Fig. 3, although the local cost $f(a, b, \alpha)$ is high, merging a and b brings m closer to the remaining nodes, thus the global cost $g(m, N', \alpha)$ is low. Notice that $g(m, N', \alpha) = 0$ when $|N'| = 1$, indicating that there is no further merging step.

Next, we use a weighting factor β to explore the trade-off between $f(a, b, \alpha)$ and $g(m, N', \alpha)$, and the overall merging cost used in our approach is defined as

$$\text{Cost}(a, b, \alpha, \beta) = (1 - \beta) \cdot f(a, b, \alpha) + \beta \cdot g(m, N', \alpha), \quad (5)$$

where $\alpha \in [0, 1]$, and $\beta \in [0, 1]$. By introducing $g(m, N', \alpha)$ and selecting an appropriate factor β, the algorithm can potentially choose node pairs that benefit the subsequent merges, even if their local merging cost is not the lowest.

B. CNN-Assisted Appropriate Factors Setting

Appropriate α and β can effectively balance the local merging cost and the global cost, and benefit the subsequent CTS steps, but the well-suited α and β may vary across different circuits. It should be emphasized that finding these factors by a simple sweeping method is unacceptable: based on our observation, the sweeping process typically takes more than 700 rounds. Therefore, in our work, we propose to address this issue by developing a CNN-based predictor.

As shown in Eq. (3), the topology generation task focuses on the spacial positions, tier ids, and capacitance of clock sinks, all of which can be encoded into images. We further apply CNN, which excels at capturing spatial features in images [33], to assist in setting the appropriate factors α and β. The model's input is a 2-channel image encoding the features of the sink set S. α and β are the outputs of the model, which are applied to Eq. (5) to help generate low-power topology.

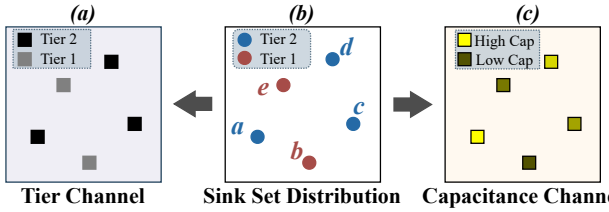

Fig. 4. A sink set S with sinks distributed on 2 tiers shown in (b) will be encoded into an image with two channels, containing the tier ids in (a), and capacitance in (c), respectively.

1) Image Encoding: To represent the unique features of a given circuit, for the clock sinks, we encode their positions, tier ids, and load capacitance into an image, which fully provides information required by Eq. (3). As shown in Fig. 4 (b), sinks are distributed on 2 tiers, and each sink has a distinct load capacitance. We encode the features into two channels of an image. In the tier channel, sinks in different tiers are represented in gray and black, respectively, while in the capacitance channel, brighter pixels indicate higher load capacitance. The spatial relationships are implicitly encoded within the image.

The image size (number of pixels) should be selected meticulously. A smaller image size could lead to overly crowded representations for circuits with a large number of sinks, causing the loss of spatial relationships between nodes; conversely, a larger image size might make it challenging for CNN to capture the positional relationships between sparsely distributed sinks. We find that setting the image size to 64×64 can effectively handle sink sets ranging from tens to thousands of sinks. Table I shows the model performance on our validation set (see Section III-B3) with different image encoding sizes (using *ResNet-18*).

TABLE I
MODEL PERFORMANCE (R^2) WITH DIFFERENT IMAGE ENCODING SIZES.

Image size	**32 × 32**	**48 × 48**	**64 × 64**	**96 × 96**	**128 × 128**
R^2 Score	0.7274	0.8860	**0.9323**	0.9242	0.9243

2) Network Structure: There are multiple structures for CNNs. We experimented with MobileNetV3 large [34], ResNet-18, ResNet-34, ResNet-50 [35] and VGG-11 [36]. Their MAE (Mean Absolute Error) and R^2 scores on the validation set (see Section III-B3) are shown in Table II.

Overall, ResNet-18 achieves the best performance and the shortest training time on the validation set; therefore, ResNet-18 is chosen as our CNN model.

TABLE II
THE PERFORMANCE AND TRAINING TIME FOR MOBILENETV3, RESNET-18, RESNET-34 AND RESNET-50.

Models	MobileNetV3	ResNet-18	ResNet-34	ResNet-50	VGG-11
MAE_a	0.0090	**0.0017**	0.0022	0.0021	0.0021
MAE_b	0.0012	**0.0003**	0.0004	0.0004	0.0004
Time	1×	**0.84×**	1.34×	2.31×	1.74×

3) Training Methodology: To train the CNN model, we need to construct a dataset, which can essentially be divided into two parts: generating the input sink sets and assigning the corresponding factor labels. To build the input sets, we manually generate 13,933 clock sink sets with random values assigned to the number, positions, tier ids, and load capacitance of clock sinks, as well as the chip size and blockage area placements, strictly referring to the distribution of sinks and obstacles in the ISPD09 benchmarks [37], as shown in Fig. 5 (a). For the randomly generated sink sets, the number of clock sinks ranges from 90 to 500 and they are arranged according to different distribution patterns. The chip area varies from 1 to $300mm^2$ with 0 to 5 blockage areas randomly placed on each chip. The dataset is entirely composed of randomly generated sink sets.

To determine the appropriate factors and assign suitable labels for each set, we iteratively obtained the corresponding combination of α and β for each sink set, along with the power consumption performance of the resulting clock trees. Specifically, we attempted to apply each combination of α and β to Eq. (5), used our method to generate the topology, and then applied network embedding to obtain the clock network, ultimately evaluating its power performance. We observed that for a given circuit, there could be multiple combinations of α and β that result in reduced power consumption. Fig. 5 (c) shows an example of the power optimization performance for each α and β, with $\alpha \in [0, 0.5]$ and $\beta \in [0, 0.06]$. As shown in Fig. 5 (c), regions with color closer to yellow indicate higher power optimization achieved by the corresponding factor combination, while darker regions represent higher power consumption. For comparison, the power performance of NNG-based method (with $\alpha = 0.1, \beta = 0$) is marked as a pink dot in the figure. We use *stable area* to denote a broadly effective region for power consumption improvement, compared with the NNG-based method. In contrast, some isolated factor combinations, termed *singular values*, may occasionally yield even better performance but are highly sensitive to minor adjustments in α or β, often causing drastic shifts in performance, as shown in Fig. 5 (c). Since we prefer factor combinations that can consistently improve the overall power performance of the clock network, we scan the factor combinations to identify the largest stable area (indicating that the power consumption can be consistently reduced) and select the central values as the labels for α and β. The encoded

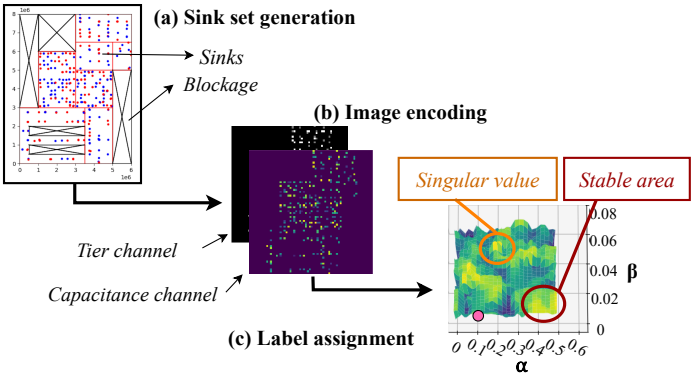

Fig. 5. The process of dataset generation; (a) generate sink sets with blockage bounds; (b) encode the features of sink sets into images; (c) traverse to get the power performance for different factor combinations. The center of a *stable area* will be chosen as the proper value of α and β.

images shown in Fig. 5 (b) are stored in the input set, while the stable factors serve as the labels for training.

It should be mentioned that for the training step, we prepare the dataset for a group of benchmarks with similar circuit scale and technical features, and then train one single model for these benchmarks, not one for each. In this way, the time cost to adopt our method in the process of CTS can be greatly reduced and the results (see Section IV) show that this strategy works well in finding the proper factors. When dealing with a circuit whose scale is much larger than those previously covered, directly applying the trained model may lead to a decline in performance. However, we can still reuse the pre-trained model via transfer learning and build a new model at a low cost. The results in Section IV-D validate the scalability of our proposed method.

IV. EXPERIMENTAL RESULTS

A. Experimental Setup

We implement the 3D-CTS flow using C/C++ on Linux environment with Intel Core i7 2.6GHz processor and 16GB memory. Buffers are inserted wherever the loading capacitance exceeds a predefined threshold. Our model is implemented in PyTorch. The dataset randomly generated in Section III-B3 is only used for training our CNN model.

ISPD09 benchmarks [37] is widely used for experiments in recent 3D-CTS works [11], [17], which contains designs from microprocessors to high-performance ASIC, derived from real IBM ASIC designs [38]. These benchmarks comprehensively account for different sink distributions and feature distinct pin capacitance allocations, thereby exhibiting an adequate level of diversity. We evaluate the performance of our approach on these benchmarks with no blockage and two-tier stacking facilitated by TSVs [6]. However, our algorithms can be applied to any tier number as well. The proposed algorithm may also be applied to other stacking technologies, such as monolithic 3D integration with MIV [39]. The number of sinks in ISPD09 benchmarks ranges from 91 to 440, with

TABLE III
PERFORMANCE OF 3D-MMM, NNG-BASED METHOD, AND OUR METHOD WITH FIXED α AND FINE-TUNED β, AND OUR METHOD WITH α, β PREDICTED BY CNN ON ISPD09 BENCHMARKS(WIRELENGTH IS DENOTED AS WL IN mm, AND POWER IN mW).

		f11	f12	f21	f22	f31	f32	f33	f34	f35	fnb1	fnb2	ratio
3D-MMM	**WL**	160	143	168	101	339	263	257	229	246	45	101	1
	#TSVs	41	44	43	18	87	65	65	40	69	76	140	1
	#Bufs	192	178	208	134	412	318	302	256	298	84	166	**1**
	Power	27.86	25.17	28.67	18.13	59.26	45.43	44.89	38.98	43.03	12.13	22.64	1
NNG-based	**WL**	142	124	148	85	293	226	224	192	211	32	80	0.8429
	#TSVs	35	33	37	27	88	57	62	54	57	43	100	0.9330
	#Bufs	210	192	204	134	404	338	320	276	298	116	202	1.0847
	Power	25.31	22.44	25.61	15.94	52.68	40.20	40.30	33.97	37.83	10.06	19.48	0.8804
Ours with α = 0.1 and fine-tuned β	**WL**	133	121	140	80	283	213	213	178	205	31	78	0.8089
	#TSVs	33	34	38	27	88	55	61	53	56	40	103	0.9233
	#Bufs	206	192	200	128	406	334	314	274	290	102	188	1.0526
	Power	**23.89**	21.94	24.52	15.14	51.23	**38.27**	38.60	**31.95**	36.96	9.76	19.02	0.8503
	β	0.032	0.012	0.028	0.004	0.008	0.016	0.010	0.014	0.028	0.034	0.012	/
Ours with α, β predicted by CNN	**WL**	133	118	140	79	279	214	213	178	199	30	75	**0.7939**
	#TSVs	35	34	37	27	88	55	61	53	52	40	102	**0.9197**
	#Bufs	202	188	200	128	406	332	314	274	294	102	186	1.0442
	Power	23.90	**21.48**	**24.41**	**14.97**	**50.63**	38.33	**38.60**	31.98	**36.01**	**9.67**	**18.68**	**0.8387**
	α	0.1015	0.1198	0.1072	0.1278	0.1068	0.1066	0.0994	0.1228	0.1043	0.1342	0.1258	/
	β	0.0104	0.0098	0.0101	0.0085	0.0096	0.0086	0.0090	0.0076	0.0087	0.0083	0.0092	/

circuit areas spanning from $5.45mm^2$ to $292.41mm^2$. The other experimental setting is aligned to [7].

3D-MMM and NNG-based methods have been widely used in the past decade [7], [9], [11], [18], [21]. Here we compare our method with the 3D-MMM method and the NNG-based method that follows Eq. (2) to calculate the merging cost, aiming at reducing the power consumption, where at each step, the method merges the node pair with the minimal merging cost. For each method, DLE-3D is further applied to ensure the number of TSVs is minimal, and DME-3D is applied to build a clock tree with minimum wirelength and zero skew. We only count the wirelength, number of TSVs, number of buffers, and power consumption of the clock tree, rather than the whole physical design. Following the practice of prior works [7], [11], [20], we obtain the clock power with SPICE simulation.

B. Evaluation of Global Cost Aware Topology Generation

As introduced in Eq. (5), when $\alpha = 0.1$, $f(a, b, \alpha)$ equals to Eq. (2), and the only difference from NNG-based method is that our algorithm additionally incorporates the impact on global merging cost, and both methods have same local cost function. As shown in Table III, with an appropriate factor β, the proposed method (denoted as *Ours ($\alpha = 0.1$ and fine-tuned β)*) can produce 3D clock trees with shorter wirelength, and less power consumption, on every benchmark. On $fnb1$ benchmark, our method achieves the highest power reduction at 19.54% compared with 3D-MMM, while the total wirelength is reduced by 31.11%. Compared with the NNG-based method, the average power consumption is reduced for 3.86%, while the total wirelength is reduced by 4.53%. Experimental results show that with a fine-tuned weighting factor β, the power consumption of the clock tree can be reduced by considering the global cost during topology generation.

Compared to NNG-based method, our approach has an additional time cost for global cost calculation in each merging step, but this is trivial – the average runtime of the conventional 3D-CTS flow is 2.61 seconds, and it only increases by 3.83% for our method, resulting in an average runtime of 2.71 seconds. Notably, although we have validated the effectiveness of applying global cost, the runtime cost for manually finding proper factor by parameter-sweeping can be unacceptable in real design flow – it takes about 1-2 hours for each benchmark. We thus leverage CNN, which averagely takes less than one second to find appropriate values for α and β.

C. Evaluation of CNN-Assisted 3D-CTS

We evaluate the performance of our model, which considers local and global merging costs with factors predicted by CNN.

The CNN model is trained for 100 epochs (taking about 30 minutes) on our randomly generated dataset before being tested on ISPD09 benchmarks. The learning rate was set to 1×10^{-5} with a batch size of 16. Adam was used as the optimizer. The dataset is split into training and validation sets at an 8:2 ratio. The pre-trained model takes images encoded from ISPD09 benchmarks as inputs and predicts the corresponding α and β, which are then applied to the proposed topology algorithm.

The experimental results are shown in Table III, the wirelength and power consumption are reduced on all benchmarks, indicating that our CNN-assisted topology generation algorithm (denoted as *Ours (α, β predicted by CNN)*) can steadily shorten the wirelength and reduce the power consumption. Compared with 3D-MMM method, on $fnb1$ benchmark, our method achieves the highest power reduction at 20.28%, while the total wirelength is reduced by 33.33%. Compared with the NNG-based method, our method still performs better on every benchmark. The power consumption is reduced by 4.73% averagely, while the total wirelength is reduced by 5.83%. The proper α and β vary for different benchmarks, demonstrating that directly applying the same factors to all circuits is inappropriate. Beyond the runtime benefits, our method outperforms *Ours ($\alpha = 0.1$ and fine-tuned β)*, whose parameters have undergone time-consuming and careful tun-

TABLE IV
PERFORMANCE OF 3D-MMM, NNG-BASED METHOD, AND OUR METHOD WITH α, β PREDICTED BY CNN ON ISPD10 BENCHMARKS (WIRELENGTH IS DENOTED AS WL IN mm, AND POWER IN mW).

		c01	c02	c03	c04	c05	c06	c07	c08	ratio
MMM	**WL**	267	639	42	107	54	44	79	48	1
	#TSVs	419	343	46	97	45	23	52	47	1
	#Bufs	454	970	172	196	96	182	240	188	**1**
	Power	59.68	126.77	20.73	26.69	12.63	15.78	26.59	18.12	1
NNG	**WL**	221	412	34	87	44	37	66	43	0.8103
	#TSVs	260	494	36	89	45	22	45	43	**0.9372**
	#Bufs	526	1024	220	258	124	164	286	196	1.1546
	Power	51.78	99.66	19.73	24.07	11.37	14.72	24.88	17.36	0.9043
Ours	**WL**	213	397	30	82	40	34	63	39	**0.7529**
	#TSVs	251	495	39	88	53	28	46	46	1.0070
	#Bufs	510	1010	214	258	114	152	258	184	1.1002
	Power	**50.38**	**97.47**	**19.22**	**23.38**	**10.90**	**14.20**	**24.24**	**16.76**	**0.8769**
	α	0.0845	0.0815	0.0877	0.0826	0.0870	0.0864	0.0873	0.0889	/
	β	0.0080	0.0069	0.0082	0.0081	0.0087	0.0080	0.0083	0.0086	/

TABLE V
PERFORMANCE OF MMM, NNG-BASED METHOD, AND OUR METHOD WITH α, β PREDICTED BY CNN ON 2D IC BENCHMARKS (WIRELENGTH IS DENOTED AS WL IN mm, AND POWER IN mW).

		f11_2D	f31_2D	f35_2D	c03_2D	c04_2D	c07_2D	ratio
MMM	**WL**	216	474	312	94	127	104	1
	#Bufs	226	486	354	199	240	234	**1**
	Power	35.53	78.14	51.87	27.77	28.52	29.15	1
NNG	**WL**	190	399	286	33	95	71	0.7367
	#Bufs	256	544	394	216	268	268	1.1187
	Power	32.03	67.81	48.64	19.24	24.28	24.92	0.8510
Ours	**WL**	187	394	284	32	94	69	**0.7252**
	#Bufs	252	546	396	206	260	262	1.0992
	Power	**31.62**	**67.17**	**48.30**	**19.04**	**24.07**	**24.73**	**0.8431**
	α	0.0795	0.082	0.0855	0.078	0.0784	0.0761	/
	β	0.0071	0.0068	0.0068	0.0069	0.0064	0.0071	/

ing, on most benchmarks. This also reveals the potential of adjusting α for power optimization.

D. Transferability

1) Larger-Scale 3D IC Benchmarks: We first validate the transferability of the CNN-assisted topology generation method on ISPD10 benchmarks [40], whose scale and sink distributions are much different from ISPD09 benchmarks and our randomly generated dataset.

Notably, ISPD10 benchmarks are from real IBM and Intel microprocessor designs [40], and do not have an inheritance relationship with the ISPD09 benchmarks. The ISPD10 benchmarks have more sinks, ranging from 981 to 2,249, and the circuit area in ISPD10 benchmarks falls between $1.5mm^2$ and $91mm^2$. To adapt the model to these larger and quite different benchmarks, we randomly generated 368 sink sets with the number of sinks ranging from 800 to 2,500 as an extended dataset, referring to the circuit sizes and number of sinks of the ISPD10 benchmarks. The CNN model from Section IV-C was further trained on the extended dataset for 100 epochs, which only takes about one minute. The resulting model takes the images encoded from ISPD10 benchmarks as input.

As shown in Table IV, our approach with transfer learning achieves reductions in wirelength and power consumption on all ISPD10 benchmarks. Compared with the 3D-MMM method (denoted as *MMM* in Table IV), the wirelength optimization rate reaches a maximum of 37.87%, and our method achieves the highest power reduction at 23.11% on $c02$ benchmark. Compared with the NNG-based method, our method still performs better on each benchmark. On average, wirelength is decreased by 6.98%, which performs even better than on ISPD09 benchmarks. Power consumption is reduced by 3.01%. Considering the large size of ISPD10 benchmarks, they may require more than 30,000 circuits to train a model from scratch, while our approach actually only needs about 2.5% extra training data over the initial dataset with 13,933 samples, to reduce the power consumption on all benchmarks, demonstrating that our model can achieve robust optimization with very little effort when dealing with new benchmarks.

2) 2D IC Benchmarks: Although our approach is initially designed for 3D-CTS (as reducing power consumption can help alleviate the thermal and reliability issues in 3D ICs [2], [7], [19]), here we show that our method can also be applied to traditional CTS on 2D ICs.

Similar to Section IV-D1, 488 sink sets are newly generated (with only one die) for transfer learning, with circuit area between $1mm^2$ and $300mm^2$, and 90 to 2,500 sinks. Six 2D IC benchmarks with different scales are selected from ISPD09 [37] and ISPD19 [40] benchmarks for evaluation, and our model from Section IV-C was further trained on the extended dataset for 100 epochs within two minutes before validation.

Results are shown in Table V. Our method reduces wirelength power consumption on each benchmark, and achieves up to 31.44% lower power consumption compared to MMM (on $c03\_2D$), and 1.28% reduction versus NNG-based (on $f11\_2D$), indicating that our model can also be applied to traditional topology generation on 2D ICs, with little effort for transfer learning. Notably, our method can achieve more improvements over NNG-based method on power optimization in 3D ICs than 2D ICs, which probably attributes to the fact that the greedy algorithm may not have fully explored the complex design solution space of 3D-CTS (due to multiple dies).

V. CONCLUSION

In this article, we propose a low-power clock topology generation method for 3D-CTS, which follows a bottom-up strategy and considers the local merging cost alongside the global cost in each step, thereby preventing the obtained solution from getting trapped in local optima. To explore the trade-off between local and global costs, we leverage CNN to set two weighting factors to help construct the clock tree topology. The experimental results show that our algorithm can effectively reduce power consumption, and can be applied across different circuit benchmarks through transfer learning.

ACKNOWLEDGMENTS

This work was supported by the Science and Technology Commission of Shanghai Municipality (STCSM) under Grant 24JD1402500.

REFERENCES

[1] M. Jackson, A. Srinivasan, and E. Kuh, "Clock routing for high-performance ICs," in *27th ACM/IEEE Design Automation Conference (DAC)*, 1990, pp. 573–579.

[2] T. Lu and A. Srivastava, "Low-power clock tree synthesis for 3D-ICs," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 22, no. 3, Apr. 2017.

[3] M. Edahiro, "A clustering-based optimization algorithm in zero-skew routings," in *30th ACM/IEEE Design Automation Conference (DAC)*, 1993, pp. 612–616.

[4] T.-H. Chao, Y.-C. Hsu, J.-M. Ho, and A. Kahng, "Zero skew clock routing with minimum wirelength," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing (TCASII)*, 1992.

[5] Cadence. Integrity 3D-IC platform white paper. [Online]. Available: https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/soc-implementation-and-floorplanning/integrity-3dic-platform.html

[6] X. Zhao, J. Minz, and S. K. Lim, "Low-power and reliable clock network design for through-silicon via (TSV) based 3D ICs," *IEEE Transactions on Components, Packaging and Manufacturing Technology (TCPMT)*, vol. 1, no. 2, pp. 247–259, 2011.

[7] T.-Y. Kim and T. Kim, "Clock tree synthesis for TSV-based 3D IC designs," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 16, no. 4, pp. 1–21, 2011.

[8] T.-Y. Kim and T. Kim, "Bounded skew clock routing for 3D stacked IC designs: Enabling trade-offs between power and clock skew," in *International Conference on Green Computing (ICGC)*, 2010.

[9] J. Minz, X. Zhao, and S. K. Lim, "Buffered clock tree synthesis for 3D ICs under thermal variations," in *2008 Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2008, pp. 504–509.

[10] T.-Y. Kim and T. Kim, "Clock tree embedding for 3D ICs," in *2010 15th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2010, pp. 486–491.

[11] D. K. Oh, M. J. Choi, and J. H. Kim, "Thermal-aware 3D symmetrical buffered clock tree synthesis," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 24, no. 3, pp. 1–22, 2019.

[12] K. Cho, C. Jang, J. Song, S. Kim, and J. Chong, "Thermal aware clock tree optimization with balanced clock skew in 3D ICs," in *The 18th IEEE International Symposium on Consumer Electronics (ISCE)*, 2014, pp. 1–2.

[13] X. Zhao, D. L. Lewis, H.-H. S. Lee, and S. K. Lim, "Low-power clock tree design for pre-bond testing of 3-D stacked ICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 30, no. 5, pp. 732–745, 2011.

[14] T.-Y. Kim and T. Kim, "Resource allocation and design techniques of prebond testable 3-D clock tree," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 32, no. 1, pp. 138–151, 2012.

[15] J.-S. Yang, J. Pak, X. Zhao, S. K. Lim, and D. Z. Pan, "Robust clock tree synthesis with timing yield optimization for 3D-ICs," in *16th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2011, pp. 621–626.

[16] M. P. D. Sai, H. Yu, Y. Shang, C. S. Tan, and S. K. Lim, "Reliable 3-D clock-tree synthesis considering nonlinear capacitive TSV model with electrical–thermal–mechanical coupling," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 32, no. 11, pp. 1734–1747, 2013.

[17] T. Lu and A. Srivastava, "Gated low-power clock tree synthesis for 3D-ICs," in *2014 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, 2014, pp. 319–322.

[18] M. Lin, H. Sun, and S. Kimura, "Power-efficient and slew-aware three dimensional gated clock tree synthesis," in *2016 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, 2016, pp. 1–6.

[19] T. Lu, C. Serafy, Z. Yang, and A. Srivastava, "Voltage noise induced dram soft error reduction technique for 3d-cpus," in *Proceedings of the 2016 International Symposium on Low Power Electronics and Design (ISLPED)*. New York, NY, USA: Association for Computing Machinery, 2016, p. 82–87.

[20] C. Ran, Q. Zeng, and Y. Cheng, "A hierarchical clock network synthesis method for 3D integrated circuits with obstacle avoidance," in *2023 International Symposium of Electronics Design Automation (ISEDA)*, 2023, pp. 345–350.

[21] W. Wang, V. F. Pavlidis, and Y. Cheng, "Zero-skew clock network synthesis for monolithic 3D ICs with minimum wirelength," in *Proceedings of the 2020 on Great Lakes Symposium on VLSI (GLSVLSI)*. Association for Computing Machinery, 2020, p. 399–404.

[22] J. Zhou, Z. Ge, C. Xi, and P. Zhou, "Inductance-aware clock network synthesis considering hierarchical interconnects in 3d ics," in *Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI)*, New York, NY, USA, 2025, p. 619–625.

[23] A. B. Kahng, L. Wang, and B. Xu, "Tritonroute: The open-source detailed router," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, pp. 547–559, 2021.

[24] S. Nagaria and S. Deb, "Designing of an optimization technique for the prediction of CTS outcomes using neural network," in *2020 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS)*, 2020, pp. 312–315.

[25] Y.-C. Lu, J. Lee, A. Agnesina, K. Samadi, and S. K. Lim, "A clock tree prediction and optimization framework using generative adversarial learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, pp. 3104–3117, 2022.

[26] A. Jain, P. Das, and A. Acharyya, "Artificial neural network based post-CTS QoR report prediction," in *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022, pp. 682–686.

[27] A. Jain, P. Das, A. Acharyya, and R. MB, "ANN-based accurate and fast post-route QoR data prediction methodology from pre-clock tree synthesis by skipping CTS and routing," in *2024 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2024, pp. 1–5.

[28] Y.-C. Lu, J. Lee, A. Agnesina, K. Samadi, and S. K. Lim, "GAN-CTS: A generative adversarial framework for clock tree prediction and optimization," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019, pp. 1–8.

[29] Y.-C. Lu, S. Pentapati, L. Zhu, G. Murali, K. Samadi, and S. K. Lim, "A machine learning-powered tier partitioning methodology for monolithic 3-D ICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 11, pp. 4575–4586, 2022.

[30] W. Guan, X. Tang, H. Lu, Y. Zhang, and Y. Zhang, "ATT-TA: A cooperative multiagent deep reinforcement learning approach for TSV assignment in 3-D ICs," *IEEE Transactions on VLSI Systems (TVLSI)*, vol. 31, no. 12, pp. 1905–1917, 2023.

[31] W. Guan, X. Tang, H. Lu, Y. Zhang, and Y. Zhang, "Thermal-aware fixed-outline 3-D IC floorplanning: An end-to-end learning-based approach," *IEEE Transactions on VLSI Systems (TVLSI)*, vol. 31, no. 12, pp. 1882–1895, 2023.

[32] W. Guan, X. Tang, H. Lu, Y. Zhang, and Y. Zhang, "A novel thermal-aware floorplanning and tsv assignment with game theory for fixed-outline 3-D ICs," *IEEE Transactions on VLSI Systems (TVLSI)*, vol. 31, no. 11, pp. 1639–1652, 2023.

[33] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[34] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for mobilenetv3," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 1314–1324.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[36] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR*, Y. Bengio and Y. LeCun, Eds., 2015.

[37] C. N. Sze, P. Restle, G.-J. Nam, and C. Alpert, "ISPD2009 clock network synthesis contest," in *Proceedings of International Symposium on Physical design (ISPD)*, 2009, pp. 149–150.

[38] C. Sze and P. Restle. Ispd 2009 contest. [Online]. Available: https://ispd.cc/contests/09/ISPD2009-cns-contest-pub.pdf

[39] P. Batude, T. Ernst, J. Arcamone, G. Arndt, P. Coudrain, and P.-E. Gaillardon, "3-D sequential integration: A key enabling technology for heterogeneous co-integration of new function with CMOS," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 2, no. 4, pp. 714–722, 2012.

[40] C. N. Sze, "ISPD 2010 high performance clock network synthesis contest: benchmark suite and results," in *Proceedings of the 19th International Symposium on Physical Design (ISPD)*, 2010, p. 143.