ASIA SOUTH PACIFIC DAC DESIGN AUTOMATION CONFERENCE

ASP-DAC 2024

29th Asia and South Pacific Design Automation Conference

An Optimization-aware Pre-Routing Timing Prediction Framework Based on Heterogeneous Graph Learning

Guoqing He, Wenjie Ding, Yuyang Ye, Xu Cheng, Qianqian Song, and Peng Cao*
Southeast University
{hegq, caopeng}@seu.edu.cn

Jan. 23, 2024

Outline

- **Introduction**
- Preliminaries
- Method
- Experimental Result and Discussion
- Conclusion

Introduction

- Static Timing Analysis (STA) engines are frequently invoked in all stages of design for timing convergence.
- Timing estimation result at placement stage is **inaccurate** due to the absence of wire parasitic.
- An **accurate** and **effective** pre-routing timing model is crucial for timing-driven placement.

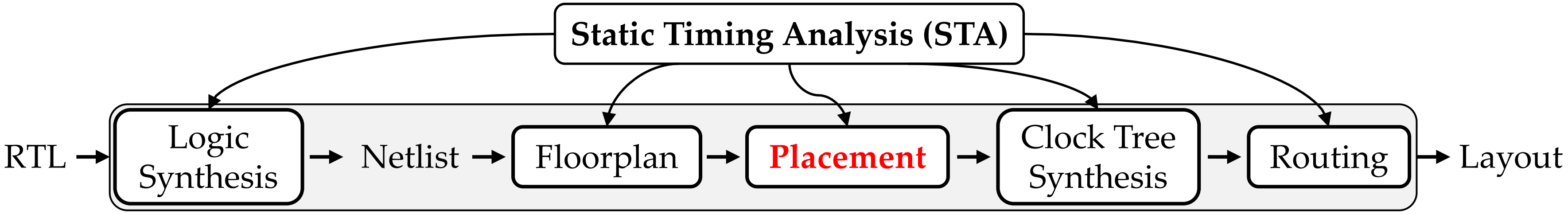

Fig. 1 Static timing analysis in design flow.

Introduction

- Analytical Methods

  Rough wire length or wire parasitic estimation

- Machine Learning (ML)-based Methods[1,2,3,4]

  Learn from real routing information to consider the impact of routing

  Transferable timing prediction model and accurate timing prediction results

  Two categories: **local** timing metric-based[1,2] and **global** timing metric-based[3,4]

1 E. C. Barboza, N. Shukla, Y. Chen, and J. Hu, "Machine learning-based pre-routing timing prediction with reduced pessimism," in *Proc. DAC*, 2019.
2 X. He, Z. Fu, Y. Wang, C. Liu, and Y. Guo, "Accurate timing prediction at placement stage with look-ahead rc network," in *Proc. DAC*, 2022.
3 Z. Guo, M. Liu, J. Gu, S. Zhang, D. Z. Pan, and Y. Lin, "A timing engine inspired graph neural network model for pre-routing slack prediction," in *Proc. DAC*, 2022.
4 P. Cao, G. He, and T. Yang, "Tf-predictor: Transformer-based pre-routing path delay prediction framework," *IEEE TCAD*, 2023.

Introduction

- Why Optimization-aware Method is Needed?

  Timing optimization techniques, e.g. gate-sizing and buffer insertion, are necessarily utilized at routing stage to fix timing violations induced by wire parasitic after routing.

  These timing changes are **non-negligible**!

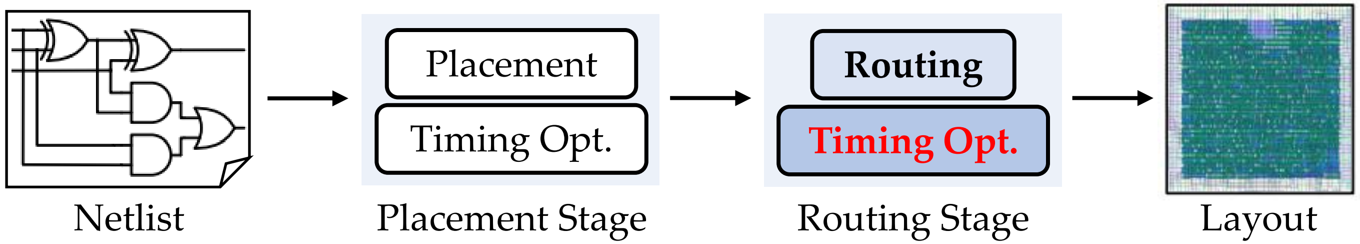

Fig. 2 Physical design flow with timing optimization procedures.

Introduction

- Two **Challenges** For Timing Optimization-aware Timing Prediction

  How to jointly consider the information of cells and pins?

  Timing optimization techniques, e.g. gate-sizing, are performed at **cell-level**, while timing metrics, e.g. slew, arrival time and slack, are changed and updated at **pin-level**.

  How to collect the information on timing paths?

  For gate-sizing techniques, the size of cells on **timing paths** with small slack can be up-sized for improving timing performance and those with large slack can be down-sized for saving area.

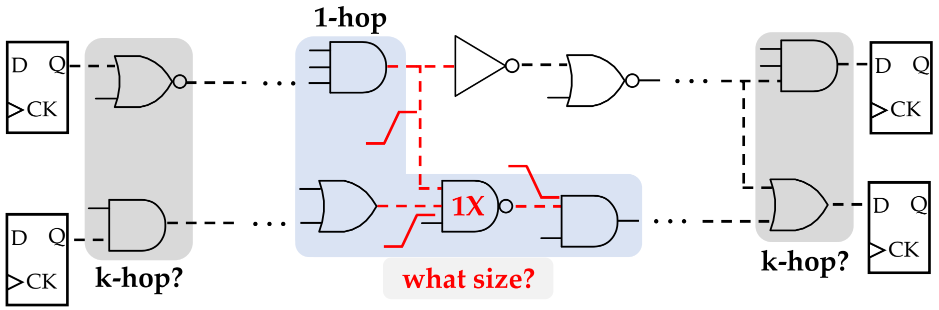

Fig. 3. Illustration of netlist before routing and timing optimization.

Introduction

- An end-to-end optimization-aware pre-routing timing prediction framework based on **H**eterogeneous **G**raph **AT**tention and **Trans**former network, **HGATTrans**, is proposed to predict timing changes introduced by later routing and associated optimization procedures at placement stage.
- A customized heterogeneous graph attention network (HGAT) is developed, which introduces a message passing mechanism between cell nodes and pin nodes to embed local features.
- Transformer network is further utilized to capture global information from the timing path.

Outline

- Introduction
- **Preliminaries**
- Method
- Experimental Result and Discussion
- Conclusion

Preliminaries

- Graph Neural Networks (GNNs)

  GNNs have been widely employed in electronic design automation (EDA) field for netlist structure learning and information mining.

  Common homogeneous GNNs: only one message passing scheme for only one relation.

  Heterogeneous GNNs: **Different message passing schemes for different relations**. The message passing scheme for node i in a heterogeneous graph can be expressed as:

$$\boldsymbol{h}_i^{(k)} = \underbrace{\boxed{\bigoplus_{r\in\mathcal{R}}}}_{\text{aggregation scheme for different relations}} \underbrace{\boxed{\gamma^{(r)}}}_{\text{update function under relation } r} \left(\boldsymbol{h}_i^{(k-1)}, \underbrace{\boxed{\bigoplus_{j\in\mathcal{N}_{(r)}(i)}}}_{\text{aggregation scheme under relation } r} \underbrace{\boxed{\phi^{(r)}}}_{\text{aggregation function under relation } r} \left(\boldsymbol{h}_i^{(k-1)}, \boldsymbol{h}_j^{(k-1)}, \boldsymbol{e}_{j,i}^{(k-1)} \right) \right).$$

Preliminaries

- Graph Neural Networks (GNNs)

For the homogeneous graph in Fig. 4(a), the embedding of node 1 in k-th layer is:

$$\boldsymbol{h}_1^{(k)} = \mathbf{W}_1 \boldsymbol{h}_i^{(k-1)} + \underbrace{\boxed{\mathbf{W}_2 \left(\boldsymbol{h}_2^{(k-1)} + \boldsymbol{h}_3^{(k-1)} + \boldsymbol{h}_4^{(k-1)} \right)}}_{N(1) = \{2,\, 3,\, 4\}}.$$

For the heterogeneous graph with two relations in Fig. 4(b), it is:

$$\boldsymbol{h}_1^{(k)} = \mathbf{W}_1 \boldsymbol{h}_i^{(k-1)} + \underbrace{\boxed{\mathbf{W}_2 \left(\boldsymbol{h}_2^{(k-1)} + \boldsymbol{h}_3^{(k-1)} + \boldsymbol{h}_4^{(k-1)} \right)}}_{N_1(1) = \{2,\, 3,\, 4\}} + \underbrace{\boxed{\mathbf{W}_3 \left(\boldsymbol{h}_5^{(k-1)} + \boldsymbol{h}_6^{(k-1)} + \boldsymbol{h}_7^{(k-1)} \right)}}_{N_2(1) = \{5,\, 6,\, 7\}}.$$

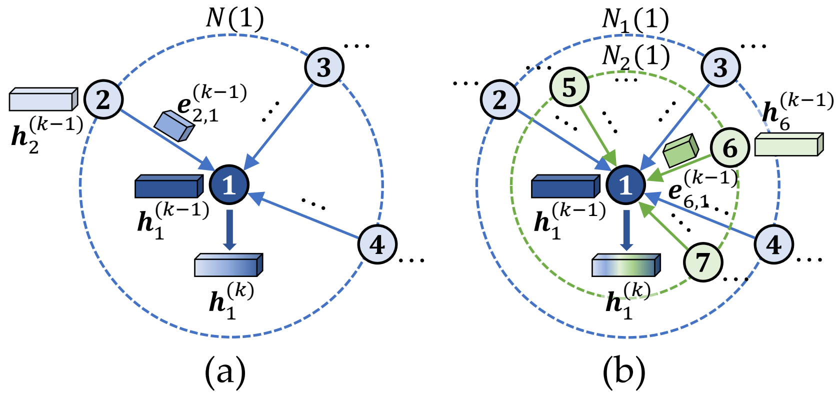

Fig. 4 An illustration of message passing in GNN. (a) Homogeneous GNN with one relation. (b) Heterogeneous GNN with two relations.

Preliminaries

- Transformer in Graph Learning

  Transformer network have achieved success in solving the over-smoothing problem of deep GNNs.

  Multi-head self-attention mechanism is at the heart of transformer network. It can be expressed as:

$$Attention\left(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}\right)=softmax\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\top}}{\sqrt{d_k}}\right)\boldsymbol{V},$$

$$MultiHead\left(\boldsymbol{X}\right)=Concat\left(head_1,...,head_h\right)\boldsymbol{W}^{O},$$

$$head_i=Attention\left(\boldsymbol{X}\boldsymbol{W}_i^{Q},\boldsymbol{X}\boldsymbol{W}_i^{K},\boldsymbol{X}\boldsymbol{W}_i^{V}\right).$$

  Transformer network helps GNNs capture **high-level and long-range** dependencies, and expands the receptive field to the entire graph.

Outline

- Introduction
- Preliminaries
- **Method**
- Experimental Result and Discussion
- Conclusion

Method

- Overview

HGATTrans, an end-to-end optimization-aware pre-routing timing prediction framework based on **H**eterogeneous **G**raph **AT**tention and **Trans**former network.

HGATTrans can perceive the cell-level optimization techniques, i.e. gate sizing.

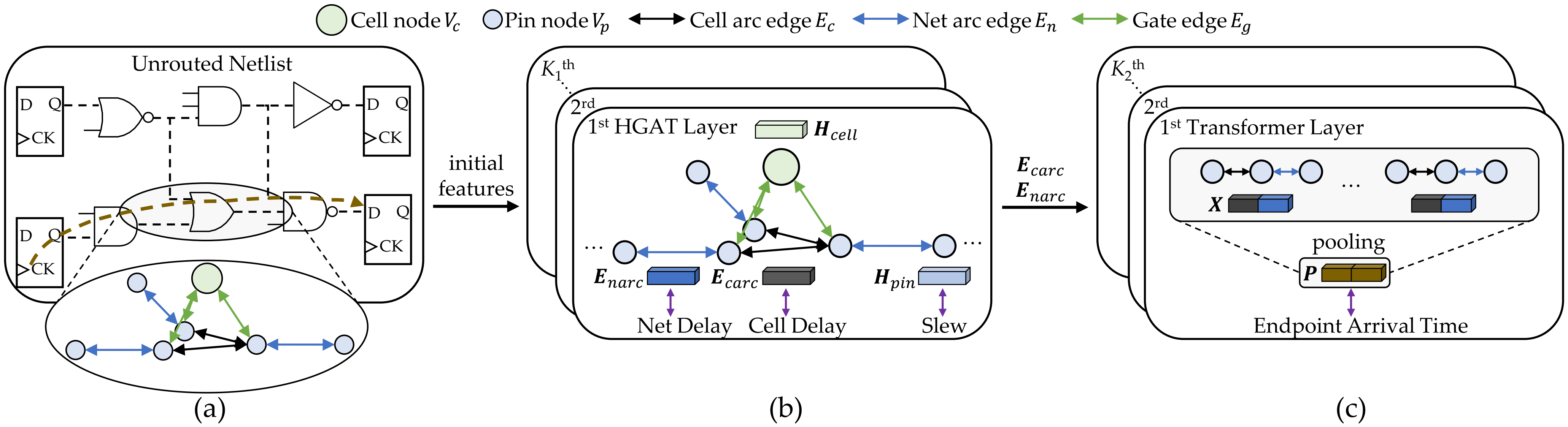

Fig. 5 Overview of the proposed timing prediction framework. (a) Heterogeneous graph representation. (b) HGAT network to generate embedding from local view. (c) Transformer network to generate embedding from global view.

Method

- Overview

The given unrouted netlist is **firstly** represented by a heterogeneous graph.

There are two types of node, i.e. cell node V_c and pin node V_p, and three types of edges, i.e. cell arc edge E_c, net arc edge E_n and gate edge E_g in the heterogeneous graph.

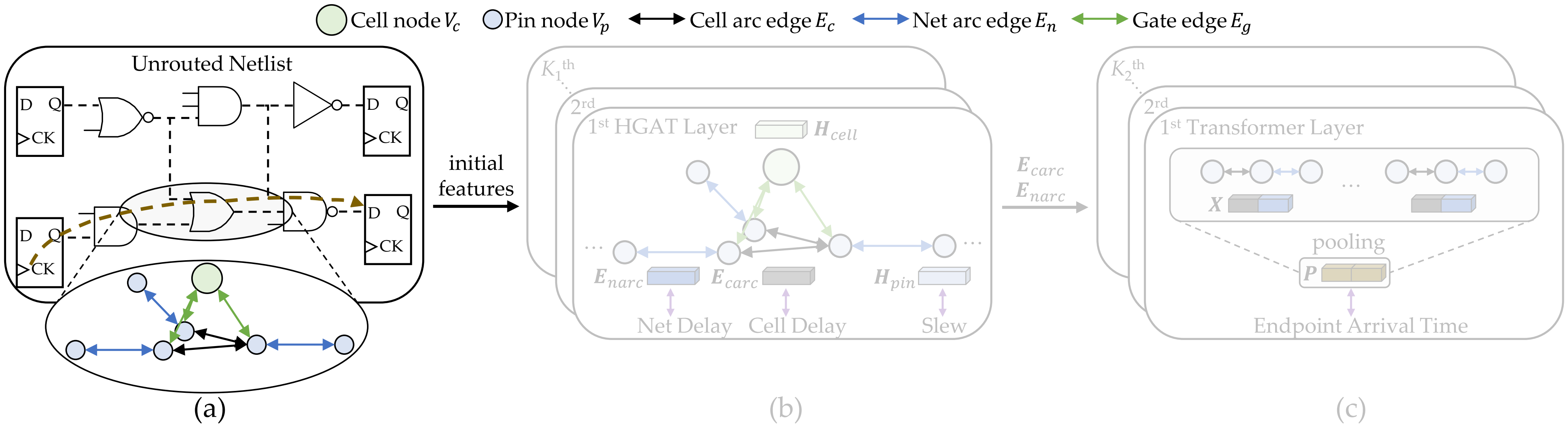

Fig. 5 Overview of the proposed timing prediction framework. (a) Heterogeneous graph representation. (b) HGAT network to generate embedding from local view. (c) Transformer network to generate embedding from global view.

Method

- Overview

  K_1-layer heterogeneous graph attention (HGAT) network is **then** utilized to generate local embeddings of cell node $\boldsymbol{H}_{cell}$, pin node $\boldsymbol{H}_{pin}$, cell arc edge $\boldsymbol{E}_{carc}$, and net arc edge $\boldsymbol{E}_{narc}$, where $\boldsymbol{H}_{pin}$, $\boldsymbol{E}_{carc}$, and $\boldsymbol{E}_{narc}$ are obtained to predict slew, cell delay, and net delay.

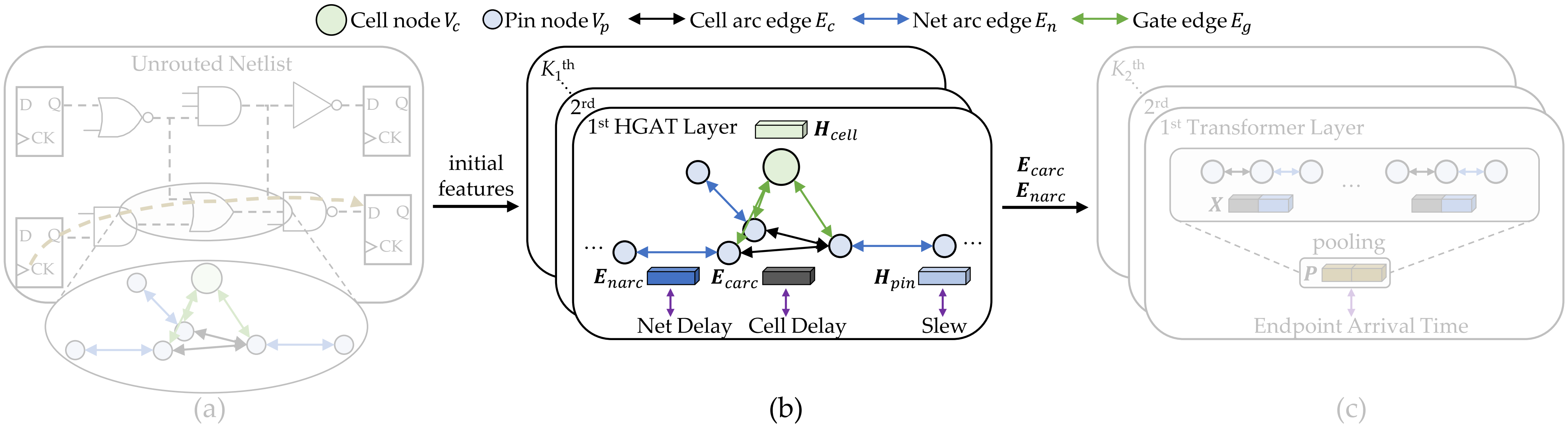

Fig. 5 Overview of the proposed timing prediction framework. (a) Heterogeneous graph representation. (b) HGAT network to generate embedding from local view. (c) Transformer network to generate embedding from global view.

Method

- Overview

  Over-smoothing problem of deep GNNs hinders the learning of global information.

  K_2-layer transformer network and pooling operation are **finally** utilized for global path embedding generation for endpoint arrival time prediction.

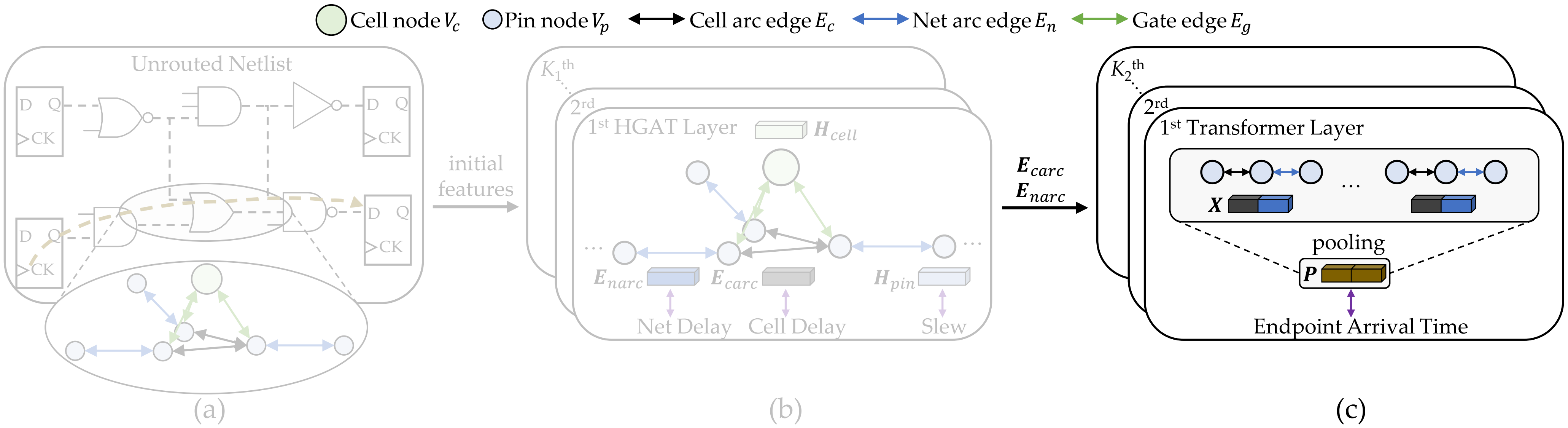

Fig. 5 Overview of the proposed timing prediction framework. (a) Heterogeneous graph representation. (b) HGAT network to generate embedding from local view. (c) Transformer network to generate embedding from global view.

Method

- Netlist Heterogeneous Graph Representation

  The message passing mechanism between cell nodes and pin nodes is introduced by **gate edge E_g** to capture the relationship between the physical and timing information of **pin** and **its corresponding cell**.

influence **cell-level timing optimization techniques** **pin-level timing metrics** **guidance**

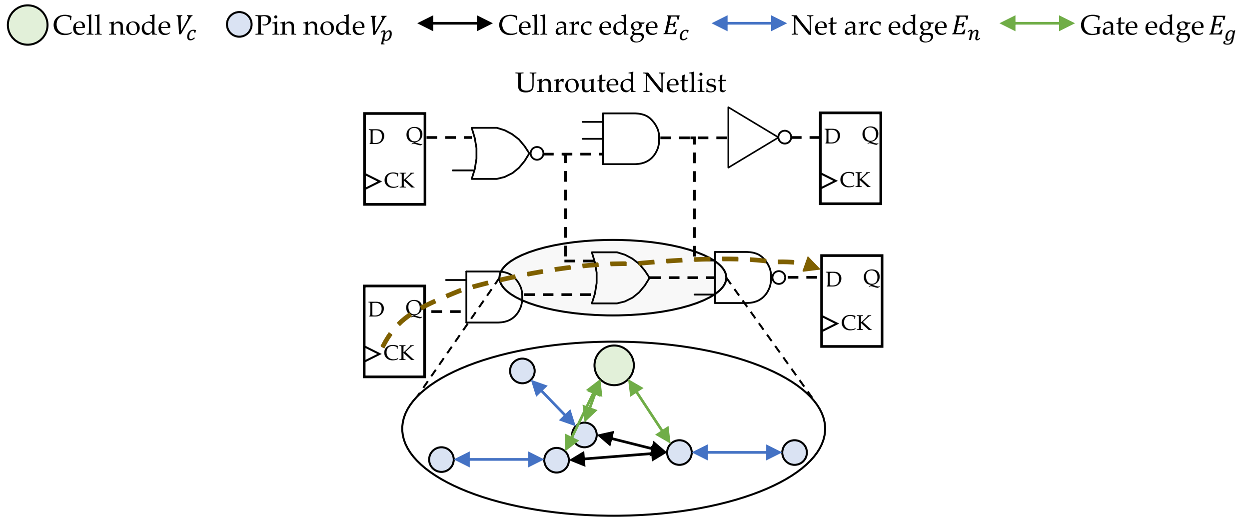

Fig. 5 (a) Heterogeneous graph representation.

Method

- Netlist Heterogeneous Graph Representation

Initial node and edge features are extracted for each type of nodes and edges to generate high-dimensional node and edge embeddings by HGAT.

Table I Initial node and edge features in the heterogeneous graph.

Type	Name	Description
Cell node	function type	function type of cell
	drive strength	drive strength of cell
	threshold voltage type	threshold voltage type of cell
	is sizable or not	cell is sizable or not
	# fanins	fanin number of cell
	# fanouts	fanout number of cell
	wst delay	worst delay of cell
	wst slew	worst pin transition time of cell
Pin node	is output pin or not	pin is cell output pin or not
	x/y coordinate	coordinate of pin in x/y direction
	r/f cap	rise/fall capacitance of pin
	r/f slew	rise/fall transition time of pin
	r/f slack	rise/fall slack of pin
Cell arc edge	r/f delay	rise/fall delay of cell arc
Net arc edge	x/y distance	distance of net along x/y direction

Method

- Node and Edge Embedding with HGAT network

  ***K_1*-layer HGAT network** for local cell node, pin node, cell arc edge, and net arc embedding generation.

  Node embedding and edge embedding are aggregated and updated by the attention mechanism and go through batch normalization layer.

  The embeddings of all layer are preserved and concatenated by residual connections. The final output of node embedding and edge embedding are $\boldsymbol{H} = \boldsymbol{W}_h^{K_1}\left[\boldsymbol{H}^{(1)}\| \cdots \|\boldsymbol{H}^{(K_1)}\right]$ and $\boldsymbol{E} = \boldsymbol{W}_e^{K_1}\left[\boldsymbol{E}^{(1)}\| \cdots \|\boldsymbol{E}^{(K_1)}\right]$.

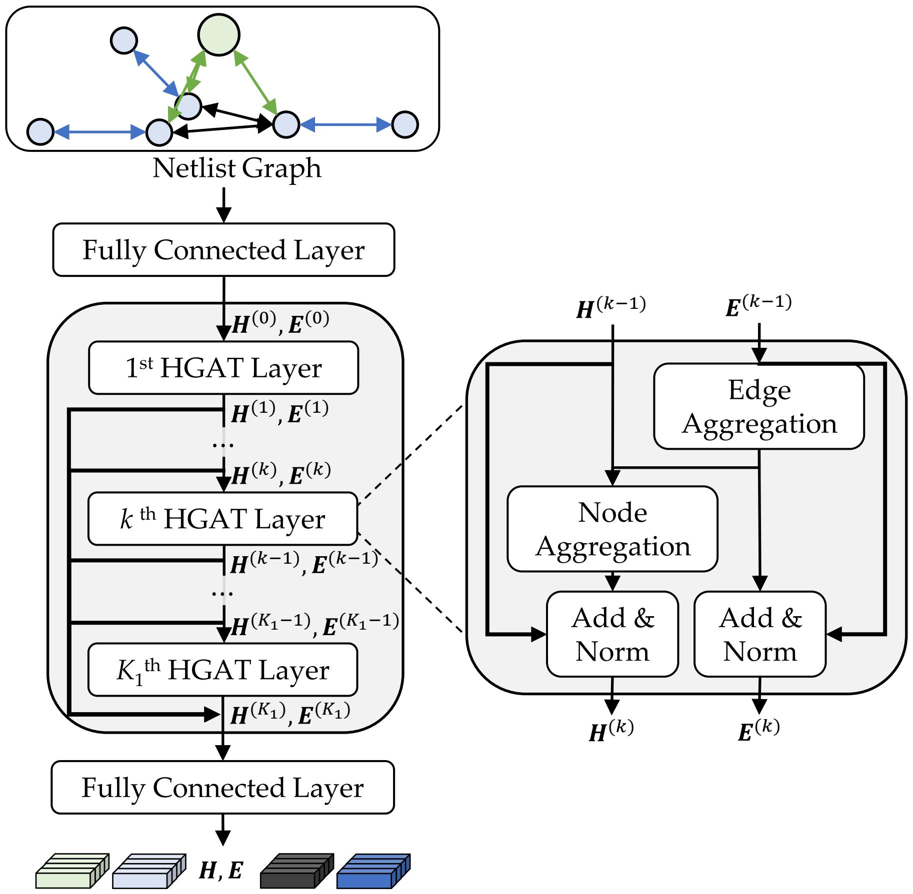

Fig. 6 The structure of HGAT, where residual connections are introduced among layers and illustrated in **bold** line. The embedding of cell node, pin node, cell arc edge, and net arc edge are generated after aggregation and update layer by layer.

Method

- Node and Edge Embedding with HGAT network

Edge aggregation: $$\boldsymbol{e}_{p_j,p_i}^{(k)} = \boldsymbol{W}_e \left[\boldsymbol{h}_{p_i}^{(k-1)} \| \boldsymbol{h}_{p_j}^{(k-1)} \| \boldsymbol{e}_{p_j,p_i}^{(k-1)}\right].$$

Node aggregation: **cell node:** $$\boldsymbol{h}_{c_i}^{(k)} = \boldsymbol{W}_{c1}\boldsymbol{h}_{c_i}^{(k-1)} + \boldsymbol{W}_{c2} \sum_{p_j \in \mathcal{N}(c_i)} \alpha_{p_j,c_i} \boldsymbol{h}_{p_j}^{(k-1)},$$

pin node: $$\boldsymbol{h}_{p_i}^{(k)} = \boldsymbol{W}_{p1}\boldsymbol{h}_{p_i}^{(k-1)} + \boldsymbol{W}_{p2}\boldsymbol{h}_{c_i}^{(k-1)} + \boldsymbol{W}_{p3} \sum_{p_j \in \mathcal{N}_{carc}(p_i)} \alpha_{p_j,p_i} \boldsymbol{h}_{p_j}^{(k-1)} + \boldsymbol{W}_{p4} \sum_{p_j \in \mathcal{N}_{narc}(p_i)} \alpha_{p_j,p_i} \boldsymbol{h}_{p_j}^{(k-1)}.$$

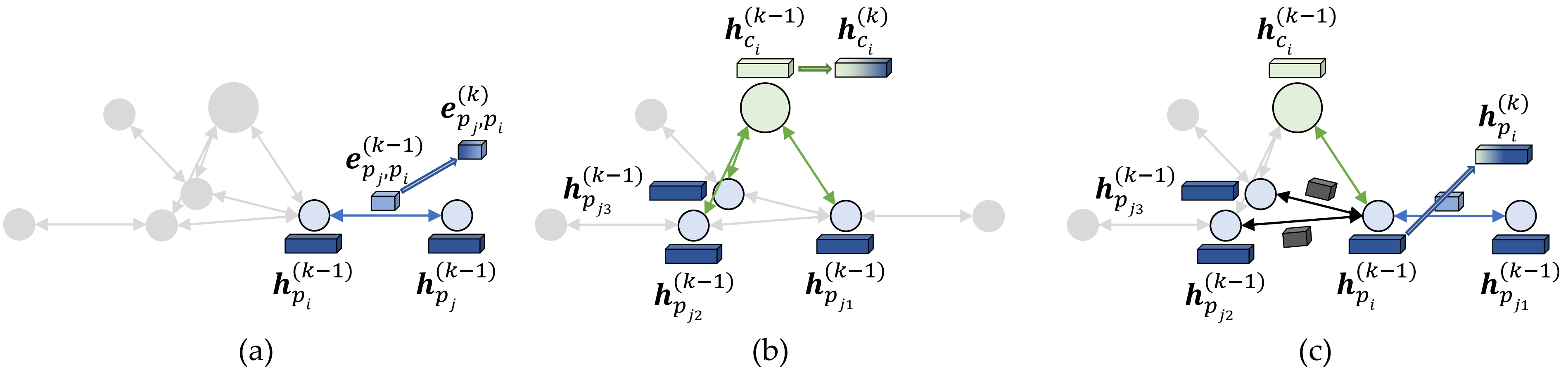

Fig. 7 Illustration of message passing schemes in the netlist heterogeneous graph. (a) Edge. (b) Cell node. (c) Pin node.

Method

- Path Embedding with Transformer network

  ***K_2*-layer transformer network** and pooling operation for path embedding generation.

  Stage embedding is the concatenation of the learned cell arc embedding and the corresponding net arc embedding of HGAT, i.e. $x_{s_i} = e_{carc_i} \| e_{narc_i}$.

  Timing and physical correlation among stages along the timing path is captured by multi-head attention.

  Path embedding is got by mean pooling, i.e.

$$p_i = \frac{1}{|S(i)|} \sum_{s_j \in \mathcal{S}(i)} x_{s_j}^{(K_2)}.$$

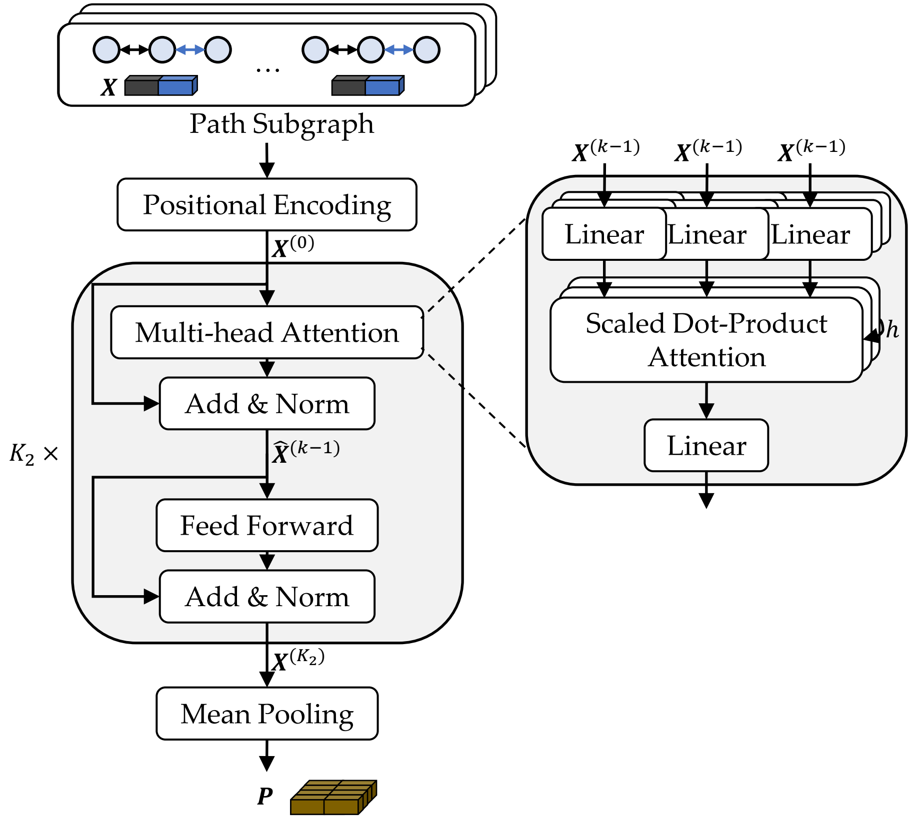

Fig. 8 The structure of transformer network and multi-head self-attention layer. Path embedding is generated by performing layer-by-layer attention calculation and pooling operation on stage embeddings.

Method

- End-to-End Multi-Objective Training

The difference of cell arc delay, net arc delay, and slew between pre- and post-routing stages are used as labels for HGAT network multi-objective training. There three loss functions are:

$$\mathcal{L}_{cd}\left(\theta_G, \phi_c | \boldsymbol{E}_{carc}\right) = \frac{1}{N_{carc}} \| MLP_c^{\phi_c}\left(\boldsymbol{E}_{carc}\right) - \Delta \boldsymbol{CD} \|_2^2,$$

$$\mathcal{L}_{nd}\left(\theta_G, \phi_n | \boldsymbol{E}_{narc}\right) = \frac{1}{N_{narc}} \| MLP_n^{\phi_n}\left(\boldsymbol{E}_{narc}\right) - \Delta \boldsymbol{ND} \|_2^2,$$

$$\mathcal{L}_{slew}\left(\theta_G, \phi_s | \boldsymbol{H}_{pin}\right) = \frac{1}{N_{pin}} \| MLP_p^{\phi_s}\left(\boldsymbol{H}_{pin}\right) - \Delta \boldsymbol{S} \|_2^2.$$

The total loss function of HGAT network is:

$$\mathcal{L}_{tot}\left(\theta_G, \phi_c, \phi_n, \phi_s\right) = \mathcal{L}_{cd}\left(\theta_G, \phi_c\right) + \mathcal{L}_{nd}\left(\theta_G, \phi_n\right) + \mathcal{L}_{slew}\left(\theta_G, \phi_s\right).$$

The difference of endpoint arrival time is used as label for transformer network training. The loss function is:

$$\mathcal{L}_{eat}\left(\theta_T, \phi_p | \boldsymbol{P}\right) = \frac{1}{N_{path}} \| MLP^{\phi_p}\left(\boldsymbol{P}\right) - \Delta \boldsymbol{EAT} \|_2^2.$$

Outline

- Introduction
- Preliminaries
- Method
- **Experimental Result and Discussion**
- Conclusion

Experimental Result and Discussion

- Experiment Setup

15 benchmarks from OpenCores were placed and routed with TSMC 22nm process, where 10 benchmarks were randomly selected as training set.

Table II Benchmark Statistics.

	Benchmark	Benchmark Statistics				
		# Cells	# Pins	# Cell arcs	# Net arcs	# Endpoints
Training	des_area	4817	15178	19870	19994	128
	systemcdes	3297	10128	12908	12924	190
	wb_dma	3494	11687	14402	14648	494
	tv80	7530	24873	33848	33676	374
	systemcaes	8917	29544	39076	39206	702
	ac97_ctrl	12378	39971	50598	50392	1839
	aes_core	21492	64374	83912	83858	535
	wb_conmax	29224	110358	155640	158000	770
	des_perf	97845	295040	375854	371394	8740
	vga_lcd	120751	385765	479122	495600	25109
	Average Train	30975	98692	126523	127969	3888
Testing	spi	3363	10862	14366	14298	229
	mem_ctrl	12333	41158	54958	55076	939
	usb_funct	12960	42136	54454	54578	125
	pci_bridge32	16858	55238	69332	69636	1464
	ethernet	49969	177484	233542	233680	10001
	Average Test	19097	65376	85330	85454	2552

Experimental Result and Discussion

- Stage Delay Prediction Performance

  HGAT achieves the **best generalization** on testing designs with average R^2 score of 0.9464.

  Two previous stage-based prediction models are prone to over-fitting, lacking local information and optimization perception.

  The introduced cell nodes, residual connections and delta labels all help improve performance.

Table III Stage Delay Comparison.

Benchmark	Stage Delay Comparison (R^2 Score)						
	Placement	DAC'19	DAC'22	HGAT w/o cell	HGAT w/o residual	HGAT w/o delta	HGAT full
des_area	0.6057	0.9944	0.9984	0.9717	0.9776	0.9795	0.9840
|	|	|	|	|	|	|	|
vga_lcd	0.5582	0.9911	0.9978	0.9450	0.9479	0.9525	0.9649
Average Train	**0.7221**	**0.9900**	**0.9973**	**0.9516**	**0.9579**	**0.9627**	**0.9714**
spi	0.6697	0.8661	0.8988	0.9306	0.9421	0.9413	0.9360
mem_ctrl	0.6835	0.8714	0.9230	0.9263	0.9414	0.9428	0.9627
usb_funct	0.6735	0.8090	0.8851	0.9243	0.9357	0.9473	0.9660
pci_bridge32	0.6620	0.8607	0.9345	0.9239	0.9334	0.9370	0.9606
ethernet	0.3486	0.7938	0.8575	0.9022	0.9021	0.9046	0.9066
Average Test	**0.6075**	**0.8402**	**0.8998**	**0.9215**	**0.9309**	**0.9346**	**0.9464**

Experimental Result and Discussion

- Endpoint Arrival Time Prediction Performance

  HGATTrans achieves the **highest accuracy** on testing designs with average R^2 score of 0.9116.

  Prediction error of global timing metric is accumulated in stage-based prediction models.

  Accuracy of transformer alone model and HGAT alone model are also limited.

  The fusion of local information and global information achieves performance improvement.

Table IV Endpoint Arrival Time Comparison.

Benchmark	Endpoint Arrival Time Comparison (R^2 Score)					
	Placement	DAC'19	DAC'22	TCAD'23	HGAT	HGATTrans
des_area	0.7272	0.9244	0.9769	0.8522	0.9794	0.9873
|	|	|	|	|	|	|
vga_lcd	-0.9893	0.9670	0.9546	0.9709	0.9108	0.9663
Average Train	**0.3040**	**0.9599**	**0.9461**	**0.9181**	**0.9265**	**0.9719**
spi	0.7292	0.7242	0.9186	0.9485	0.9627	0.9876
mem_ctrl	0.3741	0.6466	0.8242	0.9052	0.9225	0.9403
usb_funct	0.7386	0.8282	0.7974	0.8098	0.7837	0.9361
pci_bridge32	0.0276	0.3657	0.7488	0.8416	0.8539	0.9165
ethernet	-6.7674	-0.2220	0.3811	0.4312	0.5506	0.7775
Average Test	**-0.9796**	**0.4685**	**0.7340**	**0.7873**	**0.8147**	**0.9116**

Experimental Result and Discussion

- Runtime Analysis

The inference time of HGATTrans can bring 1206× **speedup** compared with traditional flow.

Table V Runtime Comparison.

Benchmark	Runtime Comparison (Second)						
	Traditional Flow			HGATTrans			Speedup
	Routing	STA	Total	HGAT	Transformer	Total	
des_area	225	11	236	0.02	0.05	0.07	3189×
systemcdes	210	9	219	0.02	0.08	0.10	2134×
wb_dma	214	10	224	0.02	0.26	0.28	793×
tv80	291	11	301	0.02	0.20	0.22	1358×
systemcaes	300	13	313	0.03	0.40	0.43	726×
ac97_ctrl	334	15	349	0.03	0.86	0.88	395×
aes_core	417	16	433	0.04	0.26	0.30	1447×
wb_conmax	621	22	643	0.07	0.41	0.49	1325×
des_perf	1041	47	1088	0.16	4.04	4.21	259×
vga_lcd	1500	81	1581	0.21	12.12	12.33	128×
Average Train	515	24	539	0.06	1.87	1.93	**1175**×
spi	231	9	240	0.02	0.15	0.17	1436×
mem_ctrl	353	14	367	0.03	0.42	0.45	810×
usb_funct	266	13	279	0.04	0.05	0.09	3245×
pci_bridge32	298	17	315	0.04	0.72	0.76	416×
ethernet	560	37	597	0.10	4.71	4.81	124×
Average Test	342	18	360	0.05	1.21	1.26	**1206**×

Outline

- Introduction
- Preliminaries
- Method
- Experimental Result and Discussion
- **Conclusion**

Conclusion

- An accurate and efficient optimization-aware pre-routing timing prediction model is desired for timing-driven placement.
- HGATTrans is proposed to calibrate the timing changes introduced by routing and associated timing optimization procedures.
- The heterogeneous message passing mechanism is customized and transformer network is introduced for optimization perception and receptive field expansion.
- The proposed model achieves better performance than previous work and significant speedup than traditional flow.

THANK YOU!