LLSM: LLM-enhanced Logic Synthesis Model with EDA-guided CoT Prompting, Hybrid Embedding and AIG-tailored Acceleration

Shan Huang*, Jinhao Li*, Zhen Yu, Jiancai Ye, Jiaming Xu, Ningyi Xu, Guohao Dai

*Equal contribution

Shanghai Jiao Tong University

Correspondence to:

Guohao Dai <daiguohao@sjtu.edu.cn>

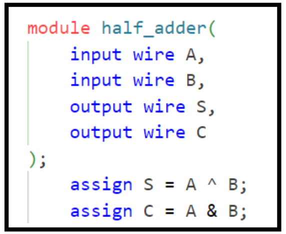
ASP-DAC 2025

Outline

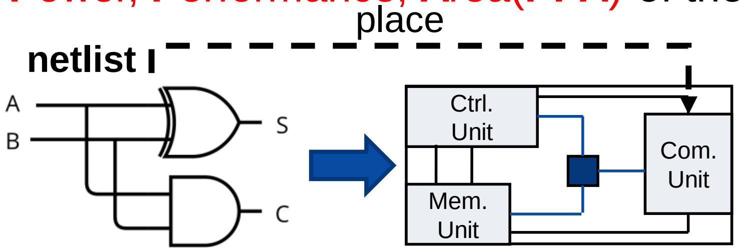
Electronic Design Automation(EDA)

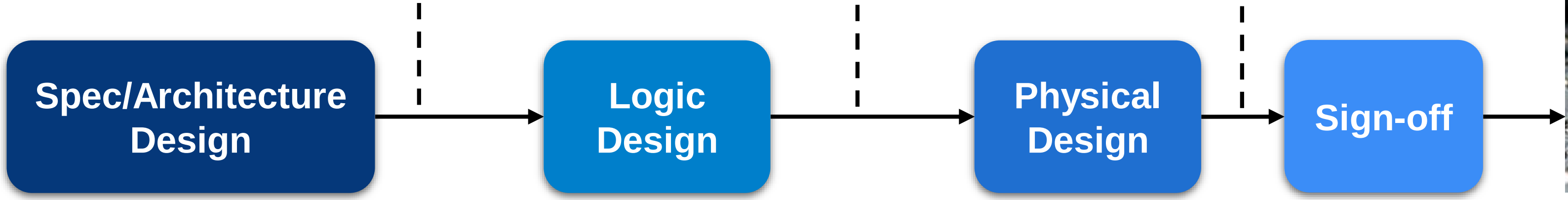
- EDA refers to the use of EDA software tools to complete the functional design, synthesis, verification, physical design of VLSI chips.
  - **Key objective**: Optimize the **P**ower, **P**erformance, **A**rea(**PPA**) of the chip.

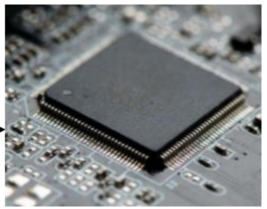
RTL code written by engineers

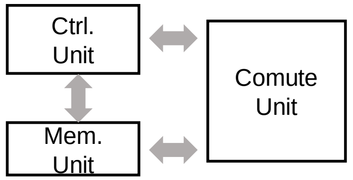


```
module half_adder(
    input wire A,
    input wire B,
    output wire S,
    output wire C
);
    assign S = A ^ B;
    assign C = A & B;
```









Logic optimization & Map to a netlist

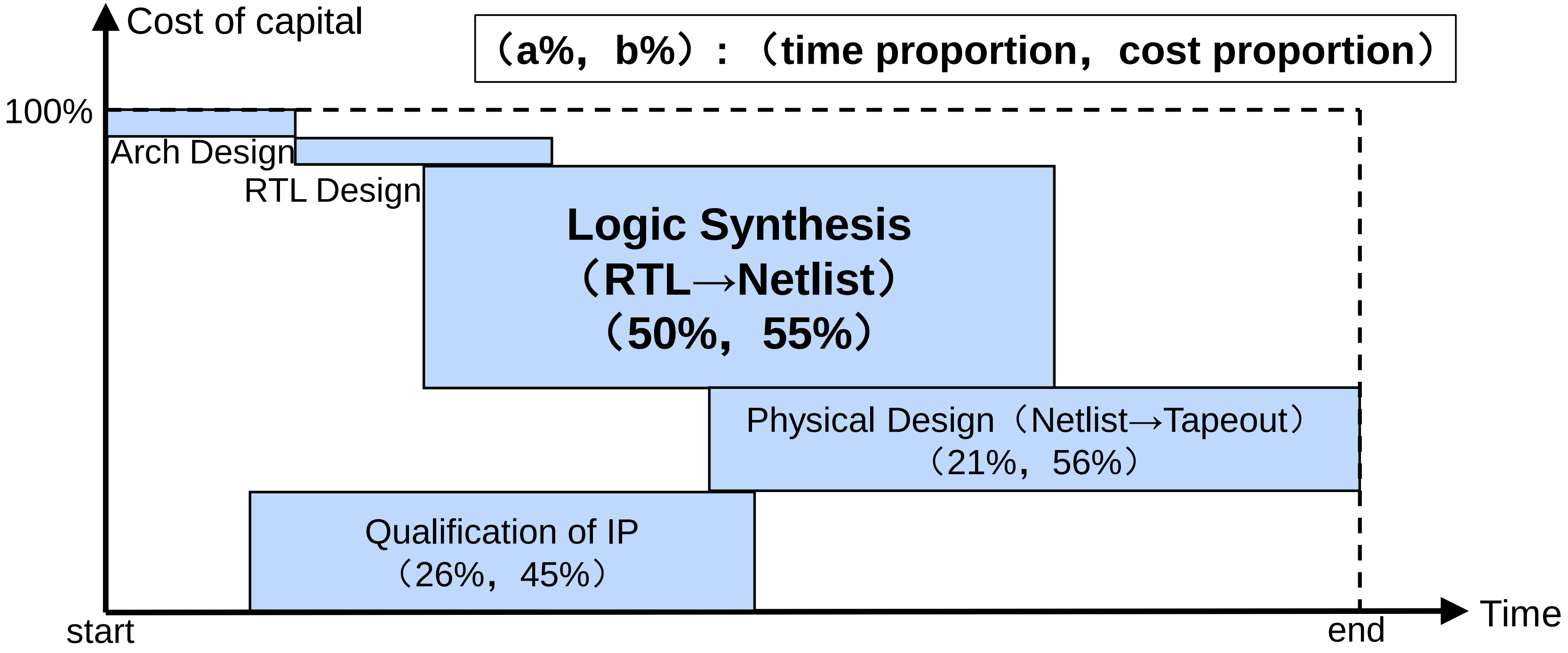
Placement & Routing

Verify functionality & manufacturability

Tapeout

Importance of logic synthesis

- Logic synthesis is **time-consuming (50%)** and has high **capital cost (55%)** in EDA process.



[1] https://eda360insider.wordpress.com/2012/02/27/system-eda-tools-attack-todays-great-bugaboo-for-soc-realization-the-software-development-overhang/

Logic Synthesis

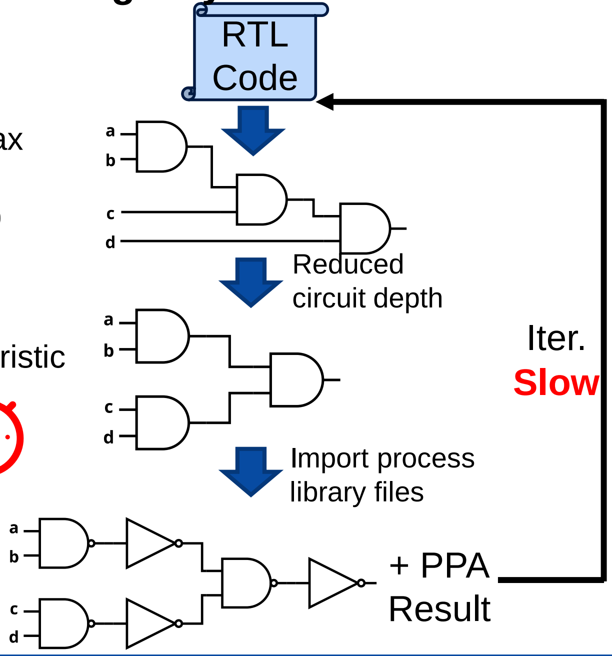
➢Logic synthesis is **iterative** in chip design. Predicting synthesis results can reduce iteration overhead.

Traditional logic synthesis flow

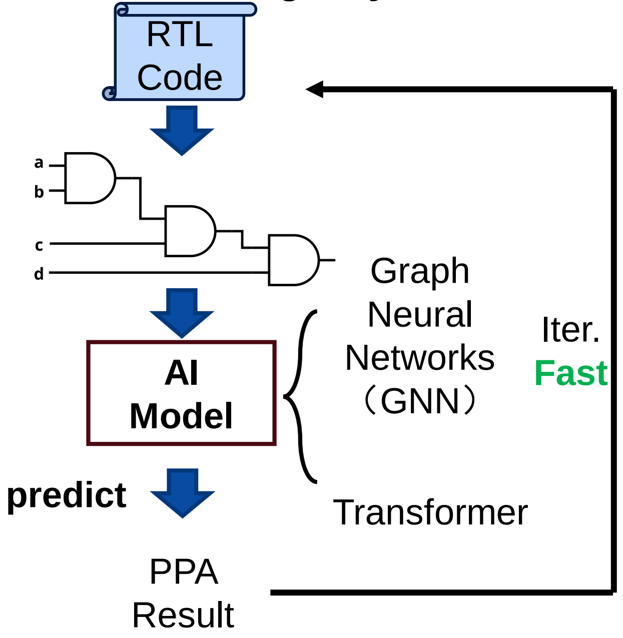
1. Translation
Fast, including syntax parsing, design checking, etc. (15%)

2. Logic optimization
Slow, extensive heuristic processes (50%)

3. Process mapping
Slow, further optimization (35%)



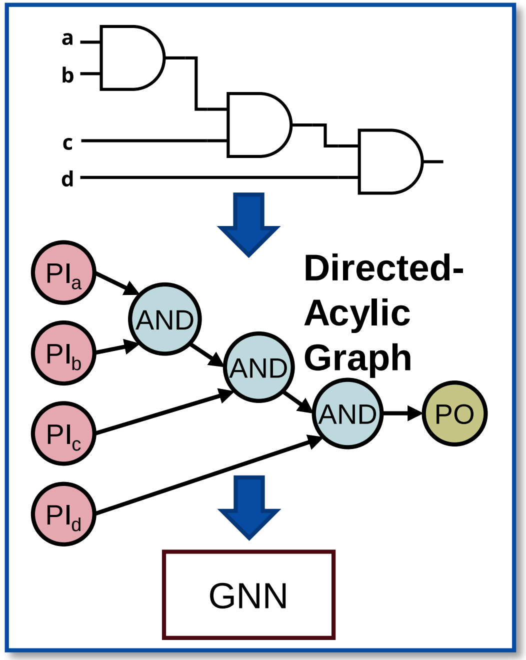
AI-assisted logic synthesis flow



Outline

GNN-based methods for Logic Synthesis

➢GNN model circuits as graphs and extract graph-level features for predicting PPA, but **face the inherent problems**

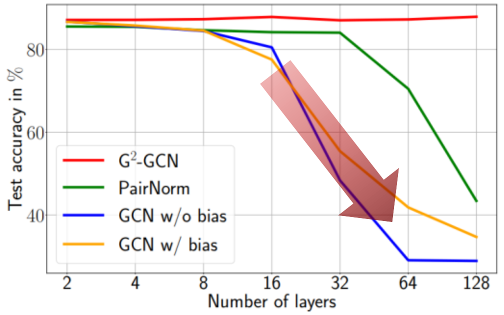


Over-smoothing[1]

GNN layers↑

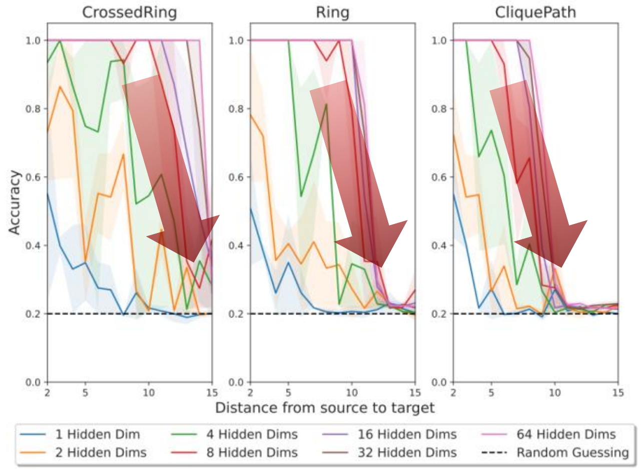
Node feature similarity↑

Accuracy↓



Over-squashing[2]

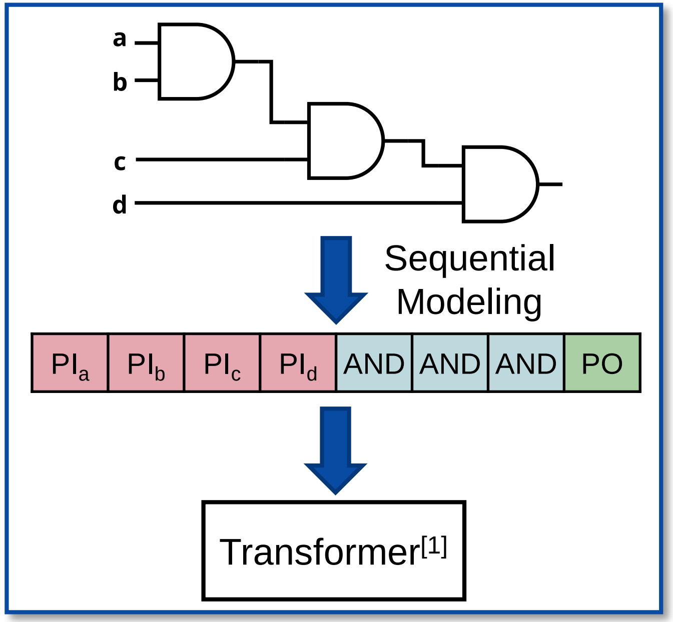
Long distance weak connection

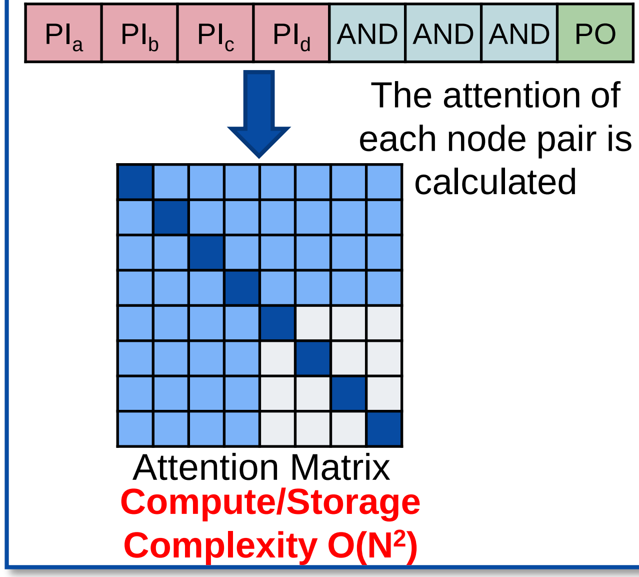


[1] Akansha S. Over-squashing in graph neural networks: A comprehensive survey[J]. arXiv preprint arXiv:2308.15568, 2023.
[2] Rusch T K, Bronstein M M, Mishra S. A survey on oversmoothing in graph neural networks[J]. arXiv preprint arXiv:2303.10993, 2023.

Transformer-based methods for Logic Synthesis

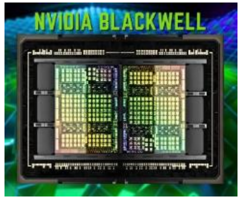
➢Transformer flats circuit to sequence, but faces **scalability problems and cannot be applied to large graphs**





PI_a PI_b PI_c PI_d AND AND AND PO
The attention of each node pair is calculated
Attention Matrix

Compute/Storage Complexity $O(N^2)$



NVIDIA B200[2], **$N = 2.08 \times 10^{12}$**

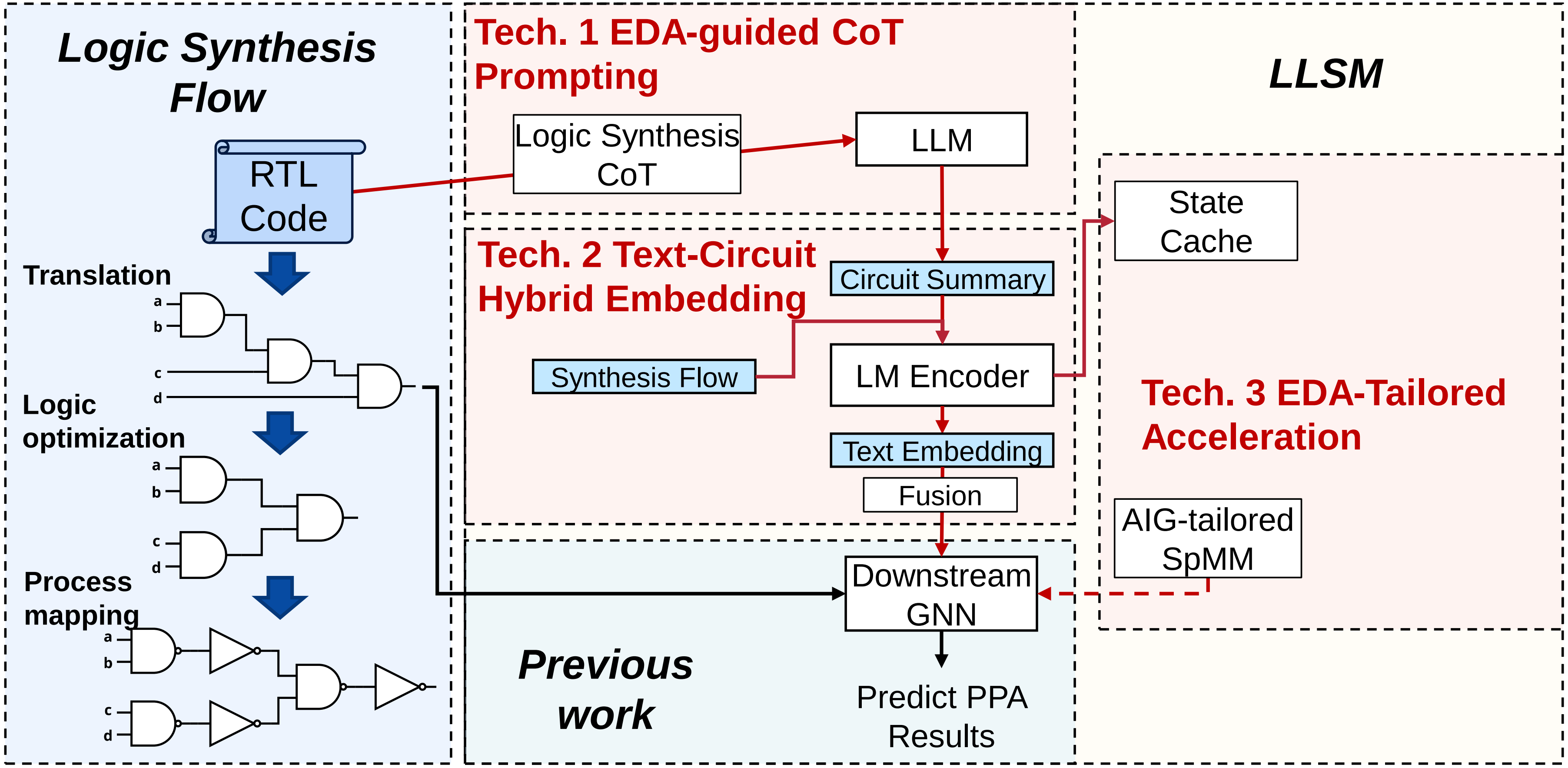
Apple M4[3], **$N = 2.8 \times 10^{11}$**

Datasets used in academia also have circuits with $N > 10^5$

[1] Xu, Ceyu, Chris Kjellqvist, and Lisa Wu Wills. "SNS's not a synthesizer: a deep-learning-based synthesis predictor." Proceedings of the 49th Annual International Symposium on Computer Architecture. 2022.
[2] https://www.nvidia.cn/data-center/technologies/blackwell-architecture/
[3] https://www.apple.com.cn/newsroom/2024/05/apple-introduces-m4-chip/

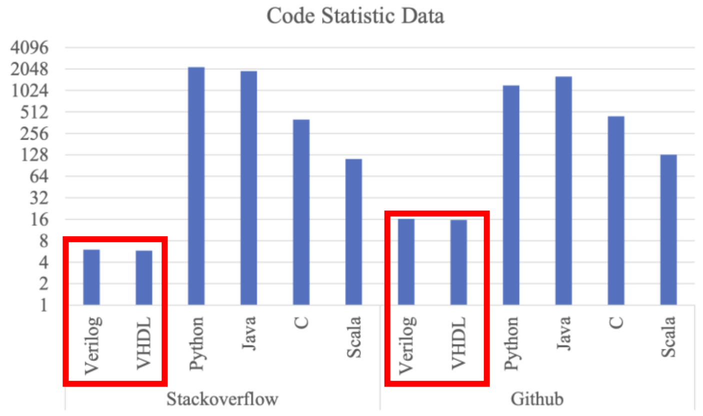
Outline

Overview

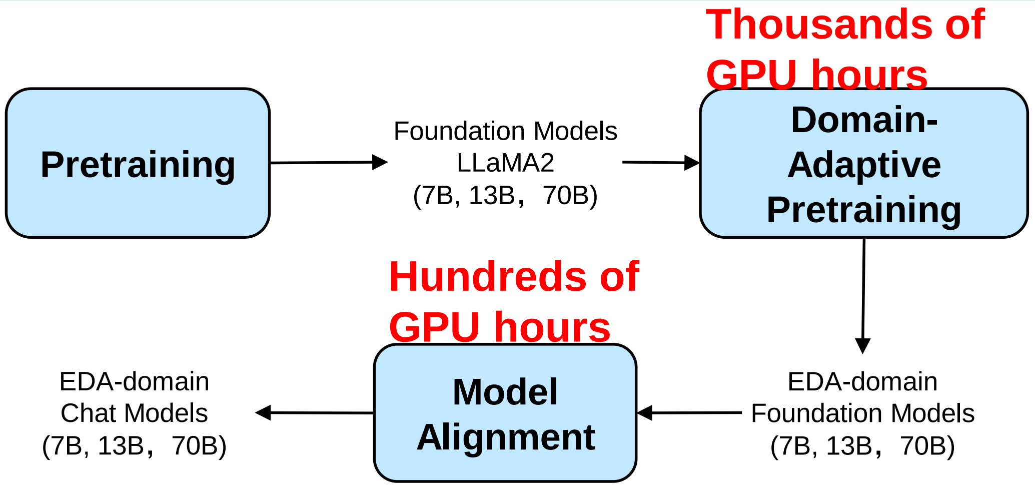

Logic Synthesis Flow
RTL Code
Translation
a
b
c
d
Logic optimization
a
b
c
d
Process mapping
a
b
c
d
Tech. 1 EDA-guided CoT Prompting
Logic Synthesis CoT
LLM
LLSM
Tech. 2 Text-Circuit Hybrid Embedding
Circuit Summary
Synthesis Flow
LM Encoder
Text Embedding
Fusion
State Cache
Tech. 3 EDA-Tailored Acceleration
AIG-tailored SpMM
Downstream GNN
Previous work
Predict PPA Results

Technique 1: EDA-guided CoT Prompting

Challenge **LLMs lack the knowledge to analyze RTL code, and it's expensive to train or fine-tune**



Lack of RTL Code Data[1]

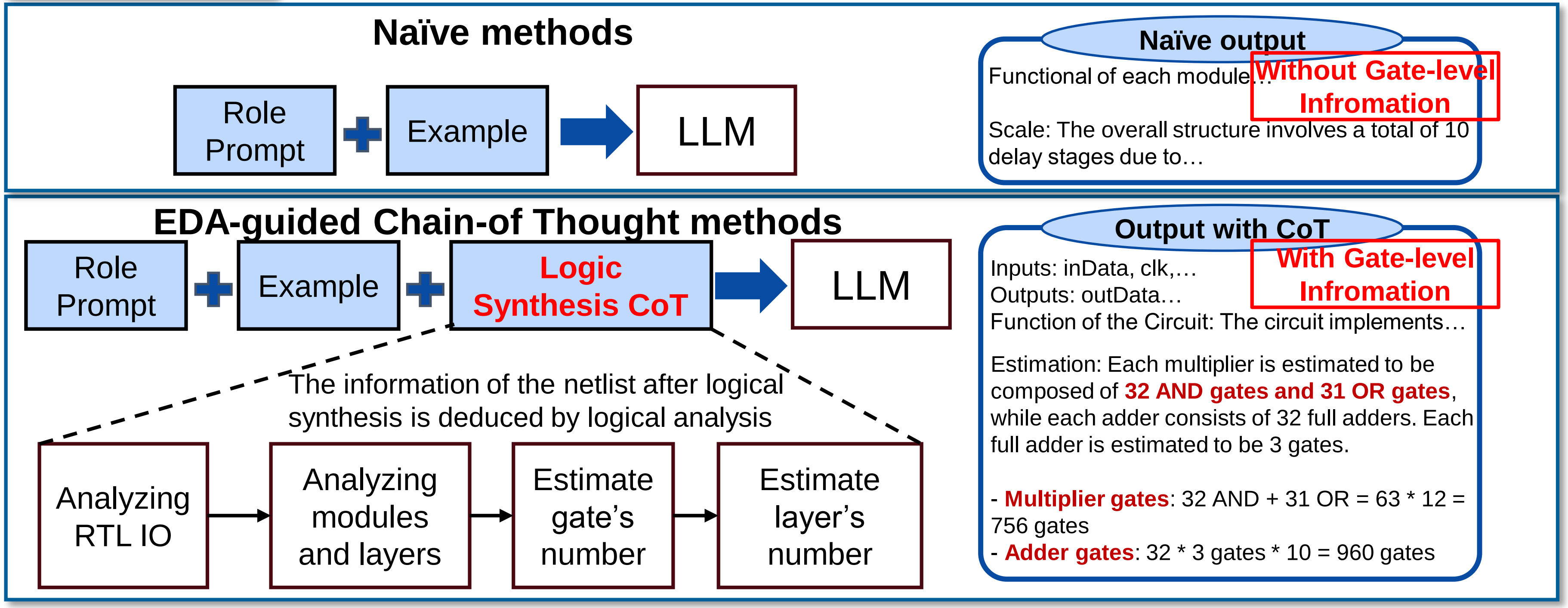

Thousands of GPU hours
Pretraining
Foundation Models LLaMA2 (7B, 13B，70B)
Domain-Adaptive Pretraining
Hundreds of GPU hours
EDA-domain Foundation Models (7B, 13B，70B)
Model Alignment
EDA-domain Chat Models (7B, 13B，70B)

High Training Cost[2]

[1] Chang, Kaiyan, et al. "Data is all you need: Finetuning llms for chip design via an automated design-data augmentation framework." Proceedings of the 61st ACM/IEEE Design Automation Conference. 2024.
[2] Liu, Mingjie, et al. "Chipnemo: Domain-adapted llms for chip design." arXiv preprint arXiv:2311.00176 (2023).

Technique 1: EDA-guided CoT Prompting

Approach **Training-free CoT method to guide LLM summarize size and gate-level information of RTL Code.**



Naïve methods
Role Prompt
Example
LLM
EDA-guided Chain-of Thought methods
Role Prompt
Example
Logic Synthesis CoT
LLM
The information of the netlist after logical synthesis is deduced by logical analysis
Analyzing RTL IO
Analyzing modules and layers
Estimate gate's number
Estimate layer's number

Naïve output

Functional of each module…

Without Gate-level Infromation

Scale: The overall structure involves a total of 10 delay stages due to…

Output with CoT

With Gate-level Infromation

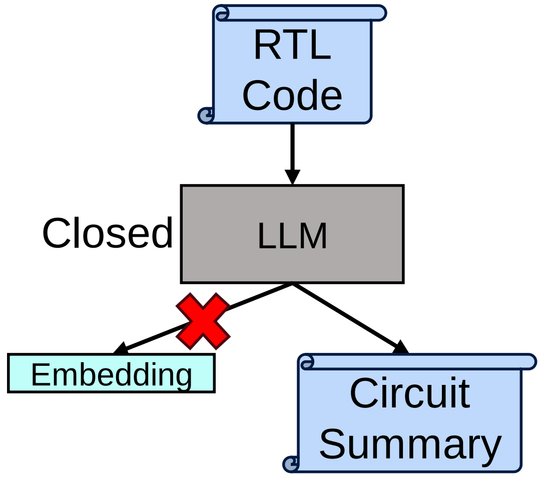
Inputs: inData, clk,…
Outputs: outData…
Function of the Circuit: The circuit implements…

Estimation: Each multiplier is estimated to be composed of **32 AND gates and 31 OR gates**, while each adder consists of 32 full adders. Each full adder is estimated to be 3 gates.

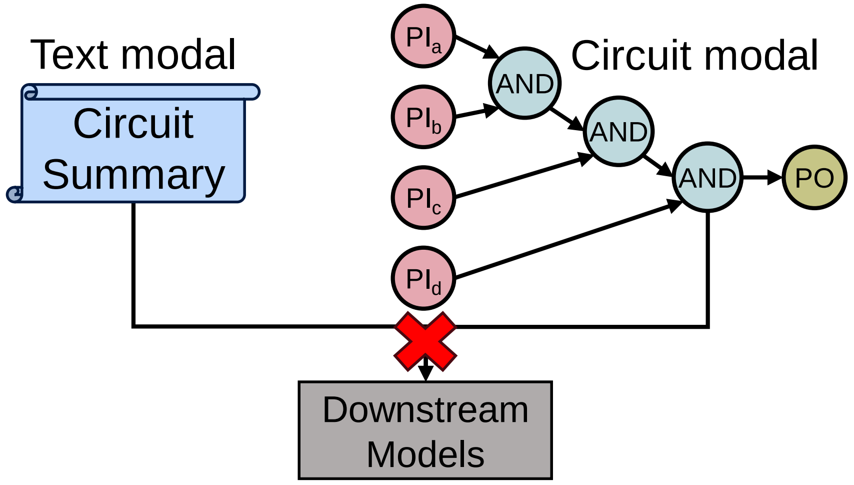
- **Multiplier gates**: 32 AND + 31 OR = 63 * 12 = 756 gates
- **Adder gates**: 32 * 3 gates * 10 = 960 gates

Technique 2: Text-Circuit Hybrid Embedding

Challenge **Closed LLM results in the inability to extract feature embeddings and circuit summary cannot be directly input into downstream models**



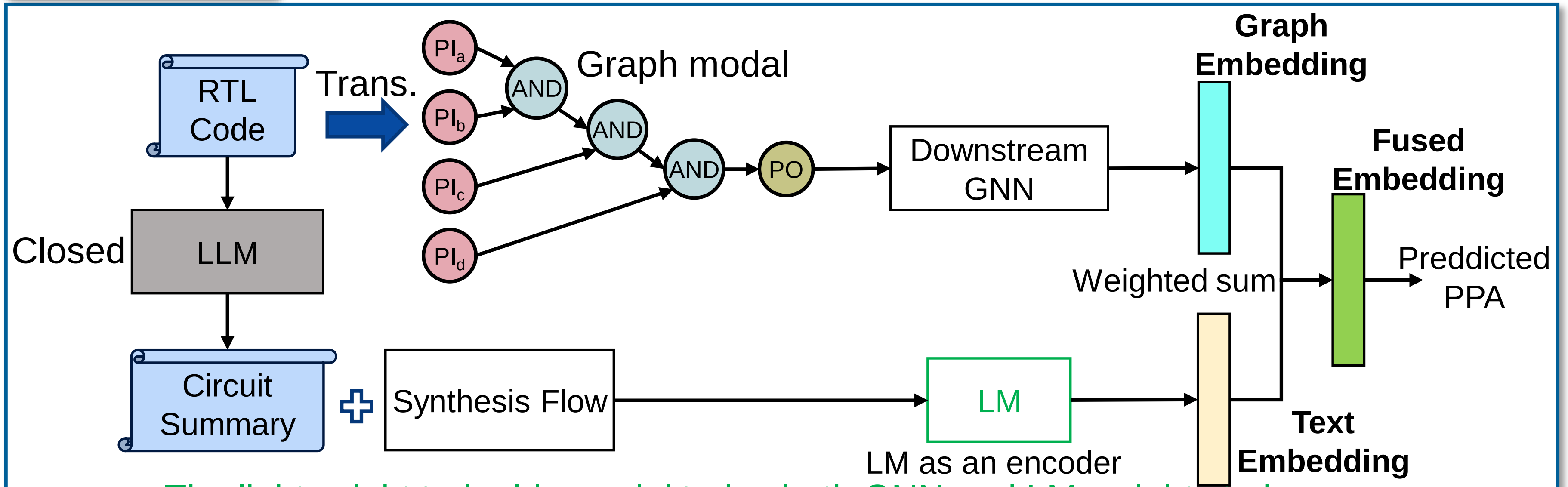
Embeddings cannot be obtained from closed-source LLMs



The circuit summary & graph cannot be the downstream model input at the same time

Technique 2: Text-Circuit Hybrid Embedding

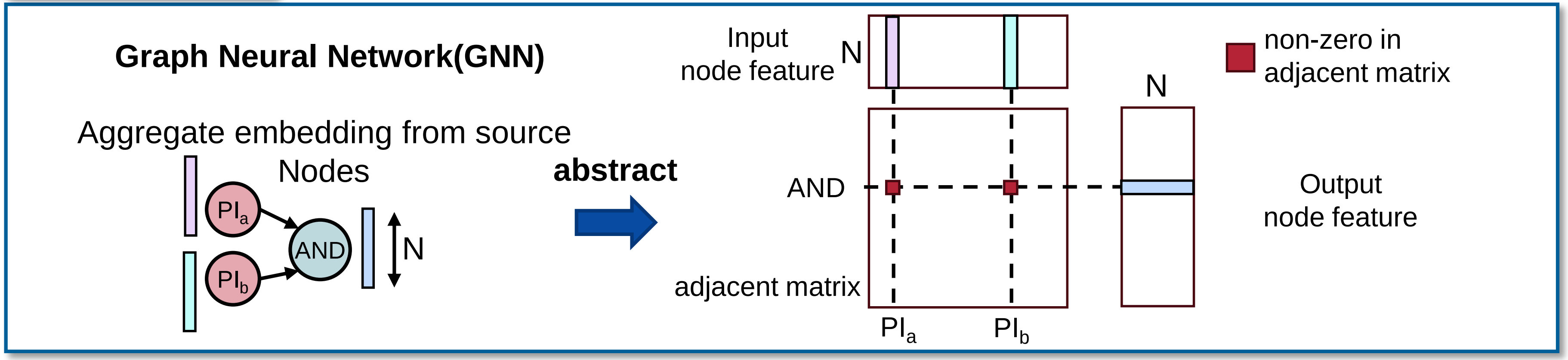
Approach **Using a small Language Model as a text encoder to generate text embedding**

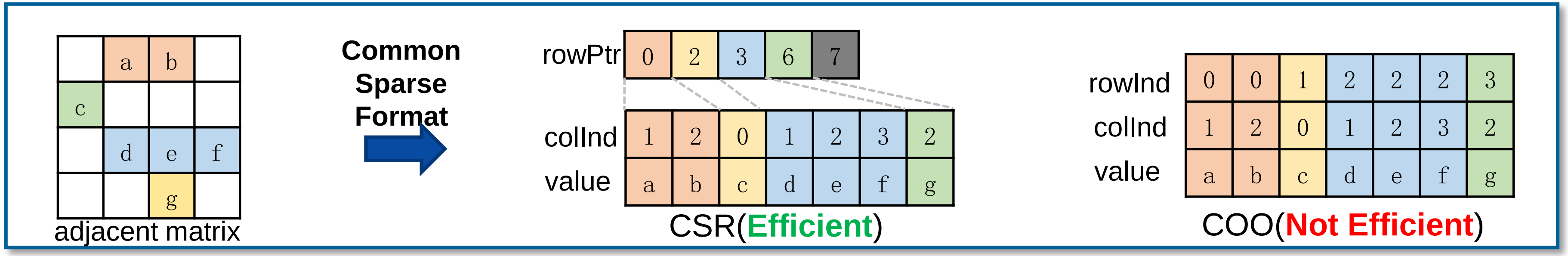


The lightweight trainable model trains both GNN and LM weights to improve prediction accuracy

Technique 3: EDA-Tailored Acceleration

Background **The bottleneck of GNN is message propagation on edges, which can be abstracted as SpMM operator**





	a	b	
c			
	d	e	f
		g	

adjacent matrix

rowPtr	0	2	3	6	7		
colInd	1	2	0	1	2	3	2
value	a	b	c	d	e	f	g

CSR(Efficient)

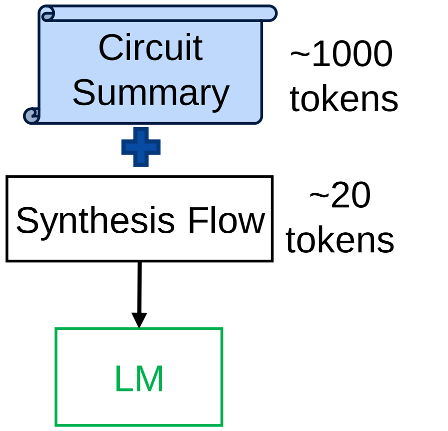
rowInd	0	0	1	2	2	2	3
colInd	1	2	0	1	2	3	2
value	a	b	c	d	e	f	g

COO(Not Efficient)

Technique 3: EDA-Tailored Acceleration

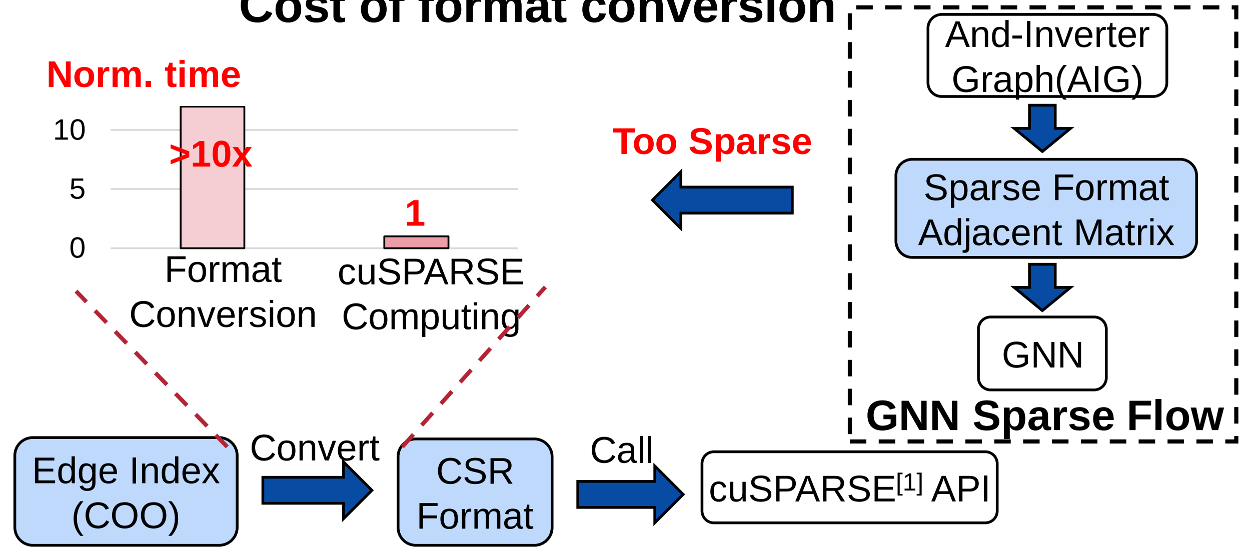
Challenge **The introduction of LM result in slower inference and More sparse circuit graphs**

Cost of LM



Two orders of magnitude slower than GNN

Cost of format conversion



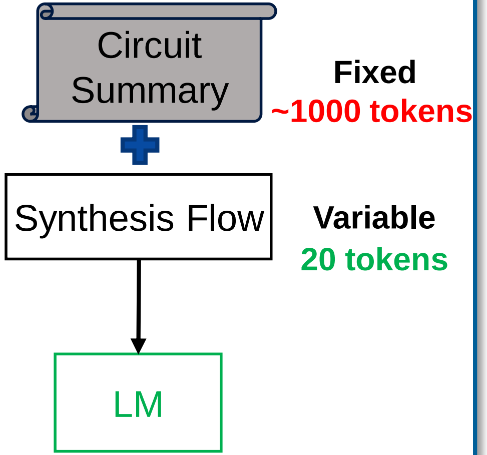
Time-consuming format conversion from COO to CSR

[1] NVIDIA sparse computing library, https://docs.nvidia.com/cuda/cusparse/index.html

Technique 3: EDA-Tailored Acceleration

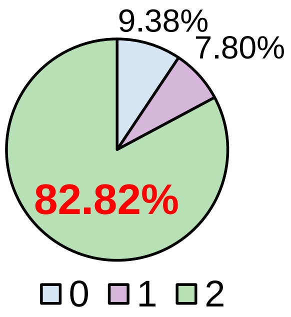
Insight **Different input changes are the logic synthesis flow, and AIG has structural features**

Redundant computing of Circuit Summary

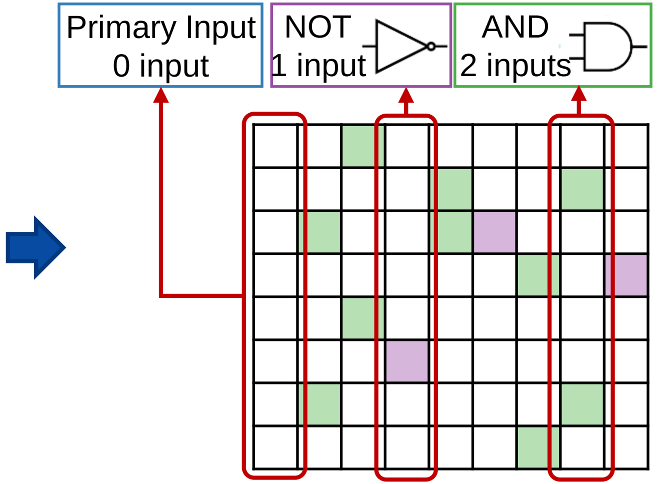


AIG's structural features

In-degree=2 is dominant



AIG In-degree Percentage

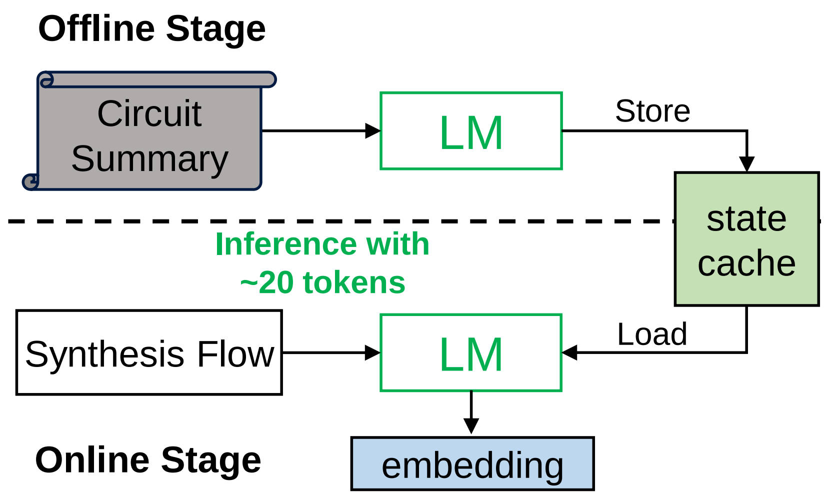


AIG's Adjacent Matrix

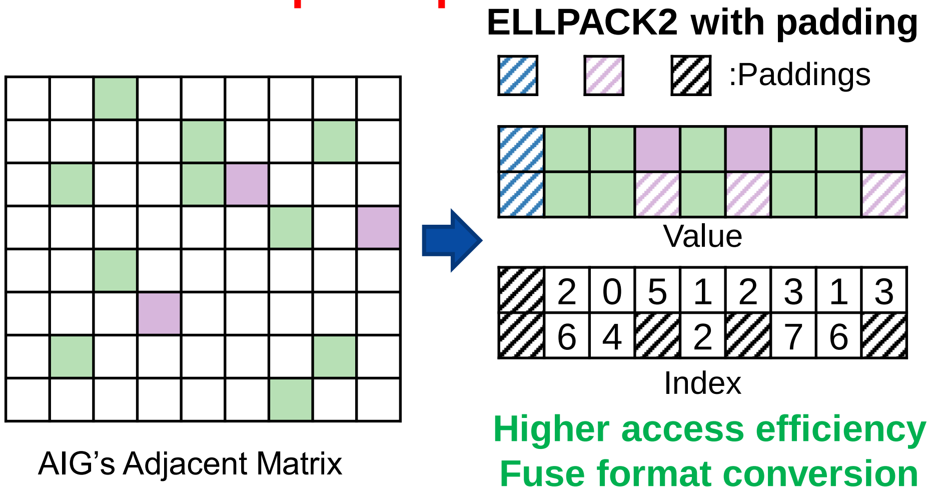
Technique 3: EDA-Tailored Acceleration

Approach **Using ELLPACK2 for efficient memory access and fuse conversion and computing on GPU**

Store state to cache



Leverage AIG structural features to improve parallelism

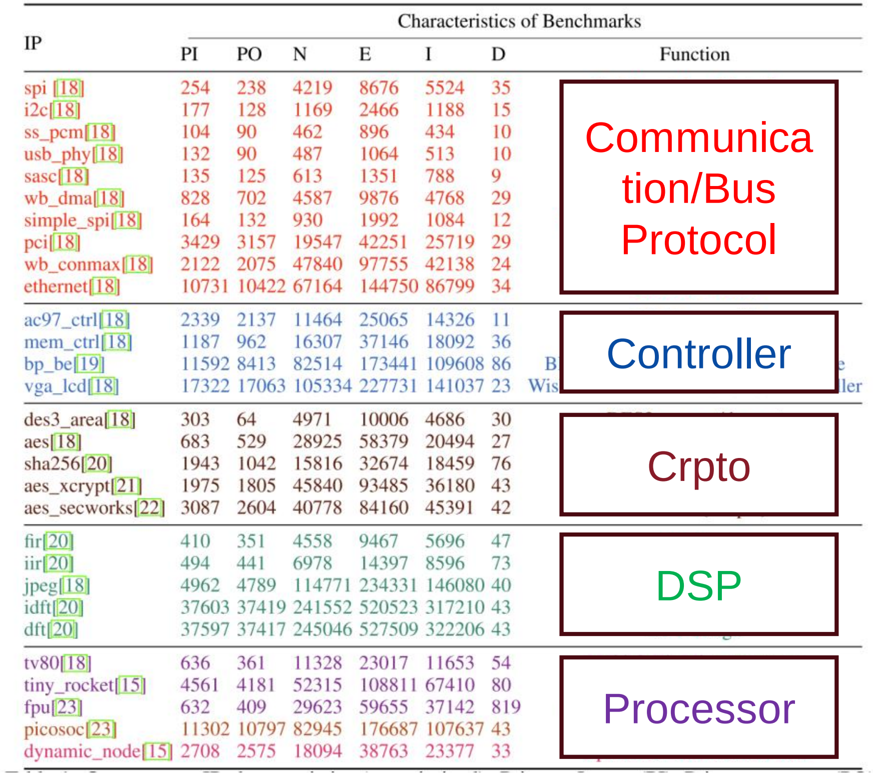


Higher access efficiency
Fuse format conversion and computation

Outline

Experiment Setup

- Setup
  - GPU: A100, nvcc 11.8, Pytorch 2.0.1, PyG v2.5.3
  - Dataset: OpenABC， 23 IP， 1500 logic synthesis flow
  - Baseline:
    - OpenABC
    - LOSTIN
  - LM-model
    - Mamba-130m
    - DeBERTa-base
  - Training
    - 20 epochs
    - Learning rate(0.1 for LM, 0.01 for GNN)



IP	Characteristics of Benchmarks						
	PI	PO	N	E	I	D	Function
spi [18]	254	238	4219	8676	5524	35	Communication/Bus Protocol
i2c[18]	177	128	1169	2466	1188	15	
ss_pcm[18]	104	90	462	896	434	10	
usb_phy[18]	132	90	487	1064	513	10	
sasc[18]	135	125	613	1351	788	9	
wb_dma[18]	828	702	4587	9876	4768	29	
simple_spi[18]	164	132	930	1992	1084	12	
pci[18]	3429	3157	19547	42251	25719	29	
wb_conmax[18]	2122	2075	47840	97755	42138	24	
ethernet[18]	10731	10422	67164	144750	86799	34	
ac97_ctrl[18]	2339	2137	11464	25065	14326	11	Controller
mem_ctrl[18]	1187	962	16307	37146	18092	36	
bp_be[19]	11592	8413	82514	173441	109608	86	
vga_lcd[18]	17322	17063	105334	227731	141037	23	
des3_area[18]	303	64	4971	10006	4686	30	Crpto
aes[18]	683	529	28925	58379	20494	27	
sha256[20]	1943	1042	15816	32674	18459	76	
aes_xcrypt[21]	1975	1805	45840	93485	36180	43	
aes_secworks[22]	3087	2604	40778	84160	45391	42	
fir[20]	410	351	4558	9467	5696	47	DSP
iir[20]	494	441	6978	14397	8596	73	
jpeg[18]	4962	4789	114771	234331	146080	40	
idft[20]	37603	37419	241552	520523	317210	43	
dft[20]	37597	37417	245046	527509	322206	43	
tv80[18]	636	361	11328	23017	11653	54	Processor
tiny_rocket[15]	4561	4181	52315	108811	67410	80	
fpu[23]	632	409	29623	59655	37142	819	
picosoc[23]	11302	10797	82945	176687	107637	43	
dynamic_node[15]	2708	2575	18094	38763	23377	33	

[1] Chowdhury A B, Tan B, Karri R, et al. Openabc-d: A large-scale dataset for machine learning guided integrated circuit synthesis[J]. arXiv preprint arXiv:2110.11292, 2021.

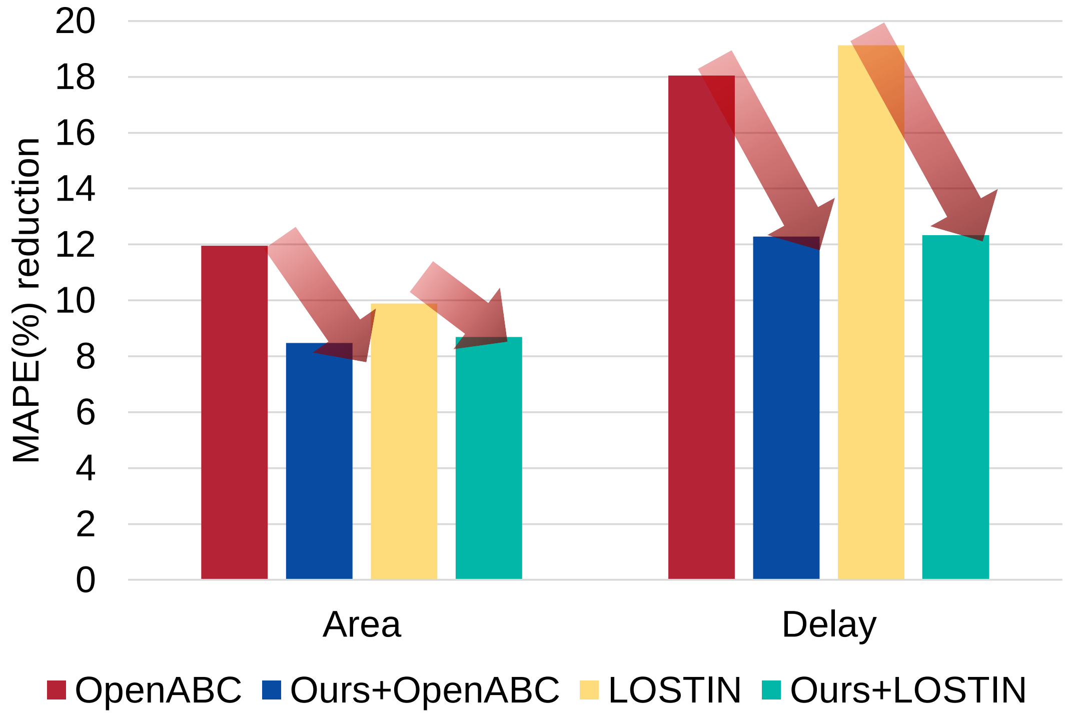
Evaluation Result

Area prediction

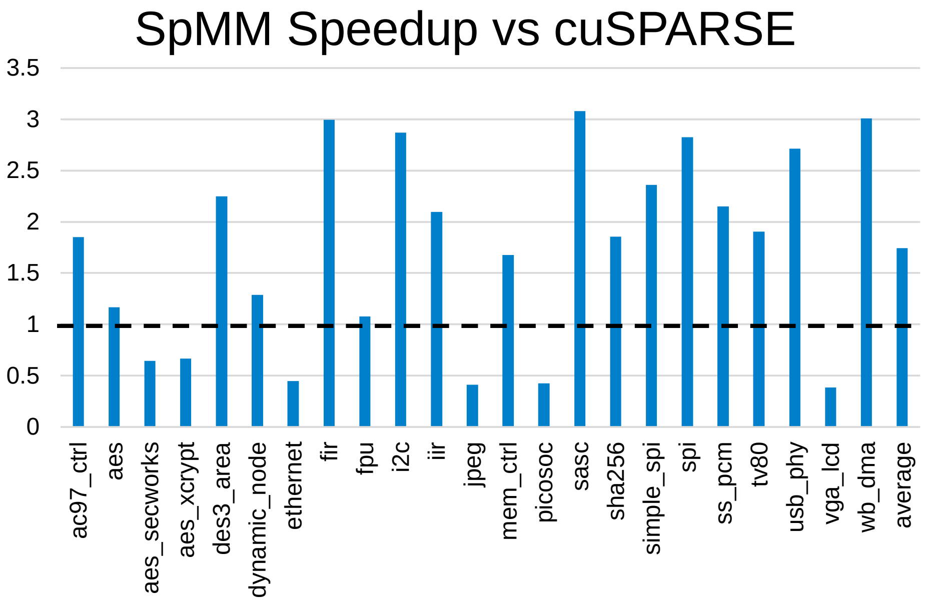
3.49% and 1.19% average MAPE reduction

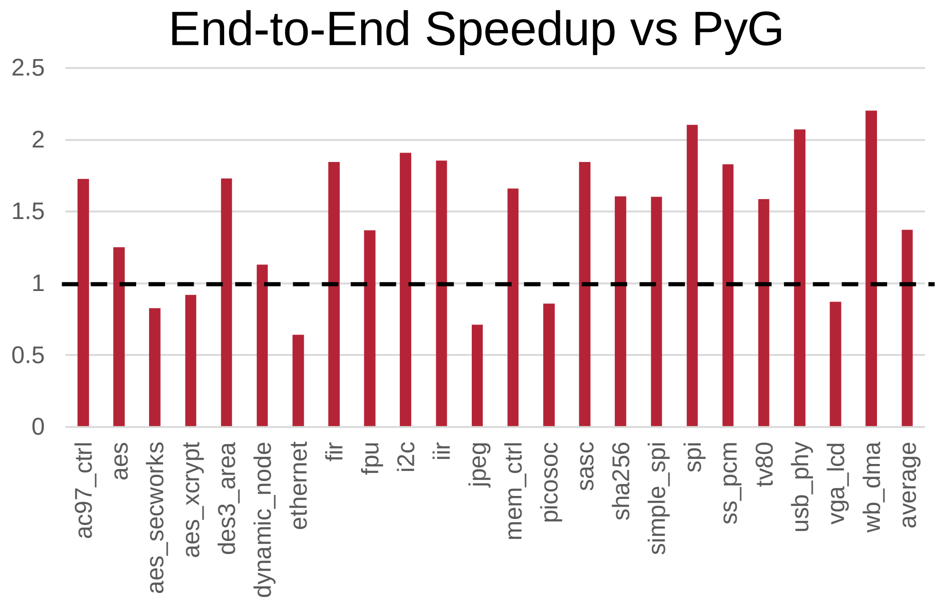
Delay prediction

5.76% and 6.80% average MAPE reduction



Speedup Result



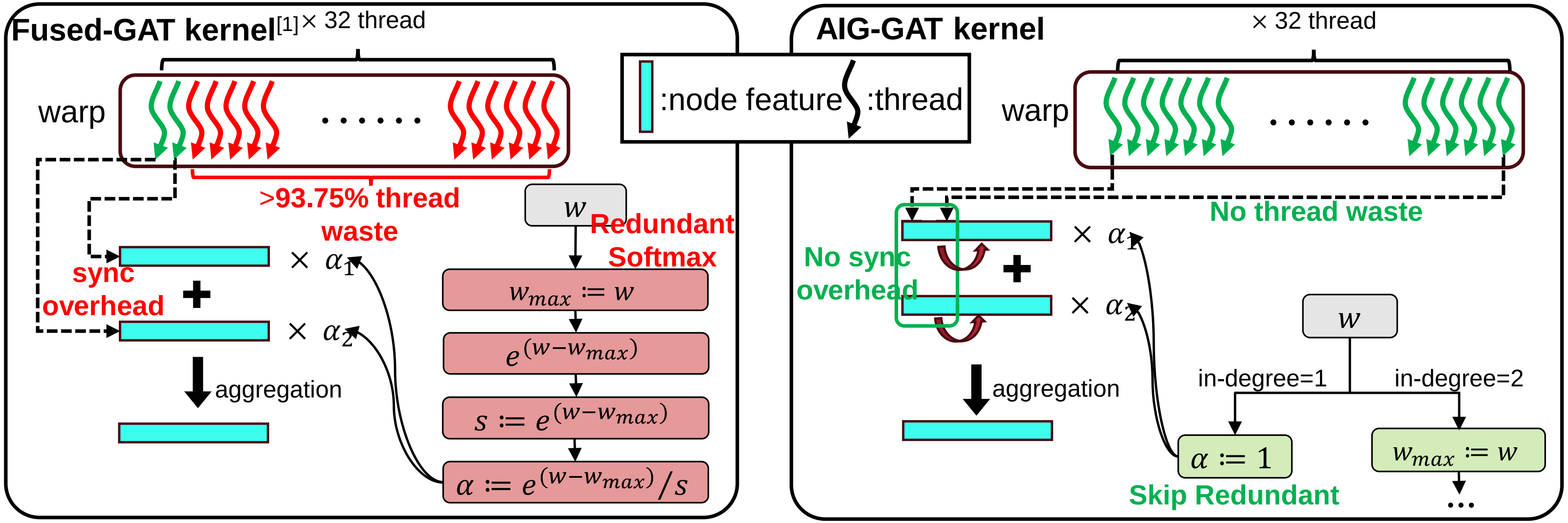


AIG-Tailored SpMM kernel achieves an average of **1.74×** speedup compared with cuSPARSE
An average of end-to-end **1.37×** speedup compared with PyG

Outline

Extension: AIG-based GAT acceleration

➢Thread workload reallocation and skip redundant computing

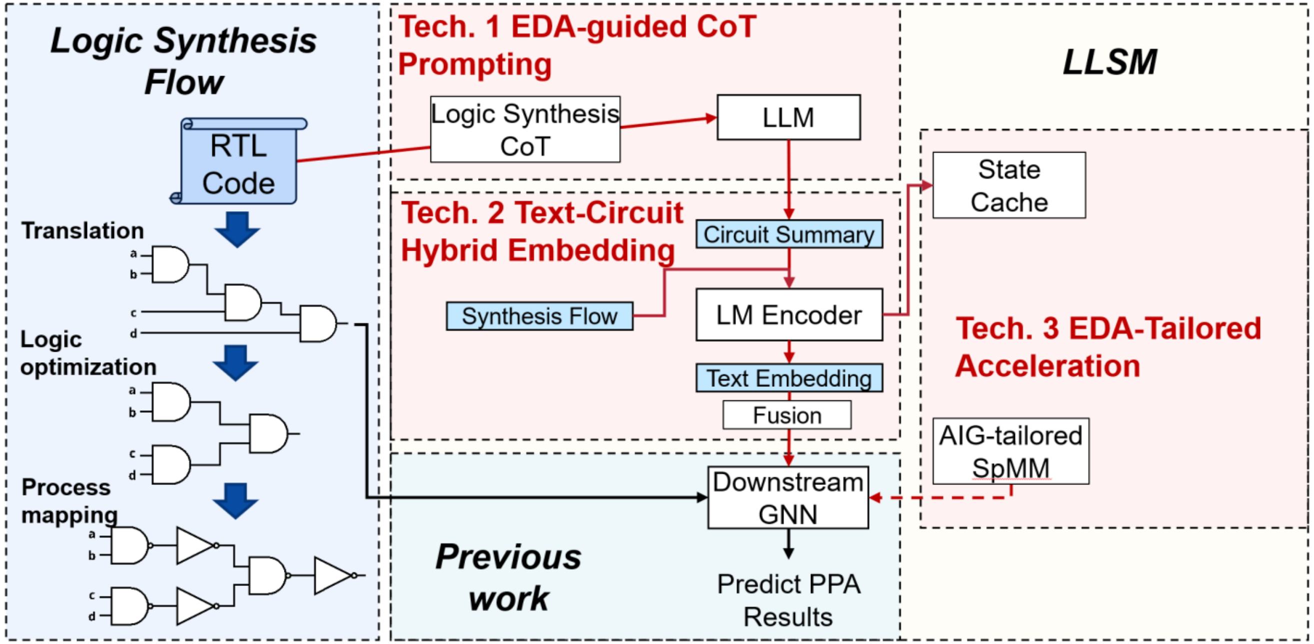


1.54x average speedup and 46.8% memory usage reduction over PyG

[1] Zhang, Hengrui, et al. "Understanding gnn computational graph: A coordinated computation, io, and memory perspective." Proceedings of Machine Learning and Systems 4 (2022): 467-484.

LLSM: LLM-enhanced Logic Synthesis Model with EDA-guided CoT Prompting, Hybrid Embedding and AIG-tailored Acceleration

Shan Huang, supervised by Prof. Guohao Dai
ironheart@sjtu.edu.cn, daiguohao@sjtu.edu.cn


Logic Synthesis Flow
RTL Code
Translation
a
b
c
d
Logic optimization
a
b
c
d
Process mapping
a
b
c
d
Tech. 1 EDA-guided CoT Prompting
Logic Synthesis CoT
LLM
Tech. 2 Text-Circuit Hybrid Embedding
Circuit Summary
Synthesis Flow
LM Encoder
Text Embedding
Fusion
Downstream GNN
Previous work
Predict PPA Results
LLSM
State Cache
Tech. 3 EDA-Tailored Acceleration
AIG-tailored SpMM