Industrial General AMP Sizing Optimization via Effective Multi-Stage LLM Reasoning Framework

NUDT-LLM4Circuit

Huiqing You, Zhien Li, Bin Guo, Xiaoshu Chen, Junbo Zhang

Zhenyu Zhao, Deng Luo

National University of Defense Technology

***Abstract*—In response to the persistent efficiency bottleneck in industrial analog circuit design, where amplifier (AMP) sizing optimization primarily depends on manual expertise and iterative simulation, the IEEE AICAS 2025 Grand Challenge has initiated a Large Language Model (LLM)-based automated amplifier design competition to develop efficient and generalizable solutions. To address this challenge, this paper presents an Effective Multi-Stage LLM Reasoning Framework (EM-SLRF). The proposed methodology introduces a multi-stage progressive optimization framework incorporating LLM chain-of-thought (CoT) reasoning, which reduces required simulation iterations through quantized LLM-driven dynamic optimization feedback, enabling rapid generation of high-performance netlists that meet industrial specifications. Furthermore, we devise a knowledge empowerment method that efficiently converts domain expertise into high-quality LLM training datasets, facilitating cross-process and cross-structure AMP sizing optimization. Comprehensive experimental evaluations of the Qwen2.5-LLM-based EM-SLRF demonstrate three critical advantages: 1) Achieving state-of-the-art figure-of-merit (FoM) performance metrics under stringent industrial constraints of low power consumption and high gain, 2) Realizing 20× acceleration compared to manual approaches while demonstrating superior efficiency over existing methods, and 3) Demonstrating consistent effectiveness in AMP sizing optimization across diverse process technologies and circuit structures.**

***Index Terms*—industrial, general, amplifier sizing optimization, large language model**

I. Introduction

Analog circuit design plays a crucial role in modern signal processing systems and mixed-signal integration, yet confronts inherent complexities arising from multi-objective performance trade-offs and process variation sensitivity[3, 12, 14]. The IEEE AICAS 2025 Grand Challenge has formulated a Qwen2.5-based automated amplifier (AMP) design task with four critical constraints: no direct process parameter input, multi-corner verification, 7B model limitation, and conventional algorithm exclusion[2].

Existing methodologies exhibit significant limitations: conventional weighted-sum approaches inadequately characterize high-dimensional design spaces, whereas evolutionary algorithms encounter combinatorial explosion challenges[7, 10, 11]. Large Language Model (LLM[1])-based solutions such as AMPagent[8] and LADAC[9] show promising progress but suffer from suboptimal knowledge utilization and computationally intensive iteration cycles. Although the Atelier framework[12] improves automation via multi-agent collaboration, it exhibits inherent efficiency bottlenecks in LLM coordination mechanisms.

This work presents an Effective Multi-Stage LLM Reasoning Framework (EM-SLRF) with three key technical contributions:

- Developed a multi-stage optimization architecture employing LLM chain-of-thought (CoT) reasoning, effectively reducing SPICE simulation iterations by 94.1% while enhancing netlist generation efficiency.
- Established a rapid knowledge empowerment technique that encodes domain expertise into compact optimization guides, enabling fast AMP sizing across various architectures and process nodes.
- Experimental results on Qwen2.5 framework show a peak figure-of-merit (FoM) of 598, achieving 20× acceleration over manual approaches. The framework demonstrates <5% FoM variation across Sky130/SMIC055NLL processes and two structures, proving industrial viability.

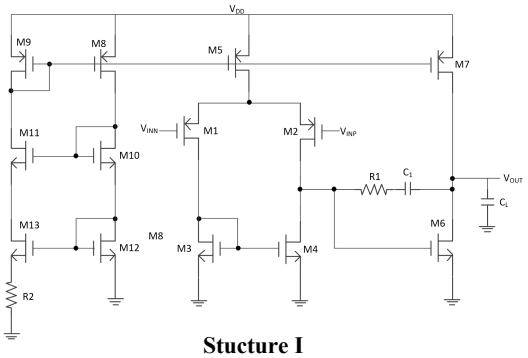
II. Background

A. Expert Knowledge

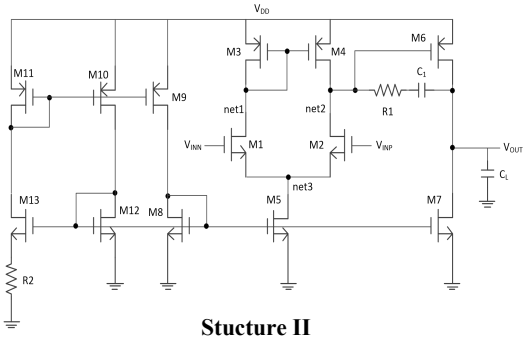
1) AMP Theory: The AMP is a high-gain, differential-input, single-ended output circuit that amplifies small input variations and is widely applied in analog signal processing. This design adopts a typical two-stage AMP (Fig.1, Structure I), with the first stage offering high gain and the second acting as an output driver.

The input stage comprises M1−M5: M1 and M2 form an NMOS differential pair for common-mode rejection; M3 and M4 serve as active loads via current mirrors; M5 provides bias current. The output stage includes M6 as a common-source amplifier and M7 for constant bias and load. Miller compensation is achieved by R1 and Cc to ensure stability.

To enhance PSRR, a self-biased current source independent of supply voltage is employed. The bias current depends only



Stucture I



Stucture II

Fig. 1. AMP Structures.

on the size ratio of M13 to M12 and resistor R2, allowing flexible control via proper sizing.

2) Sizing Strategy: For the AMP Structure I shown in Fig.1, there is a trade-off between the size of each device and its performance indicators, as shown in Table I:

TABLE I
SIZING STRATEGY

Metric	Adjustment Strategy and Impact on Other Metrics
Gain	Increasing the bias current or the width-to-length ratio (W/L) of input transistors to enhance Gain, but degrades power efficiency, increases area overhead, and reduces phase margin.
GBW	Reducing transistor dimensions to improve GBW, but increases input-referred noise and degrades device matching.
Noise	Enlarging input transistors reduces noise and improves matching, but degrades GBW and SR.
SR	Scaling up output-stage transistors to increase SR, by providing higher drive current, but elevates power consumption.
PSRR	Larger input transistors reduce noise and improve matching but degrade GBW and SR.
Power	Lowering bias currents to reduce power dissipation, but degrades SR and GBW.

B. Chain of Thought

The CoT, introduced by Wei et al.[13], enhances LLM reasoning and interpretability. Initially applicable only to models with over 100B parameters, its scope is limited. To extend CoT to smaller models, CoT-Finetuning is proposed, incorporating reasoning steps into training supervision. Unlike standard datasets with only inputs and answers, CoT-Finetuning adds intermediate reasoning, either manually annotated[15] or generated by larger LLMs[5].

III. METHODOLOGY

To overcome limitations in existing AMP sizing approaches, this paper proposes the EM-SLR, as illustrated in Fig.2. Given performance requirements, the Planning Agent coordinates global task allocation. As shown in Fig.2 ①, the sizing task is divided into a three-stage progressive optimization. Each stage begins with netlist initialization (Fig.2 ②), followed by iterative optimization (Fig.2 ③). Concurrently, expert knowledge is rapidly injected into the LLM for cross-structure and cross-process generalization (Fig.2 ④). Detailed implementations of the sizing and knowledge empowerment methods are presented in Sections III-A and III-B, respectively.

A. AMP Sizing Optimization

1) Multi-Agent: To achieve fast and efficient AMP sizing optimization, we construct a multi-agent architecture comprising a Planning Agent, Initiation Agent, Resizing Agent, Simulation (Sim.) Agent, and Optimization (Opt.) Agent. This framework decomposes the complex AMP sizing process. The detailed description of each agent is presented in Table II.

TABLE II
AGENT INTRODUCTION

Agent	Introduce
Planning	Receive performance requirements, allocate tasks based on LLM feedback, and invoke agents for optimization.
Initiation	Collaborate with LLM using a prompt with process and performance specs to generate code, then verify and execute it to obtain robust initialization parameters.
Resizing	Post-process the LLM's response to extract netlist adjustment strategies, using flexible percentage and fixed step size optimization methods while updating state flags in real time.
Sim.	Collaborates with Ngspice to dynamically update netlist parameters, triggers real-time simulations based on strategy and status flags, and automatically retrieves simulation results.
Opt.	Based on simulation results and state flags, dynamic prompts are generated to guide the LLM in producing both optimization operations and updated state flags for netlist optimization.

2) Multi-Stage: This paper proposes a three-stage progressive AMP sizing methodology leveraging the dynamic decision feedback capability of expert-knowledge-empowered LLM CoT reasoning:

Stage I: Circuit Optimizing with Ideal Current. Using an ideal current source, the initialization Agent and LLM co-generate netlists based on process parameters. Ngspice simulations extract key metrics with tt-corner-only evaluation. Post-training LLM feedback guides iterative optimization until specifications are met.

Stage II: Circuit Optimizing with Bias Current. A biasing circuit replaces the ideal current source. The Agent-LLM system inherits Stage I results to generate bias-incorporated

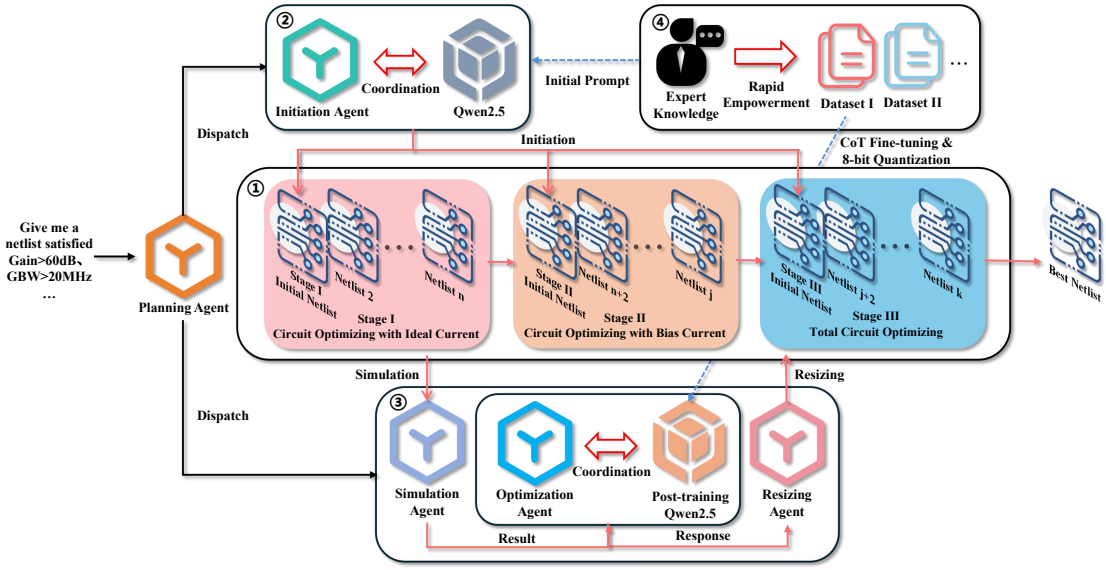


Fig. 2. The Fromwork of EM-SLRF

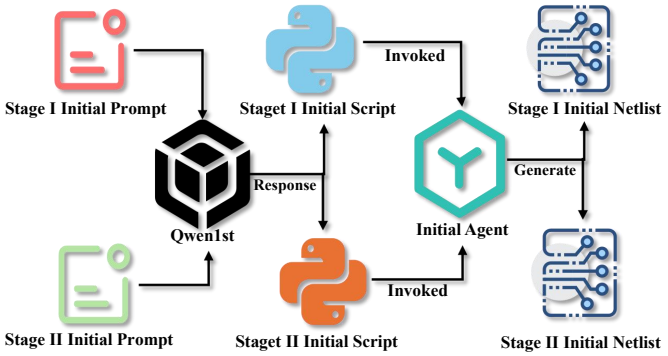


Fig. 3. Initiation Process

netlists. Target current optimization requires only tt-corner current simulations, enabling rapid convergence.

Stage III: Total Circuit Optimizing. Holistic optimization integrates prior results with full-corner verification. The co-optimization framework ensures maximizing industrial FoM through corner-aware performance tuning.

Performance verification at each stage enables efficient retargeting for custom specifications through requirement parameter modification.

3) Initiation Cooperated with LLM: To achieve robust automatic netlist initialization, we leverage LLM's dual capabilities in code generation and natural language processing. The framework processes design specifications through structured prompt engineering to generate executable initialization code. An Initiation Agent subsequently validates and executes the code in sandboxed environments, ensuring parameter accuracy and system stability as illustrated in Fig.3.

4) Iterative Optimization: The iterative optimization framework based on multi-agent collaboration with post-training LLM constitutes a core component of EM-SLRF. This three-stage process operates cyclically: (1) The Simulation Agent conducts Ngspice simulations using current state markers; (2) The Optimization Agent generates collaborative prompts for the LLM to derive optimization strategies; (3) The Resizing Agent implements parameter adjustments and updates state markers, completing the iteration cycle.

B. Rapid Expert Knowledge Empowerment

Experimental evaluation reveals insufficient circuit domain knowledge in Qwen2.5 models during pre-training, particularly in interpreting simulation metrics. To address this, we develop a pipeline for injecting process-architecture-specific knowledge into LLMs. As illustrated in Fig.4, our framework: (1) Condenses expert optimization strategies (Table I) into simulation iterations, (2) Dynamically analyzes simulation results and current states as training inputs, and (3) Generates CoT reasoning chains with optimization strategies and state transitions as training outputs. The pipeline maintains process-architecture agnosticism - adapting to new AMP configurations only requires minimal adjustments to corresponding training data components while preserving core framework integrity. Dataset specifications are detailed in the upper panel of Fig.4.

1) LoRA Fine-tuning: Based on the dataset generated by the above pipeline, this method uses the widely applied efficient fine-tuning framework - LoRA[6] to fine-tune the Qwen2.5 LLM.

Specifically, for dataset $D = \{q_i, r_i, a_i \mid i = 0, 1, \ldots, n\}$, q_i, r_i and a_i represent the task description, reasoning process, and final answer, respectively. We use the following loss function to train Qwen2.5-7B-Instruct:

$$\mathbb{L} = \frac{1}{B}\sum_{b=0}^{B}\frac{1}{K-s}\sum_{k=s}^{K}\ell(f_\vartheta(x_{1,k}^b,), x_{k+1}^b) \tag{1}$$

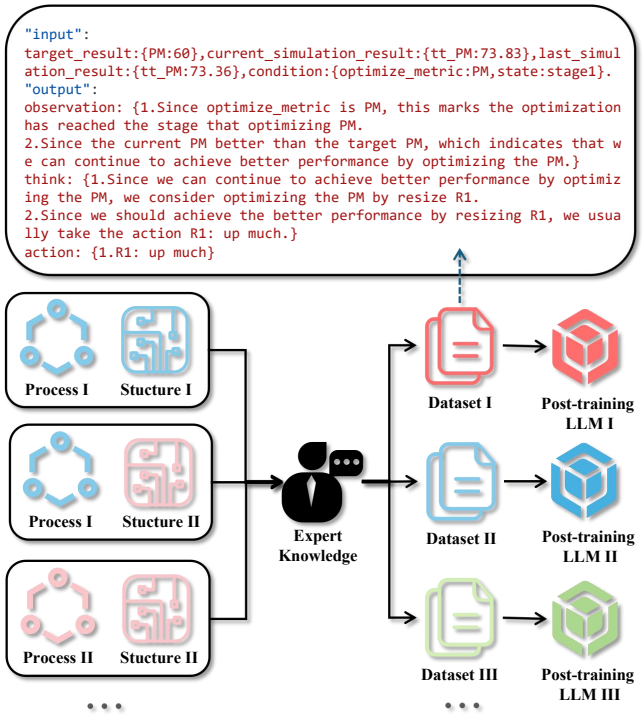


Fig. 4. Expert Knowledge Empowerment LLM

B represents the training batch size, K indicates the length of the output sequence $x = q_i \oplus r_i \oplus a_i$, ($\oplus$ represents string concatenation operation), ℓ represents the cross-entropy loss function, f_θ is the forward propagation process of the LLM (parameter θ), s is the index position of $r_i \oplus a_i$ in. Under the supervision of this loss function, LoRA will gradually optimize the corresponding trainable parameters.

2) LLM Quantization: In this work, the Q8_0 quantization method[4] provided by llama.cpp is used to quantize the model fine-tuned with LoRA, mapping the full-precision model parameters to an 8-bit integer representation. The entire quantization process is as follows: set the full precision weight matrix as $W \in \mathbb{R}^{m\times n}$, for any element in the matrix, it is mapped using the following formula:

$$q(w) = \text{clip}\left(\text{round}\left(\frac{W}{s}\right), -127, 127\right) \tag{2}$$

The quantization scaling factor optimally maps values to the [-127, 127] range of 8-bit signed integers. Jointly applying round(·) and clip(·) operators ensures integer compliance while preserving distribution characteristics. This calibration-free method achieves computational efficiency while maintaining model fidelity.

IV. Experiment

A. Experiment Setup

Hardware Environment: The experiments in subsection IV-F was conducted on the Armv9-based computing cloud platform (T-Head Yitian 710 CPU) provided by the IEEE AICAS 2025 Grand Challenge organizing committee. All other experiments, including LLM fine-tuning and quantization, were performed on an Intel Core i9-13900K with 1X Nvidia RTX 4090.

LLM Configuration: Our experiments involved fine-tuning, quantization, and testing of Qwen2.5-coder-(0.5B, 1.5B, 3B, 7B) and Qwen2.5-Instruct-(0.5B, 1.5B, 3B, 7B) models. We also evaluated mainstream LLMs, including DeepSeek, ChatGPT, Gemini, and Claude.

Evaluation Metrics: To quantify industrial AMP netlist performance, we adopted the widely recognized Figure of Merit (FoM) [12], calculated as follow:

$$\text{FoM} = \frac{GBW\,[\text{MHz}] \cdot C_L\,[\text{pF}]}{\text{Power}\,[\text{mW}]}. \tag{3}$$

Additionally, to evaluate LLM performance on our specific tasks, we employed the Exact Match (EM) metric, which measures how frequently the model's output precisely matches the correct answer.

B. Initiation Model Selection and Validation

To identify the optimal base model, we evaluated four scales and compared EM-SLRF initialization (Code Generation) with LLM-based direct calculation (Direct Calc.) in Table III. Results show: (1) While model size demonstrates a proportional relationship with EM, it concurrently induces an increase in computational runtime; (2) Qwen2.5-coder-3B achieves optimal efficiency-performance balance, selected as final configuration.

TABLE III
Impact of Model Size on Runtime and EM: Code Generation vs. Direct Calc.

Method	Model	Runtime (s)	EM (%)
Code Generation	coder-0.5B	4.21	25.1
	coder-1.5B	5.17	54.5
	coder-3B	6.36	91.8
	coder-7B	11.2	94.1
Direct Calc.	Instruct-7B	72.4	23.7

C. Evolution of Performance Metrics During Optimization

To analyze the impact of our optimization strategies on key metrics during iterative refinement, we recorded post-iteration simulation results across five process corners (ff, fs, sf, ss, tt) for all performance indicators, as illustrated in Fig.5.

Fig.5 demonstrates three critical findings: (1) Our strategy achieves significant improvements in critical metrics(Gain, Idc, and GBW); (2) CoT reasoning enables accurate circuit state evaluation and adjustment strategy generation, sustaining optimization trajectories; (3) Full convergence within 30 iterations. These collectively validate the methodology's efficacy.

D. Time Optimization Analysis

To select the optimal inference LLM, we evaluated five Qwen scales fine-tuned with identical datasets (Table IV).

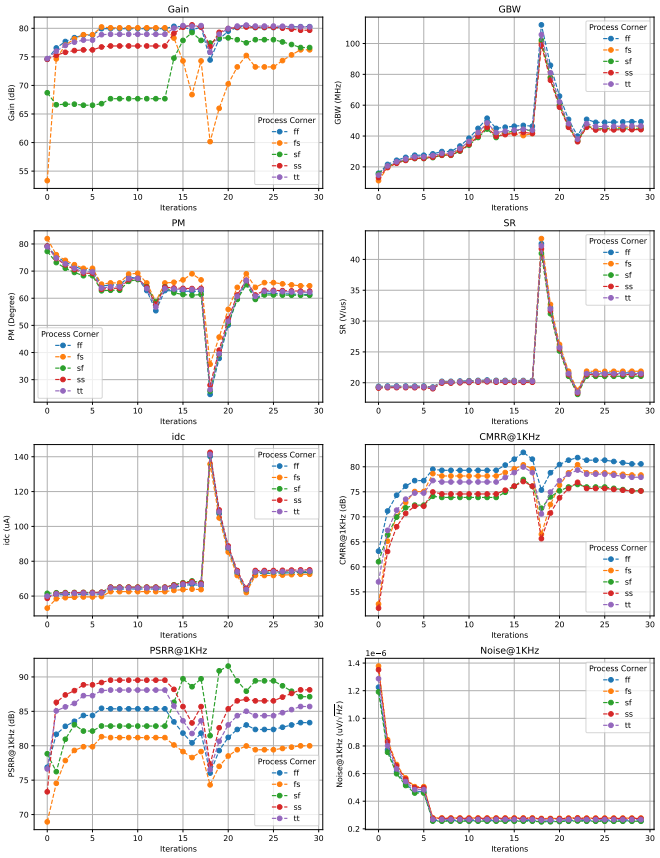


Fig. 5. Iterative Optimization Profiles of Key Performance Metrics

TABLE IV
PERFORMANCE COMPARISON ACROSS DIFFERENT MODEL SIZES

Model Size	Runtime (s)	EM (%)
Instruct-7B	65.58	99.38
Instruct-3B	35.04	99.38
Instruct-1.5B	21.33	99.69
Instruct-0.5B	12.21	99.53
Instruct-0.5B($Q8\_0$)	10.74	99.53

The 0.5B quantized model (Qwen2.5-instruct-0.5B-Q8_0) reduces inference time by 83.7% versus 7B while maintaining accuracy. Concurrent EM-SLRF optimization via three-stage refinement (Fig.6) cuts simulation iterations by 94.1%, yielding 93.9% total time reduction. In addition, comparative analysis reveals that the EM-SLRF methodology achieves a 20× acceleration when benchmarked against conventional manual processing techniques.

E. Comparative Analysis with Other LLMs

In order to study the performance of existing mainstream LLMs Deepseek, ChatGPT, Gemini, and Claude in designing AMP circuits, we asked them the same prompt and obtained the results shown in the table V. The existing mainstream LLMs for generating circuit gain have not met the standards, and other metrics are also not satisfactory when considered

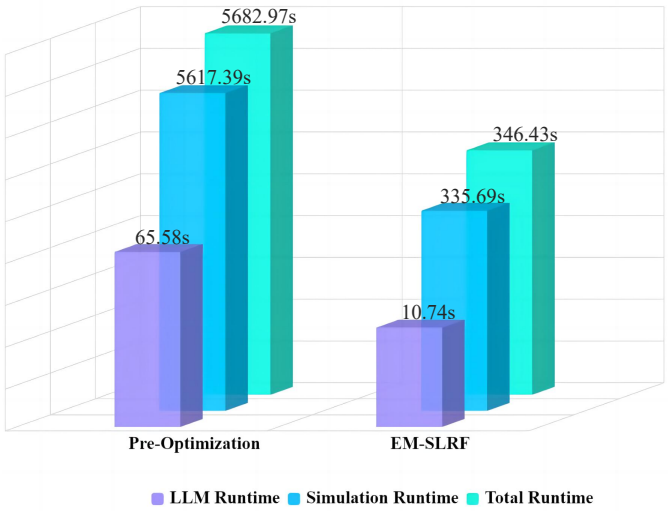


Fig. 6. Runtime Comparison: Pre-Optimization vs. EM-SLRF

comprehensively. Our framework can generate better netlists through optimization while meeting the competition's requirements, achieving the highest FoM value.

TABLE V
PERFORMANCE COMPARISON BETWEEN OUR METHOD AND MAINSTREAM LLMS

Metric	Our Method	Deepseek	ChatGPT	Gemini	Claude
Gain (dB)	77.38	27.31	16.71	14.01	25.77
Gbw (MHz)	40.32	8.13	21.65	8.01	124.75
Pm (°)	59.25	88.71	91.68	104.37	102.44
SR (V/μs)	20.95	22.34	185.28	28.12	61.35
Idc (μA)	74.94	206.12	79.17	459.59	819.72
CMRR@1KHz (dB)	79.04	64.78	45.20	23.39	39.89
PSRR@1KHz (dB)	86.20	57.92	43.11	35.59	35.49
Noise@1KHz ($\mu V/\sqrt{Hz}$)	0.24	0.28	0.51	25.29	0.59
FoM	**597.81**	43.83	303.85	19.37	169.11

F. Comparative Analysis with Other Methods

To verify the effectiveness of our method, we compared it with the methods of other competitors in the IEEE AICAS 2025 Grand Challenge, and the specific experimental results are shown in Table 6. It can be seen that our method achieved the highest levels in idc and Gain, while the industrial FoM value also reached its peak, proving that we have achieved state-of-the-art performance in FoM metrics.

G. Cross-Process and Cross-Structure AMP Sizing Validation

To verify the universality of our method, we selected the AMPs of the two structures in Fig1, while also choosing two different processes: Sky130 and SMIC055NLL for pairing. The specific experimental results are shown in Table VII. From the experimental results in Table VII, it can be seen that under various processes and structures, our method can still achieve industrial high-performance netlist generation, validating the effectiveness of our method for cross-process and cross-structure AMP sizing optimization.

TABLE VI
PERFORMANCE COMPARISON BETWEEN OUR METHOD AND OTHER TEAM

Metric	Our Method	Team I	Team II	Team III	Team IV	Team V	Team VI
Gain (dB)	**77.38**	48.42	71.64	77.31	71.06	72.46	57.89
GBW (MHz)	40.31	**103.95**	29.26	28.78	35.56	24.58	51.91
Pm (°)	59.24	60.38	55.59	61.21	**62.93**	60.66	56.66
SR (V/μs)	20.94	**109.32**	50.92	20.16	20.23	17.88	39.22
Idc (μA)	**74.94**	201.29	131.68	134.37	454.54	435.10	341.48
CMRR@1KHz (dB)	79.03	110.24	77.24	75.27	100.58	74.48	**58.36**
PSRR@1KHz (dB)	86.19	69.01	73.99	89.65	81.47	88.88	**67.76**
Noise@1KHz ($\mu V/\sqrt{Hz}$)	0.24	0.17	0.43	0.15	**0.12**	0.23	0.36
FoM	**597.73**	573.77	246.91	238.02	86.93	62.78	168.91

TABLE VII
PERFORMANCE COMPARISON OF AMP SIZING ACROSS DIFFERENT PROCESSES AND STRUCTURES

Metric	Structure I		Structure II	
	SKY130	SMIC055NLL	SKY130	SMIC055NLL
Gain (dB)	77.38	91.79	75.99	91.43
Gbw (MHz)	40.31	46.85	40.38	47.09
Pm (°)	59.25	61.08	61.37	60.02
SR (V/μs)	20.95	48.25	21.11	36.47
Idc (μA)	74.94	86.78	75.09	86.27
CMRR@1KHz (dB)	79.04	97.27	74.30	72.81
PSRR@1KHz (dB)	79.04	87.64	83.46	101.62
Noise@1KHz ($\mu V/\sqrt{Hz}$)	0.24	0.09	0.29	0.11
FoM	597.81	599.86	597.51	606.49

V. CONCLUSION

This work presents a multi-agent LLM-based framework for AMP sizing, integrating progressive optimization and expert knowledge empowerment to enable efficient, cross-process, and cross-structure design. Experiments demonstrate superior performance in speed and accuracy over traditional methods, significantly shortening the design cycle. The proposed knowledge empowerment approach ensures scalability across diverse circuit tasks. Future work may explore reinforcement learning for dynamic agent coordination and integration with EDA workflows to enhance practical usability.

REFERENCES

[1] Josh Achiam et al. "Gpt-4 technical report". In: *arXiv preprint arXiv:2303.08774* (2023).

[2] TIANCHI AICAS T-HEAD. "LLM-Based Automated Analog Circuit Design Track". IEEE AICAS 2025 Grand Challenge 2025. https://tianchi.aliyun.com/competition/entrance/532287/information. 2025.

[3] Hong Cai Chen et al. "DocEDA: Automated Extraction and Design of Analog Circuits from Documents with Large Language Model". In: *arXiv preprint arXiv:2412.05301* (2024).

[4] Tim Dettmers et al. "Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale". In: *Advances in neural information processing systems* 35 (2022), pp. 30318–30332.

[5] Cheng-Yu Hsieh et al. "Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes". In: *arXiv preprint arXiv:2305.02301* (2023).

[6] Edward J Hu et al. "Lora: Low-rank adaptation of large language models". In: *arXiv preprint arXiv:2106.09685* (2021).

[7] Tuotian Liao and Lihong Zhang. "Efficient parasitic-aware hybrid sizing methodology for analog and RF integrated circuits". In: *Integration* 62 (2018), pp. 301–313.

[8] Chengjie Liu et al. "Ampagent: An llm-based multi-agent system for multi-stage amplifier schematic design from literature for process and performance porting". In: *arXiv preprint arXiv:2409.14739* (2024).

[9] Chengjie Liu et al. "LADAC: Large language model-driven auto-designer for analog circuits". In: *Authorea Preprints* (2024).

[10] Wenlong Lyu et al. "An efficient Bayesian optimization approach for automated optimization of analog circuits". In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 65.6 (2017), pp. 1954–1967.

[11] Keertana Settaluri et al. "Automated design of analog circuits using reinforcement learning". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41.9 (2021), pp. 2794–2807.

[12] Jinyi Shen et al. "Atelier: An Automated Analog Circuit Design Framework via Multiple Large Language Model-based Agents". In: *Authorea Preprints* (2024).

[13] Jason Wei et al. "Chain-of-thought prompting elicits reasoning in large language models". In: *Advances in neural information processing systems* 35 (2022), pp. 24824–24837.

[14] Yuxuan Yin et al. "Ado-llm: Analog design bayesian optimization with in-context learning of large language models". In: *arXiv preprint arXiv:2406.18770* (2024).

[15] Ping Yu et al. "ALERT: Adapt language models to reasoning tasks". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2023, pp. 1055–1081.