Transactions on Computational and Scientific Methods | Vo. 6, No. 02, 2026
ISSN: 2998-8780
https://pspress.org/index.php/tcsm
Pinnacle Science Press

Adaptive Multi-Objective Timing Optimization for Multi-Mode Multi-Corner Signoff

Lauri Lahtinen
Dakota State University, Madison, USA
l.lahtinen97@gmail.com

Abstract: In the backend design flow of ultra-large-scale integrated circuits (ULSI), static timing analysis (STA) has become the most widely adopted method for timing signoff. However, multi-mode multi-corner (MMMC) timing analysis significantly increases the complexity of STA, making timing convergence during signoff more challenging. Based on a UMC 28 nm process technology and a ULSI backend design project, this work proposes an automated and accurate approach to resolve timing violations during signoff. The XTop tool is employed in a cross-platform environment to optimize timing violations, replacing the traditional method that requires manual scripting to back-annotate violating paths to the place-and-route (PR) stage for iterative fixes. Experimental results demonstrate that XTop can automatically and precisely repair a large number of hold-time violations without degrading setup timing performance.

Keywords: Static timing analysis; multi-mode multi-corner; ultra-large-scale integrated circuits; XTop; timing optimization

1. Introduction

With the rapid development of the information society, the integrated circuit industry has been widely applied in various domains, such as the Internet, healthcare, military, and communications [1]. The rapid advancement of integrated circuit technology has driven continuous process scaling, increasing chip complexity and operating frequency, thereby posing significant challenges to backend design [2]. In digital integrated circuit backend design, timing optimization is critical to ensuring correct functionality and meeting design specifications. It occupies a pivotal position in chip design and has become one of the primary bottlenecks in advanced-process ultra-large-scale integrated circuit design [3]. For advanced-process ULSI designs, the large number of standard cells, numerous modules, and high operating frequencies impose more stringent timing optimization requirements, making timing closure increasingly complex.

Static timing analysis (static timing analysis, STA), as one of the most important components of digital backend timing signoff, has become the most widely used timing analysis method. This approach simplifies and accelerates the verification and analysis of all timing violations in a design [4]. In advanced-process digital backend design flows, multi-mode multi-corner analysis is commonly adopted to ensure that chips operate reliably under various extreme conditions [5]. However, multi-mode multi-corner timing analysis increases the complexity of STA signoff, and achieving timing convergence under multi-mode multi-corner scenarios has become a key issue in digital backend design [6-7].

During the first timing signoff in digital backend design, unresolved timing violations often remain. The use of manual scripting within implementation tools to iteratively fix these violations is overly complex and unsuitable for ultra-large-scale designs. This paper adopts multi-mode multi-corner timing analysis in the backend signoff stage of ultra-large-scale digital integrated circuit design. By leveraging the XTop tool across platforms in conjunction with the Innovus tool, automated and accurate repair of timing violations is achieved, thereby improving the efficiency of timing closure.

2. Related work

Recent advances in intelligent data analysis increasingly emphasize the importance of capturing structural dependencies when modeling complex systems. A representative study in this direction is presented in [8], where anomaly ranking is performed through the identification of latent structural deviations combined with reconstruction consistency. This work demonstrates that structural irregularities embedded within relational data can serve as effective signals for anomaly detection and risk assessment.

Building on this idea, subsequent studies further explore relational representation learning for modeling complex interactions among entities. Graph-based modeling frameworks and relational reasoning mechanisms have been shown to effectively capture multi-hop dependencies and propagate information across structured data environments [9-11]. These approaches highlight the importance of explicitly modeling relationships among entities when performing intelligent analysis and prediction tasks.

Beyond structural dependencies, temporal dynamics also play a critical role in many real-world systems where behaviors evolve continuously over time. To address this challenge, residual-regulated forecasting approaches introduce second-order differencing mechanisms to mitigate the effects of non-stationary temporal patterns and improve prediction stability [12]. Such techniques provide useful insights for capturing evolving system behaviors under dynamically changing conditions.

Recent studies further extend this line of work by integrating structural information with temporal modeling. Joint structural-temporal learning frameworks enable systems to capture both spatial correlations and time-varying patterns within large-scale environments [13]. By simultaneously modeling structural relationships and temporal evolution, these frameworks significantly improve predictive robustness in dynamic system monitoring tasks.

A number of studies explore anomaly perception and performance monitoring from a spatiotemporal representation learning perspective. Graph-structured temporal modeling, self-supervised spatiotemporal learning, and structure-temporal collaborative detection strategies enable systems to identify abnormal behavioral patterns within large-scale distributed environments [14-17]. These works demonstrate that combining structural dependencies with temporal dynamics can significantly enhance the capability of intelligent monitoring systems.

In addition to representation learning, system-level optimization and adaptive decision mechanisms have also attracted increasing attention. Reinforcement learning and hierarchical decision frameworks provide effective solutions for addressing complex optimization problems such as task scheduling and resource allocation [18-21]. Through sequential decision-making and feedback-driven learning processes, these methods enable systems to adaptively optimize operational performance under varying workloads and system conditions.

Another important methodological direction focuses on causal reasoning and bias-aware learning mechanisms. Causal inference frameworks leverage structured knowledge representations to distinguish causal relationships from spurious correlations, thereby enabling more reliable decision-making processes [22-25]. These approaches are particularly valuable in environments where hidden confounding factors or distribution shifts may influence system performance.

Recent progress in large language models and intelligent agent systems further expands the methodological landscape. Memory-driven planning strategies and dynamic retrieval mechanisms allow intelligent agents to perform long-horizon reasoning tasks more effectively [26-28]. Meanwhile, hierarchical parameter-freezing techniques improve the efficiency of large model training by balancing computational cost and model performance [29].

Generative learning frameworks have also introduced powerful tools for structured representation modeling. Diffusion-based generative models with conditional control enable flexible synthesis of structured representations and provide new opportunities for controllable generation and representation learning [30]. In addition, uncertainty-aware modeling approaches improve the reliability of AI systems by incorporating risk-awareness mechanisms into model inference and decision processes [31].

As intelligent systems become increasingly distributed, multi-agent coordination has become another important research topic. Context-aware trust evaluation frameworks provide robust mechanisms for managing collaboration among multiple intelligent agents and ensuring stable system coordination in complex environments [32].

Finally, several studies explore machine-learning-driven optimization in large-scale computational infrastructures. Techniques such as predictive autoscaling, serverless inference scheduling, and intelligent load forecasting enable efficient resource utilization and scalable system management [33-35]. In parallel, recent research in electronic design automation introduces data-driven approaches for accelerating timing analysis and debugging in complex design flows [36]. Complementary analytical studies on statistical timing analysis further provide theoretical foundations for modeling delay variations and timing constraints in large-scale systems [37].

Together, these studies provide a comprehensive methodological foundation spanning structural representation learning, temporal modeling, causal reasoning, optimization strategies, and intelligent system coordination. The integration of these complementary techniques inspires the methodological design of the proposed framework and provides theoretical support for building robust and scalable intelligent systems.

3. Static Timing Analysis

The digital integrated circuit backend design flow is primarily realized using electronic design automation (electronic design automation, EDA) tools. Innovus is a commonly used place and route (place and route, PR) implementation tool. After chip physical design is completed, final physical verification, functional verification, and timing verification must be performed prior to tape-out [38]. Multi-mode multi-corner STA has become a fundamental requirement for timing verification.

3.1 Significance of Timing Analysis

STA is an essential method for timing verification in digital integrated circuit backend design. It analyzes the delay of each path through exhaustive enumeration [39]. The design is partitioned into a series of timing paths, and based on the delay of each path, setup and hold times are analyzed to determine whether each path satisfies timing constraints [40]. Dynamic timing analysis (dynamic timing analysis, DTA), also known as timing simulation, is another method for timing verification. It validates the logical functionality of devices under actual delay conditions and requires input stimulus; however, its runtime is relatively slow [41]. Compared with DTA, STA runs significantly faster and provides more comprehensive coverage of timing paths. Therefore, STA is mandatory in backend timing signoff. Since DTA is slower and cannot comprehensively check all critical paths, it is generally used selectively.

3.2 Challenges

During chip design, to ensure that a digital chip operates as expected, designers must verify whether the arrival times of clock and data signals at registers satisfy setup and hold constraints, and perform timing optimization on paths that violate these constraints. In backend design, STA is performed under multi-mode multi-corner conditions during timing signoff. Typically, full timing convergence is not achieved in the first signoff, and some timing violations remain [42]. The reasons for this situation are manifold, commonly including:

(1) Differences in the timing library files used during the physical design stage and timing signoff stage;

(2) Differences in the timing analysis models used by tools in the physical design stage and timing signoff stage;

(3) More comprehensive multi-mode multi-corner scenarios during timing signoff compared with the physical design stage;

(4) Amplification or reduction of clock skew effects during timing signoff compared with the physical design stage.

3.3 Traditional Timing Repair Methods

For timing violations encountered during timing signoff, the traditional solution is to manually write repair scripts and back-annotate them to Innovus for engineering change order (engineering change order, ECO) fixes. Hold violations are typically resolved by increasing the delay of data paths, for example by inserting buffer cells into data paths through scripting. Although this method can fix hold violations, it cannot monitor setup timing changes in real time and may result in setup degradation. Manual repair may achieve satisfactory results for small-scale chips with few violations. However, for ultra-large-scale integrated circuit designs, especially under multi-mode multi-corner timing signoff conditions where numerous violations require fixing, manual repair becomes impractical. Therefore, a technical tool capable of monitoring multi-scenario timing variations in real time during timing violation optimization is required.

4. Application of XTop

The timing optimization process imposes stringent requirements on timing optimization tools. During optimization, the tool must not only support large-scale data processing capability but also ensure that any modification to cells or routing is reflected in real time across the entire chip and across all process corners. This enables timely detection of newly introduced timing violations caused by layout adjustments under different paths or operating corners. Therefore, the optimization framework must continuously monitor timing variations across multiple paths while maintaining global design consistency.

In addition, modifications to cells or routing may introduce design rule violations (DRC), which must also be considered during optimization. To address these challenges, this work adopts the XTop tool to provide systematic solutions for timing violations during signoff. The optimization flow can be scripted and directly invoked by the PR tool, enabling efficient and scalable resolution of timing violations in ultra-large-scale integrated circuit designs.

To further improve optimization efficiency and robustness, this work incorporates methodological principles inspired by several recent system-level optimization approaches.

First, the hierarchical resource coordination strategy proposed by Yang et al. in the Cost-TrustFL framework is leveraged to guide large-scale optimization management [43]. Cost-TrustFL fundamentally introduces a hierarchical federated learning architecture that combines cost-aware scheduling with lightweight reputation evaluation to coordinate distributed learning tasks across heterogeneous cloud environments. In this work, we apply and adapt the hierarchical coordination principle to timing optimization by organizing repair operations across different optimization scopes (path-level, module-level, and chip-level). By adopting and building upon this hierarchical control concept, the XTop-driven repair flow can prioritize critical violating paths while maintaining global optimization stability across multiple MMMC scenarios.

Second, adaptive multi-objective decision mechanisms are incorporated by building upon the reinforcement learning-based optimization strategy proposed by Lyu et al. [44]. Their work introduces a deep reinforcement learning framework for multi-objective adaptive rate limiting in microservice architectures, where the learning agent dynamically balances conflicting objectives such as latency, throughput, and system stability. Inspired by this methodology, the present work leverages and extends the idea of adaptive multi-objective optimization to timing closure. During XTop-driven repairs, the optimization process applies multi-objective decision strategies that simultaneously consider setup timing, hold timing, and design rule constraints. By adopting this adaptive balancing principle, the repair flow can dynamically select appropriate optimization actions such as buffer insertion, cell resizing, or routing adjustment while preventing degradation of setup timing performance.

Third, predictive resource management mechanisms are incorporated by drawing upon the proactive scheduling strategy introduced by Ni et al. in Predictive-LoRA [45]. Predictive-LoRA fundamentally proposes a proactive inference management system that predicts workload demands and allocates resources in advance to avoid fragmentation and latency spikes in serverless LLM inference environments. In this study, we apply and extend the predictive scheduling concept to timing repair tasks. The XTop optimization framework leverages predictive estimation of violation propagation across multiple timing paths and process corners. By incorporating this predictive analysis mechanism, the repair process can anticipate potential timing degradations introduced by optimization actions and proactively adjust repair strategies, thereby improving timing convergence efficiency under multi-mode multi-corner conditions.

Through the integration of hierarchical coordination, adaptive multi-objective decision making, and predictive optimization strategies, the XTop-based repair flow builds upon and extends existing optimization methodologies to better address the complexity of MMMC timing signoff in ultra-large-scale integrated circuit designs.

4.1 XTop Timing Repair Flow

After place and route (PR), the netlist is extracted using the Innovus tool, and the parasitic parameter file (SPEF) is generated. The extracted netlist and SPEF file are then imported into PrimeTime (PT) for multi-mode multi-corner STA signoff. As illustrated in Figure 1, once timing violations are detected, the automated XTop repair flow is initiated.

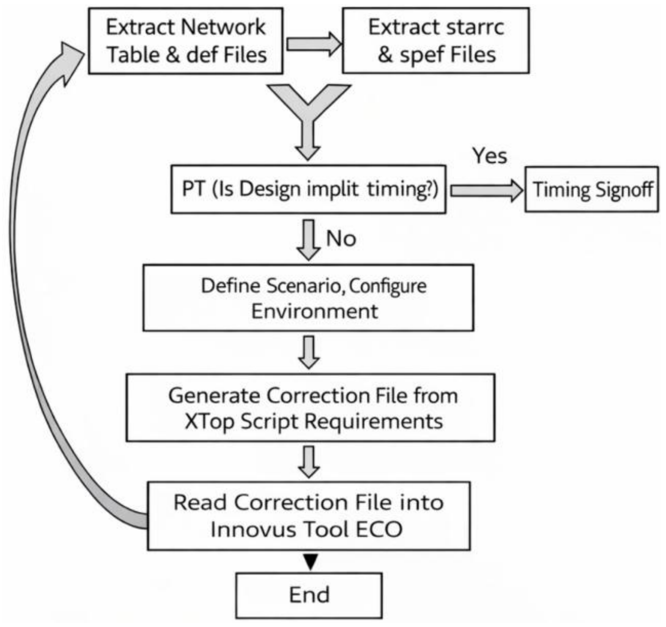
Extract Network Table & def Files
Extract starrc & spef Files
PT (Is Design implit timing?)
Yes
Timing Signoff
No
Define Scenario, Configure Environment
Generate Correction File from XTop Script Requirements
Read Correction File into Innovus Tool ECO
End

Figure 1. XTop Timing Repair Flow

Before repair, the number of available licenses and scenarios must be determined. The number of concurrent processes is constrained by the license count. Scenario selection should at minimum include the worst-timing and highest-violation scenarios reported by PT. Scenarios may be selected as cross-combinations of operating modes and process corners to ensure sufficient coverage while remaining within the allowable parallel process range.

According to project requirements, the repair script specifies:

the type of timing violation to be fixed (setup, hold, or transition),

the timing analysis mode (GBA or PBA),

the repair strategy (cell sizing or buffer insertion).

After configuration, the XTop process parses violation information from PT reports for the selected scenarios. Duplicate violation paths are automatically filtered before optimization begins. The repair strategy can be formally expressed as follows.

For setup timing constraints:

$$T_{clk} \geq T_{data} + T_{setup} + M_{setup}$$

For hold timing constraints:

$$T_{data} + M_{hold} \geq T_{hold}$$

For transition constraints:

$$T_{transition} \leq T_{limit} - M_{transition}$$

where M_{setup}, M_{hold}, and $M_{transition}$denote the safety margins applied during optimization.

The tool first performs simulated optimization using cell sizing to estimate post-repair timing results. For unresolved violations, buffers are inserted along data paths based on the delay characteristics of different buffer cells. The number of inserted buffers is determined according to the slack value of each path to ensure precise correction.

After optimization, the tool automatically generates Tool Command Language (TCL) scripts, which are executed in Innovus to perform ECO modifications and rerouting. The updated netlist and parasitic files are then re-extracted for subsequent timing verification. If certain violation paths cannot be resolved through sizing or buffering, the tool reports the failure reasons through a diagnostic command such as "report_fail_reasons".

4.2 Practice in an Ultra-Large-Scale Chip

The proposed method was applied to an ultra-large-scale chip based on the UMC 28 nm process. The design included 22 scenarios, operated at 1.2 GHz, and had a scale of 2.4 billion gates. Multi-mode multi-corner STA was performed in PT after PR. The initial PT results showed 1 setup violation and 3113 hold violations across all scenarios.

XTop was employed for timing optimization, focusing primarily on hold violations. The repair process was executed over four iterative rounds. In the first round, both setup and hold violations were addressed. The remaining three rounds focused on hold violations only. Each ECO iteration required approximately 90 minutes.

Table 1 presents the PT timing results after four repair iterations. WNS represents the worst negative slack, measured in nanoseconds (ns). After four iterations, all hold violations were successfully closed, and setup timing was not degraded due to hold optimization. As shown in the line chart of Figure 2, the number of violations decreases with each repair iteration.

Table 1: Timing Comparison Before and After Four Iterative Repairs Using XTop

PT Results	Hold WNS (ns)	Number of Hold Violations	Setup WNS (ns)
Initial	-0.115	3113	-0.067
1st Iteration	-0.031	1361	0
2nd Iteration	-0.007	99	0
3rd Iteration	-0.002	8	0
4th Iteration	0	0	0

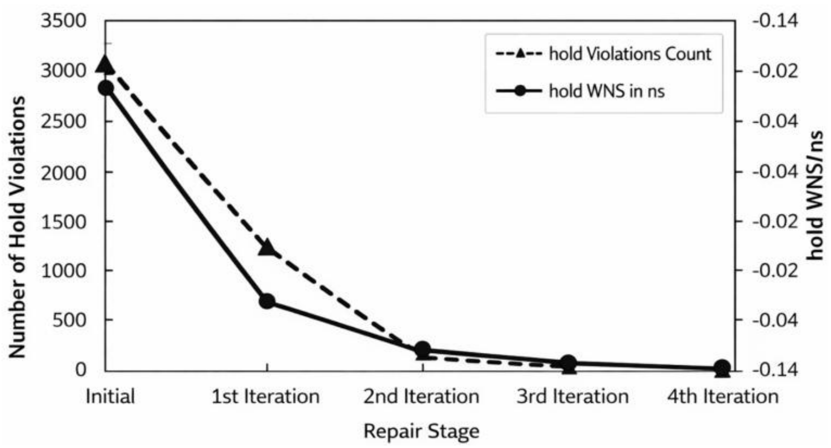

Figure 2. Hold Timing Change Over Four Iterations

After four iterations, all hold violations were eliminated. Examination of the repair scripts indicates that optimization was first performed by adjusting cell drive strength or cell type to improve timing paths. For paths that remained unresolved, buffer insertion was applied along data paths to increase path delay and achieve hold closure. The tool identifies delay characteristics of different buffer cells and determines the appropriate insertion quantity according to the slack of each path. The numbers of sized and inserted cells in the four iterations are summarized in Table 2 and Figure 3. As the number of violations decreases, the counts of sized and inserted cells correspondingly decline.

Table 2: Statistics of Sized and Inserted Cells Over Four XTop Repair Iterations

Iteration	Sized	Inserted
1st Iteration	7997	2607
2nd Iteration	1322	393
3rd Iteration	67	33
4th Iteration	5	3

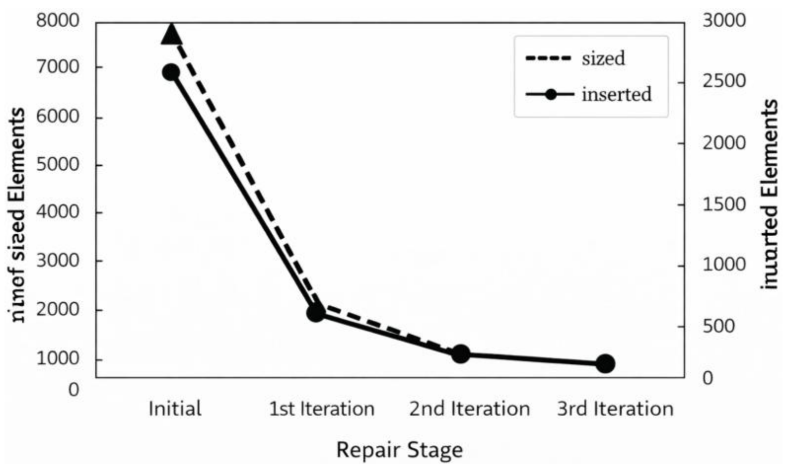

Figure 3. Sized and Inserted Cell Counts Over Four Iterations

5. Conclusion

In the backend timing signoff stage of ultra-large-scale integrated circuit design, this paper proposes an automated and precise timing optimization approach using XTop under a multi-mode multi-corner analysis environment. The proposed method replaces the traditional manual scripting approach for violation repair and has been validated in an advanced UMC 28 nm ultra-large-scale backend design project.

Experimental results demonstrate that XTop can accurately repair timing violations during the multi-mode multi-corner signoff stage. After four rounds of automated iterative repair, 100% of hold violations were successfully closed. Moreover, the tool is capable of monitoring timing variations across multiple scenarios in real time during the repair process, effectively preventing setup degradation caused by hold optimization.

References

[1] A. B. Kahng, J. Lienig, I. L. Markov, and J. Hu, VLSI Physical Design: From Graph Partitioning to Timing Closure, vol. 312. Netherlands: Springer, 2011.

[2] D. Blaauw, K. Chopra, A. Srivastava, and L. Scheffer, “Statistical timing analysis: From basic principles to state of the art,” IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 27, no. 4, pp. 589-607, Apr. 2008.

[3] S. Liu, Z. Wang, F. Liu, Y. Lin, B. Yu and M. D. Wong, "Sign-off timing considerations via concurrent routing topology optimization," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 44, no. 5, pp. 1942-1953, 2024.

[4] R. Chadha and J. Bhasker, Static Timing Analysis for Nanometer Designs. Springer-Verlag, 2009.

[5] R. J. Baker, CMOS: Circuit Design, Layout, and Simulation. Hoboken, NJ, USA: John Wiley & Sons, 2019.

[6] M. Chentouf, F. Stevmelin, and Z. E. A. A. Ismaili, “Power-aware hold optimization for ASIC physical synthesis,” Integr., VLSI J., vol. 76, pp. 13-24, 2021.

[7] S. Roy, P. M. Mattheakis, L. Masse-Navette, and D. Z. Pan, “Clock tree resynthesis for multi-corner multi-mode timing closure,” in Proc. Int. Symp. Phys. Des. (ISPD), Mar. 2014, pp. 69-76.

[8] H. Chen, R. Wu, C. Chen, H. Feng, Y. Nie and Y. Lu, "Anomaly Ranking for Enterprise Finance Using Latent Structural Deviations and Reconstruction Consistency," 2026.

[9] K. Cao, Y. Zhao, H. Chen, X. Liang, Y. Zheng and S. Huang, "Multi-Hop Relational Modeling for Credit Fraud Detection via Graph Neural Networks," 2025.

[10]X. Song, Y. Huang, J. Guo, Y. Liu and Y. Luan, "Multi-scale Feature Fusion and Graph Neural Network Integration for Text Classification with Large Language Models," arXiv preprint arXiv:2511.05752, 2025.

[11]C. Chiang, "Collaborative Machine Learning for Risk Ranking Under Concurrent Class Imbalance and Distribution Shift," 2026.

[12]Y. Ou, S. Huang, R. Yan, K. Zhou, Y. Shu and Y. Huang, "A Residual-Regulated Machine Learning Method for Non-Stationary Time Series Forecasting Using Second-Order Differencing," 2025.

[13]Q. Zhang, "An Artificial Intelligence Framework for Joint Structural-Temporal Load Forecasting in Cloud Native Platforms," arXiv preprint arXiv:2602.22780, 2026.

[14]C. Hua, "Anomaly Perception and Early Fault Prediction in Cloud Services via Graph-Structured Temporal Representation Learning," 2026.

[15]Y. Liu, "AI-Driven Performance Degradation Identification via Self-Supervised Spatiotemporal Graph Modeling in Microservice Systems," 2026.

[16]D. Wu, "Deep Learning Approach to Structure-Temporal Collaborative Anomaly Detection in Microservice Architectures," 2026.

[17]Y. Wang, "Adaptive Graph Construction and Spatiotemporal Contrastive Learning for Intelligent Cloud Service Monitoring," 2026.

[18]C. Nie, "Adaptive ETL Task Scheduling via Hierarchical Reinforcement Learning with Joint Rewards for Latency and Load Balancing," 2026.

[19]Y. Wang, "An AI-Based Temporal-Structural Fusion Framework for Robust Backend Load Prediction in Cloud-Native Environments," 2026.

[20]B. Chen, "FlashServe: Cost-Efficient Serverless Inference Scheduling for Large Language Models via Tiered Memory Management and Predictive Autoscaling," 2025.

[21]Y. Ma, "Anomaly detection in microservice environments via conditional multiscale GANs and adaptive temporal autoencoders," Transactions on Computational and Scientific Methods, vol. 4, no. 10, 2024.

[22]R. Ying, Q. Liu, Y. Wang and Y. Xiao, "AI-Based Causal Reasoning over Knowledge Graphs for Data-Driven and Intervention-Oriented Enterprise Performance Analysis," 2025.

[23]S. Sun, "CIRR: Causal-Invariant Retrieval-Augmented Recommendation with Faithful Explanations under Distribution Shift," arXiv preprint arXiv:2512.18683, 2025.

[24]S. Li, Y. Wang, Y. Xing and M. Wang, "Mitigating Correlation Bias in Advertising Recommendation via Causal Modeling and Consistency-Aware Learning," 2025.

[25]H. Feng, Y. Wang, R. Fang, A. Xie and Y. Wang, "Federated Risk Discrimination with Siamese Networks for Financial Transaction Anomaly Detection," Proceedings of the 2nd International Conference on Digital Economy and Computer Science, pp. 231-236, 2025.

[26]Y. Wang, R. Yan, Y. Xiao, J. Li, Z. Zhang and F. Wang, "Memory-Driven Agent Planning for Long-Horizon Tasks via Hierarchical Encoding and Dynamic Retrieval," 2025.

[27]Y. Luan, "Long Text Classification with Large Language Models via Dynamic Memory and Compression Mechanisms," Transactions on Computational and Scientific Methods, vol. 4, no. 7, 2024.

[28]T. Guan, "A Multi-Agent Coding Assistant for Cloud-Native Development: From Requirements to Deployable Microservices," 2025.

[29]J. Guo, "Balancing Performance and Efficiency in Large Language Model Fine-Tuning through Hierarchical Freezing," Transactions on Computational and Scientific Methods, vol. 4, no. 6, 2024.

[30]R. Liu, L. Yang, R. Zhang and S. Wang, "Generative Modeling of Human-Computer Interfaces with Diffusion Processes and Conditional Control," arXiv preprint arXiv:2601.06823, 2026.

[31]S. Pan and D. Wu, "Trustworthy Summarization via Uncertainty Quantification and Risk Awareness in Large Language Models," 2025 6th International Conference on Computer Vision and Data Mining (ICCVDM), pp. 523-527, 2025.

[32]K. Gao, H. Zhu, R. Liu, J. Li, X. Yan and Y. Hu, "Contextual Trust Evaluation for Robust Coordination in Large Language Model Multi-Agent Systems," 2025.

[33]J. Xu, Z. Han, L. Jin, S. Wu, H. Yan and L. Shi, "FACT: Fast and Accurate Multi-Corner Predictor for Timing Closure in Commercial EDA Flows," Proceedings of the 2024 ACM/IEEE International Symposium on Machine Learning for CAD, pp. 1-7, 2024.

[34]J. Zhou et al., "SetupKit: Efficient Multi-Corner Setup/Hold Time Characterization Using Bias-Enhanced Interpolation and Active Learning," 2025 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2025.

[35]W. Lv, Y. Xia, X. Chen and L. Kuang, "ViTAD: Timing Violation-Aware Debugging of RTL Code Using Large Language Models," arXiv preprint arXiv:2508.13257, 2025.

[36]D. Mishagli, E. Koskin and E. Blokhina, "Gate-Level Statistical Timing Analysis: Exact Solutions, Approximations and Algorithms," arXiv preprint arXiv:2401.03588, 2024.

[37]S. Hassoun, C. Cromer and E. Calvillo-Gámez, "Static Timing Analysis for Level-Clocked Circuits in the Presence of Crosstalk," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 22, no. 9, pp. 1270-1277, 2003.

[38]A. B. Kahng, U. Mallappa, and L. Saul, “Using machine learning to predict path-based slack from graph-based timing analysis,” in Proc. IEEE 36th Int. Conf. Comput. Des. (ICCD), Oct. 2018, pp. 603-612.

[39]V. Betz, J. Rose, and A. Marquardt, Architecture and CAD for Deep-Submicron FPGAs, vol. 497. New York, NY, USA: Springer Science & Business Media, 2012.

[40]J. Lamoureux and S. J. E. Wilton, “On the interaction between power-aware FPGA CAD algorithms,” in Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD), Nov. 2003, pp. 701-708.

[41]A. Agarwal, D. Blaauw, and V. Zolotov, “Statistical timing analysis for intra-die process variations with spatial correlations,” in Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD), Nov. 2003, pp. 900-907.

[42]S. Naunheim, F. Mueller, V. Nadig, Y. Kuhl, J. Breuer, N. Zhang, et al., “Holistic evaluation of a machine learning-based timing calibration for PET detectors under varying data sparsity,” Phys. Med. Biol., vol. 69, no. 15, Art. no. 155026, 2024.

[43]J. Yang, J. Chen, Z. Huang, C. Xu, C. Zhang and S. Li, "Cost-TrustFL: Cost-Aware Hierarchical Federated Learning with Lightweight Reputation Evaluation across Multi-Cloud," arXiv preprint arXiv:2512.20218, 2025.

[44]N. Lyu, Y. Wang, Z. Cheng, Q. Zhang and F. Chen, "Multi-Objective Adaptive Rate Limiting in Microservices Using Deep Reinforcement Learning," Proceedings of the 4th International Conference on Artificial Intelligence and Intelligent Information Processing, pp. 862-869, Oct. 2025.

[45]Y. Ni, X. Yang, Y. Tang, Z. Qiu, C. Wang and T. Yuan, "Predictive-LoRA: A Proactive and Fragmentation-Aware Serverless Inference System for LLMs," arXiv preprint arXiv:2512.20210, 2025.