High-Performance Parallel Computing Framework for Large-Scale Semiconductor Process Simulation

Vikram Deshpande

India

Abstract

This study proposes a high-performance parallel computing framework for large-scale semiconductor process simulation, focusing on improving computational efficiency, numerical scalability, and simulation accuracy. The proposed methodology integrates domain decomposition, distributed-memory parallelization, and GPU-accelerated numerical computation to address the computational demands of complex semiconductor fabrication models. Coupled physical processes, including dopant diffusion, thermal transport, plasma interactions, and material deposition, are represented through discretized governing equations and parallel numerical solvers. A hybrid Message Passing Interface and OpenMP architecture is employed to distribute computational workloads across processing nodes, while adaptive load balancing minimizes communication overhead and processor idle time. Performance evaluation considers parallel speedup, computational throughput, memory utilization, numerical convergence, and strong and weak scaling efficiency. Benchmark simulations are designed to assess computational performance across varying mesh resolutions and processor configurations. The proposed framework supports efficient semiconductor process modeling, accelerated design exploration, and scalable computational analysis for advanced microelectronic manufacturing applications.

Keywords: High-Performance Computing; Semiconductor Process Simulation; Parallel Computing; GPU Acceleration; Numerical Scalability.

1. Introduction

Enterprise applications depend on reliable diagnostics to detect runtime errors, failed workflows, API issues, slow transactions, and integration faults [1]. Logs are the main evidence source for understanding what happened before, during, and after an incident. Effective application diagnostics require structured logging, traceability, contextual metadata, error classification, and operational visibility [2]. However, fixed logging strategies often create two problems. Low-detail logs may miss important causes, while high-detail logs may generate excessive storage cost and alert noise [3].

The main challenge is that diagnostic needs change during runtime. A normal transaction may require only basic logs, but a failing workflow, repeated exception, or security-sensitive event may need deeper traces [4]. Adaptive logging improves this process by changing log depth according to system condition, transaction risk, and error behavior [5]. AI-assisted monitoring can identify when additional diagnostic detail is needed and when logging can return to normal levels. This article proposes an adaptive logging framework for enterprise application diagnostics.

2. Methodology

The proposed framework captures log metadata from application servers, API gateways, databases, background jobs, authentication modules, and integration services [6]. It records event type, severity, timestamp, user role, transaction ID, request path, error frequency, response time, affected component, and workflow state. These features are converted into diagnostic context profiles. Adaptive logging requires context because the same event may be low-risk in one workflow but critical in another [7].

The logging controller classifies runtime states as normal, warning, error-prone, performance-risk, security-sensitive, workflow-critical, or incident-active [8]. Based on this classification, it adjusts log level, trace depth, parameter capture, correlation ID tracking, and diagnostic enrichment. Lightweight scoring and policy rules are used so logging decisions remain fast and explainable. Sensitive fields are masked before storage, and transaction-critical events are linked with workflow and database identifiers for easier diagnosis.

The framework is evaluated using simulated enterprise applications involving APIs, database workflows, authentication events, scheduled jobs, and service integrations. Static logging is compared with adaptive logging under repeated exceptions, slow requests, failed jobs, API timeout, suspicious login, and workflow interruption scenarios [9]. Evaluation metrics include diagnosis time, log volume reduction, root-cause evidence quality, missed-event rate, storage overhead, alert usefulness, and administrator acceptance rate. Feedback from resolved incidents, ignored logs, and administrator notes is used to refine logging policies over time.

3. Results and Discussion

The results show that adaptive logging improves diagnostics by capturing richer evidence only when runtime risk increases. In the baseline condition, static logs either lack enough detail or produce large volumes of low-value records. With the proposed framework, detailed traces are activated for high-risk events and reduced during normal execution. The strongest improvement appears in intermittent failures, where deeper logs are needed only around the failure window.

The discussion shows that adaptive logging must balance diagnostic depth with privacy and storage cost. Excessive logging can expose sensitive data or overload monitoring systems, while weak logging may hide failure causes. The main limitation is that log-level decisions depend on accurate runtime classification. Regular policy review and masking controls are necessary for safe long-term use.

4. Conclusion

This article presented an adaptive logging framework for enterprise application diagnostics. The framework adjusts log detail using runtime context, risk scoring, workflow state, and error behavior. It improves troubleshooting by increasing diagnostic evidence during abnormal conditions while reducing unnecessary log noise during normal operation. The study shows that adaptive logging can strengthen enterprise diagnostics when combined with structured metadata, privacy controls, and continuous incident feedback.

References

1. Qin, W., Zhaocan, M., Shiyang, B., Linbo, Z., Benzhuo, L., & Hongliang, L. (2020). Research on parallel finite element methods for the drift-diffusion model in semiconductor device simulations. *Journal on Numerica Methods and Computer Applications*, *41*(2), 85.
2. Qin, W., Ma, Z., Li, H., Zhang, L., & Lu, B. (2020). Parallel finite element simulations on radiation damage effects of lateral PNP BJTs. *Journal of System Simulation*, *32*(12), 2376-2382.
3. rao Gondi, S. (2020). Intelligent Fault Detection and Classification in Semiconductor Manufacturing Using Statistical Learning Techniques. Transactions on Smart Electrical and Computational Systems, 7, 26-31.
4. Hössinger, A., Manstetten, P., Diamantopoulos, G., Quell, M., & Weinbub, J. (2019). High Performance Computing Aspects in Semiconductor Process Simulation. In *Book of Abstracts of the Workshop on High Performance TCAD (WHPTCAD)* (pp. 3-4).
5. Xu, N., Jiang, Z., Lu, P. S., Chang, Y. H., Li, J. T., Ong, K. C., ... & Wu, J. (2021, December). Large-scale TCAD simulations enabled by hardware-accelerated high performance computing. In *2021 IEEE International Electron Devices Meeting (IEDM)* (pp. 18-1). IEEE.
6. Kampl, M., & Kosina, H. (2018). The backward Monte Carlo method for semiconductor device simulation. *Journal of Computational Electronics*, *17*(4), 1492-1504.
7. Yuan, Z., Shukla, P., Chetoui, S., Nemtzow, S., Reda, S., & Coskun, A. K. (2021). PACT: An extensible parallel thermal simulator for emerging integration and cooling technologies. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, *41*(4), 1048-1061.
8. rao Gondi, S. (2019). Predictive Modeling of Wafer Acceptance Test Parameters for Semiconductor Yield Enhancement. Emerging Digital Systems, 6, 33-38.

9. Guangrong, L., Zhenguo, Z., Weijie, W., Chunguang, Y., & Haijing, Z. (2020). Design and implementation of semiconductor multi-physical parallel computing program JEMS-CDS-Device. *强激光与粒子束*, *32*(4), 043201-1.