SCA / HPCAsia 2026
Everything with HPC - AI, Cloud, QC and Future Society
Date ··· January 26-29, 2026 | Venue ··· Osaka International Convention Center(Osaka, Japan)

Optimization of Supercomputer Graph Processing Performance Based on Graph500 Benchmark

I. Performance Enhancement Strategy Using XGBoost Machine Learning Model

Korea Institute of Science and Technology Information
Hyungwook Shim | shw@kisti.re.kr ··· Minho Suh | mhsuh@kisti.re.kr

Abstract

As global competition in deploying large-scale LLMs intensifies, graphical processing performance has become a critical factor in modern HPC systems. Many countries and supercomputing centers are now adopting GPU-dedicated or hybrid AI computing architectures, accelerating the development and diffusion of LLM services. To support this transition, it is essential to establish an effective strategy for improving graph processing performance beyond traditional hardware-centric approaches.

This study develops a tree-based machine learning model using the Graph500 dataset to predict GTEPS performance and identify key factors influencing HPC graphical processing. Based on the XGBoost analysis, the research proposes an optimization strategy that considers the combined effects of memory, problem scale, core allocation, and computation time. The results provide practical insights for enhancing HPC performance in AI-driven industrial environments.

KEYWORDS
Graph500, HPC, AI computing, LLM, XGBoost

Theoretical Background of the XGBoost Model

XGBoost is a gradient boosting–based ensemble method that builds multiple decision trees sequentially, with each new tree trained to correct the residual errors of the previous predictions. The model aggregates the outputs of all trees to generate the final prediction:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i)$$

At each boosting step t, a new tree is added to update the model:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i)$$

Model training minimizes an objective function that combines a loss term and a regularization term to control model complexity:

$$L^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t)}\right) + \sum_{k=1}^{t} \Omega(f_k)$$

XGBoost optimizes this objective using a second-order Taylor expansion, exploiting both gradient and Hessian information for efficient learning. In this study, a logarithmic transformation was applied to the target variable, and the final prediction is recovered using:

$$\hat{y}_i = e^{f(x_i)} - 1$$

Analysis

The XGBoost model was applied to the November 2024 Graph500_bfs dataset to learn nonlinear relationships affecting GTEPS performance. Eight key variables—including Memory, Problem Scale, Nodes per Core, Memory per Node, computation time (per node/core), and GTEPS (per node/core)—were selected as predictors. All variables were log-transformed and scaled to improve model stability.

The dataset was divided into 80% training and 20% testing, and the model was iteratively optimized for high explanatory power and low error. The final model achieved strong performance, with an MAE of 2,465.32, an RMSE of 2,514.24, and an R^2 score of 0.96. A 5-fold cross-validation RMSE of 0.84 further confirmed excellent generalization and minimal overfitting.

Table 1. Performance Evaluation Results

Metric	Value
Test Data MAE	2,465.32
Test Data MSE	6,321,379.29
Test Data RMSE	2,514.24
Test Data R^2 Score	0.96

The XGBoost model's learned function is expressed as a weighted sum of log-transformed features.

$$f(X) = b + w_1 \cdot \log(1+A) + w_2 \cdot \log(1+B) + w_3 \cdot \log(1+C) + \cdots + w_8 \cdot \log(1+H) \quad (5)$$

Key variables influencing GTEPS included Memory (TB), Problem Scale, Nodes per Core, and Memory per Node. Execution-time variables (C_Time) had moderate influence, while GTEPS-per-core/node served as adjustment factors. Feature importance results confirmed Memory (0.28) and Problem Scale (0.21) as the most impactful contributors to performance prediction.

Table 2. Feature Importance Results

Metric	Variable	Weights
A	Memory (TB)	0.28
B	Problem Scale	0.21
C	Nodes per Core	0.15
D	Memory per Node	0.12
E	C_Time per Node	0.08
F	C_Time per Core	0.07
G	GTEPS per Core	0.05
H	GTEPS per Node	0.04

Conclusion

The analysis confirmed that Memory capacity and Problem Scale are the most critical variables for enhancing graph processing performance, while Nodes per Core and Memory per Node also play important roles in resource-efficient parallel computation. Although computation-time variables showed lower influence, excessive execution time still led to performance degradation, indicating areas for further optimization. The model's incorporation of existing GTEPS metrics enabled more accurate predictions, demonstrating that effective performance improvement requires not only hardware expansion but also strategic resource allocation and utilization. While the study is limited to 2024 data, ongoing monitoring will be essential given the rapidly evolving nature of LLM and AI workloads.

ACKNOWLEDGMENTS
This research was supported by Korea Institute of Science and Technology Information (KISTI).(No. K25L2M1C1)