Hariprasad Thorve, 2026, 14:3
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Predictive Resource Estimation for Efficient HPC Cluster Scheduling Using Machine Learning and Slurm Simulation

Hariprasad Thorve[1], Pratik Ghadage[2], Muskan Shaikh[3], Prafull Barathe[4], Dr. R.V. Babar[5], Prof.A.P. Joshi[6]

Department of Artificial Intelligence and Data Science Suman Ramesh Tulsiani Technical Campus, Faculty of Engineering Kamshet, Pune-410405, India,

Abstract- High-performance computing (HPC) environments depend on effective resource allocation to maintain throughput, fairness, and utilization. In practice, many jobs request more memory and execution time than they consume, causing frag-mentation and delays. This paper presents a framework for predictive resource estimation using supervised machine learning, monitoring, and Slurm simulation. Historical job attributes are used to estimate actual memory demand. Linear Regression, Decision Tree, and XGBoost were evaluated. XGBoost achieved the best performance with lowest error and highest score. The study integrates prediction, observability, and simulation to improve scheduling efficiency in institutional HPC clusters.

Keywords: HPC scheduling, machine learning, resource estimation, Slurm, XGBoost.

I. INTRODUCTION

High-performance computing systems are essential for sci-entific, engineering, and data-intensive workloads that require coordinated access to CPUs, memory, storage, and intercon-nect resources to achieve high performance. Because sched-ulers must allocate resources before execution begins, their decisions rely heavily on user-specified requirements, such as the requested memory and wall-clock time. These values are often conservative rather than accurate. Users tend to over-request resources to avoid premature termination; however, this behavior creates inefficiencies across the cluster. Overallocation affects HPC systems in several ways. This reduces the scheduler's ability to pack jobs efficiently, weakens backfilling opportunities, increases queue waiting times, and can leave reserved resources idle while other jobs remain pend-ing. As clusters become more shared and workload diversity increases, improving the accuracy of pre-execution resource requests becomes increasingly important.

This study investigates a predictive framework that uses historical job data to estimate the actual memory requirements more realistically. Rather than treating prediction as an isolated machine learning exercise, this study connects three com-plementary components: resource prediction using supervised learning, operational monitoring for efficiency analysis, and Slurm simulation for safe policy-level evaluation. The resulting perspective is practical and deployment-oriented, focusing on how predictive estimates can improve the scheduler behavior and cluster utilization in real institutional environments. The overall research narrative and technical basis are derived from the uploaded project material and prior paper.

II. PROBLEM STATEMENT AND OBJECTIVES

A recurring challenge in HPC administration is the mis-match between the requested and actual resource usage. When users specify memory or runtime far above what their jobs truly consume, schedulers reserve more capacity than is nec-essary. Although this protects individual jobs from failure, it can degrade the system-wide efficiency and fairness. The central problem addressed in this study is how to estimate resource demand more accurately before job execu-tion. The main objective is to predict actual memory usage from historical job characteristics and evaluate whether these predictions can support better scheduling. To

support this goal, this study pursues four specific objectives.

III. RELATED FOUNDATION AND PROJECT BASIS

This study is grounded in two layers of source material. The first is a prior research study on the use of AI/ML methods to improve utilization in HPC and HTC environments, which established a broader motivation for data-driven optimization in supercomputing workflows. The second is the project's implementation material, including notebook-based regression experiments, supporting notes on monitoring tools, and a discussion of the Slurm simulator as a safe environment for scheduler testing.

Earlier work in this area has shown that machine learning can be effective in predicting queue wait time, execution behavior, and job resource demand. Previous studies have ex-plored ensemble learning, probabilistic scheduling models, and scheduler-integrated prediction pipelines. The current project adapts this general direction to a focused implementation cen-tered on memory estimation and scheduler support. Instead of reproducing the wait-time prediction framing of the previous study, this study emphasizes predictive resource estimation as a means to reduce over-allocation and improve job placement efficiency.

IV. PROPOSED FRAMEWORK

The framework consists of:

- Historical Data Preparation Operational job records are collected from scheduler-level logs and preprocessed to remove incomplete, inconsistent, or irrelevant entries. Only meaningful attributes related to job submission, exe-cution, and resource allocation are retained. This cleaned dataset forms the basis for regression-based prediction and scheduler interpretation.
- Machine Learning-Based Prediction Selected job at-tributes are used as input features, while actual memory usage is treated as the target variable. Supervised regres-sion models are trained to learn the relationship between job characteristics and observed resource consumption. This enables the system to generate more accurate pre-execution estimates compared to user-provided values.
- Monitoring-Based Analysis Prediction results are ana-lyzed alongside system monitoring metrics such as un-derutilized memory, CPU inefficiency, job failures, and overall cluster utilization trends. This step provides oper-ational context and helps determine whether prediction-driven adjustments effectively reduce resource wastage and improve system performance.
- Scheduler Validation Using Slurm Simulation Instead of directly applying changes to a live production cluster, the predicted resource behaviors are evaluated in a Slurm simulation environment. This allows repeated experi-mentation with scheduling policies, resource allocation strategies, and workflow configurations in a safe and controlled setting without impacting active users.

V. DATASET AND FEATURES

The implemented workflow uses scheduler-derived job fea-tures that are commonly found in HPC accounting records. These fields encode the submission behavior, allocated re-sources, and observed execution characteristics. Together, they form a compact but useful representation of memory estima-tion. The project pipeline uses the fields summarized in Table I.

TABLE I
DATASET FEATURES

Feature	Description
JobID	Unique ID
SubmitTime	Submission time
RequestedTime	Requested runtime
RunTime	Actual runtime
AllocatedProcessors	Assigned CPUs
RequestedMemory	Requested memory
UsedMemory	Actual memory

VI. METHODOLOGY

A. Data Preprocessing The dataset was first cleaned by removing missing values and excluding irrelevant

columns. An exploratory analysis was performed to understand the feature relationships and assess the correlations. A regression-ready subset was then prepared in which UsedMemory served as the target variable.

B. Train-Test Workflow The processed dataset was divided into training and testing subsets using a standard train-test split. Feature scaling was applied using StandardScaler to stabilize the model behavior across variables with different numerical ranges. In the implemented notebook, both the predictors and target-related transformations were handled consistently to support regression training.

A. Models Used

1. **Linear Regression:** This model serves as a simple base-line for further analysis. It is easy to interpret but assumes a mostly linear relationship between the inputs and memory usage.
2. **Decision Tree Regression**: This method captures non-linear splits in the data and can effectively model rule-based patterns, although it may overfit if not carefully controlled.
3. **XGBoost Regression**: Extreme Gradient Boosting is a more advanced ensemble method capable of learning complex nonlinear relationships while maintaining strong predictive performance on structured data.

B. Evaluation Metrics

Evaluation Metrics The comparison relies primarily on two indicators.

Mean Squared Error (MSE)

Coefficient score (R^2-style goodness measure reported in the notebook output)

These metrics provide a practical basis for selecting the most effective estimator among the tested models.

VII. MODEL EXPERIMENTAL RESULTS

The regression outputs from the project experiments indi-cated clear performance differences across the tested models. Among the three candidates, XGBoost provides the most fa-vorable balance of accuracy and generalization in the available implementations. Table II summarizes the results reported in the project notebook.

TABLE II
MODEL ACCURACY RESULT

Model	MSE	Score
Linear Regression	0.317202	0.660378
Decision Tree	0.146034	0.844324
XGBoost	0.136119	0.857133

XGBoost achieved the best performance among all models.

VIII. SLURM SIMULATOR

Slurm Simulator for Scheduler Validation Applying new prediction-aware policies directly to live HPC environments is risky. Changes to job submission plugins, allocation strategies, priority settings, or resource estimation rules can negatively affect active users if deployed without prior evaluation. There-fore, the Slurm simulator plays an important role in the proposed framework.

The simulator provides a controlled environment in which scheduler behaviors can be examined without disturbing the production workloads. It supports repeated testing of job submission logic, accounting flows, reservations, preemption strategies, and policy-level changes under near-realistic con-ditions. In the context of this study, the simulator is valuable because it allows predicted resource estimates to be studied as scheduling inputs rather than merely as offline predictions. For example, one can analyze how improved memory estimation might influence the queue length, backfilling efficiency, or node utilization under alternative policies. Therefore, the simu-lator acts as a bridge between notebook-level machine learning results and deployment-oriented scheduler experimentation.

IX. MONITORING

Prediction alone does not fully explain why an HPC cluster is underutilized in the real world. A cluster may suffer from poor efficiency owing to excessive

memory reservation, low CPU usage, user estimation errors, job failures, misconfigured policies, or infrastructure issues. Monitoring tools help distin-guish between these causes.

The project's monitoring perspective emphasizes the use of dashboards and observability platforms, such as Grafana, Prometheus, InfluxDB, and ElasticSearch, to track job effi-ciency patterns. These tools can expose wasted CPUs, un-derused memory allocations, failed jobs, timeout trends, and aggregate utilizations across users or queues. When combined with predictive resource estimation, monitoring becomes espe-cially useful because it helps administrators to verify whether the predicted corrections correspond to measurable operational gains. In other words, monitoring provides a feedback loop necessary to transform the model output into scheduler-level decisions and policy tuning.

Therefore, a practical deployment strategy would include not only a prediction service but also an efficiency dashboard that continuously compares requested, predicted, and actual usage over time.

X. RESULTS DISCUSSION

The experimental results indicate that machine learning can narrow the gap between user-requested memory and actual job usage. Although the evaluated score values are moderate rather than extremely high, the practical value of the model lies in directional improvement: it consistently moves estimates closer to real consumption than static user requests in several sample cases.

The superiority of XGBoost in this implementation suggests that memory demand patterns are not purely linear and are likely to arise from interactions among runtime behavior, pro-cessor allocation, and user-request characteristics. The weaker behavior of Linear Regression supports this interpretation, whereas the instability of the Decision Tree model reflects the limitations of using a single-tree estimator on structured workload data.

From a scheduler perspective, the most important implica-tion is not merely numerical error reduction but also improved allocation realism. If memory reservations are closer to the actual need, the scheduler can potentially fit more jobs onto the available nodes, reduce the idle reserved capacity, and improve the fairness of the queued workloads. This is particularly important in academic clusters, where shared access, limited hardware budgets, and bursty demand often make efficient scheduling a major operational concern.

The results also reinforce the importance of integrating predictions with observability and simulations. A good model without monitoring cannot explain the broader utilization picture, and a good model without scheduler testing cannot be safely operationalized in clinical practice. Therefore, the combined workflow is more valuable than any one component in isolation.

TABLE III
PREDICTION MODELS RESULT

Data	Test1	Test2	Test3
Actual Values	409600.0	409600.0	409600.0
Requested Memory	972800.0	972800.0	409600
Linear Regression	409369.5	410062.8	410062.8
Decision Tree	409600.0	409600.0	40600.0
XGBoost	409599.3	409598.6	409604.4

XI. ADVANTAGES OF THE PROPOSED APPROACH

The proposed resource estimation framework offers several practical advantages.

First, it supports pre-execution resource estimation using fields already available in common scheduler logs, making it suitable for real HPC workflows without requiring sophisti-cated instrumentation.
Second, it reduces dependence on conservative manual resource requests, which are a major cause of memory over-allocation in shared clusters.

Third, it connects prediction to monitoring and simulator-based scheduler analysis, enabling safer and more evidence-based administrative decisions.

Fourth, it is especially relevant for campus and institutional clusters, where production experimentation must be cautious, yet efficiency improvements can significantly improve access and throughput for multiple users.

XII. LIMITATIONS AND FUTURE SCOPE

This study is based on the available project implementation and supporting source materials rather than full production-scale deployment. The reported model comparison is limited to the experiments already executed in the supplied notebook, and the evaluation does not yet include large-scale temporal validation, feature importance ranking, or cross-system gener-alization studies.

Several directions can be taken to strengthen this work in the future. The first is to expand the dataset and include a broader accounting history across more workload types. The second is to extend the prediction beyond memory usage to include wall-clock runtime estimation, as inaccurate time requests also strongly affect scheduling quality. Third, feature importance analysis and cross-validation were performed to better understand the model robustness. Fourth, prediction should be integrated into an actual job submission workflow with monitoring-backed feedback and simulation-driven policy tuning. Additional models, including random forest ensembles, temporal learning approaches, and hybrid prediction systems, may also further improve performance.

XIII. CONCLUSION

This study presented a predictive resource estimation frame-work aimed at improving scheduling efficiency in high-performance computing (HPC) environments. The experimen-tal results clearly indicate that XGBoost Regression outper-forms the other evaluated models, achieving the lowest mean squared error and the highest R^2 score among the tested ap-proaches. Decision Tree Regression also demonstrated strong performance, while Linear Regression served as a baseline with comparatively lower accuracy.

TABLE IV
MODEL PERFORMANCE (R2 %)

Model	**Score (R^2)**
Linear Regression	0.6603
Decision Tree Regression	0.8443
XGBoost Regression	**0.8572**

When compared with other regression models such as Las-soLarsIC, ElasticNetCV, and Ridge Regression, the proposed models achieved significantly higher predictive performance. This highlights the effectiveness of advanced and ensemble-based methods in capturing the nonlinear patterns present in HPC workload data.

TABLE V
OTHER RESEARCH MODEL PERFORMANCE (R2 %)

Model	**Score (R^2)**
LassoLarsIC Regression	0.1736
ElasticNetCV Regression	0.1723
Ridge Regression	0.1734

Furthermore, the sample predictions show that the estimated memory values are much closer to the actual usage than the user-requested allocations, which are often overestimated. This improvement has important practical implications, including better resource allocation, reduced wastage, improved schedul-ing decisions, and enhanced overall cluster utilization.

By integrating predictive modeling with monitoring and Slurm-based simulation, the proposed approach provides a practical and reliable pathway for improving HPC schedul-ing efficiency in academic and institutional environments. It also enables safer evaluation of scheduling strategies without affecting production systems.

Overall, this work lays a strong foundation for intelligent and adaptive HPC resource management. Future enhancements may include extending the approach to runtime prediction, incorporating larger datasets, and integrating the model into real-time scheduling systems for further optimization.

Acknowledgment
The authors thank the department and institute for support.

REFERENCES

1. Paokin, A.V., Nikitenko, D.A. "Approbation of Methods for Supercom-puter Job Queue Wait Time Estimation." (2023)
2. Jancauskas, V., Piontek, T., Kopta, P., Bosak, B. "Predicting Queue Wait Time Probabilities for Multi-Scale Computing." (2019)
3. Vercellino, C., Scionti, A., Varavallo, G., Viviani, P., Vitali, G., Terzo, O. "A Machine Learning Approach for an HPC Use Case: Jobs Queuing Time Prediction." (2023)
4. Okafor, N.U., Lusch, B., Vishwanath, V. "Queue Wait Time Prediction in High Performance Computing (HPC) Systems." (2022)
5. Krakov, D., Feitelson, D.G. "High-Resolution Analysis of Parallel Job Workloads." (2017)
6. Pathak, A.R. "Highlights on Utilizing Machine Learning for High Performance Computing Systems." (2021)
7. Dunn, B. "Optimizing High Performance Computing Systems: Resource Utilization and Throughput by Leveraging Machine Learning." (2022)
8. Nissimov, A., Feitelson, D.G. "Probabilistic Backfilling." (2019)
9. Tanash, M., Dunn, B., Andresen, D., Hsu, W., Yang, H., Okanlawon, A. "Improving HPC System Performance by Predicting Job Resources via Supervised Machine Learning." (2021)
10. HUJI Parallel Workloads Archive. "HPC2N Seth Workload Log." https://www.cs.huji.ac.il/labs/parallel/workload/lhpc2n/