Job Grouping Based Intelligent Resource Prediction Framework

BY BESTE OZTOP[1], BENJAMIN SCHWALLER[2], VITUS J. LEUNG[2], JIM BRANDT[2], BRIAN KULIS[1], MANUEL EGELE[1], AYSE COSKUN[1]

[1]Boston University; [2]Sandia National Laboratories

Job Scheduling Strategies for Parallel Processing (JSSPP)
June 3, 2025

Introduction

```
#!/bin/bash -l
#$ -P peaclab-mon
#$ -N my_job_name
#$ -o ./my_job_output.out
#$ -l mem_per_core=18G
#$ -pe omp 28
#$ -l gpu_c=6.0
#$ -l gpus=2
#$ -l h_rt=4:00:00
python my_python_code.py
```

slurm workload manager

grid engine

FUJITSU

Altair | PBS Works™

- Allocates resources based on user demand.
- Resources are not released until the job is completed.

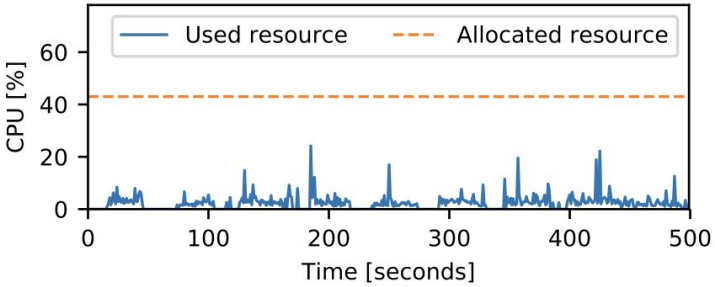
Underprediction

- Early job terminations
- Reduced QoS of users

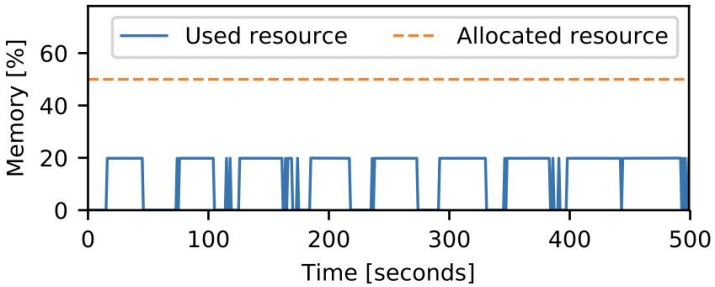
Overprediction

- Job backfilling inefficiencies
- Increased wait time
- Valuable computing and memory resources sitting idle

Introduction

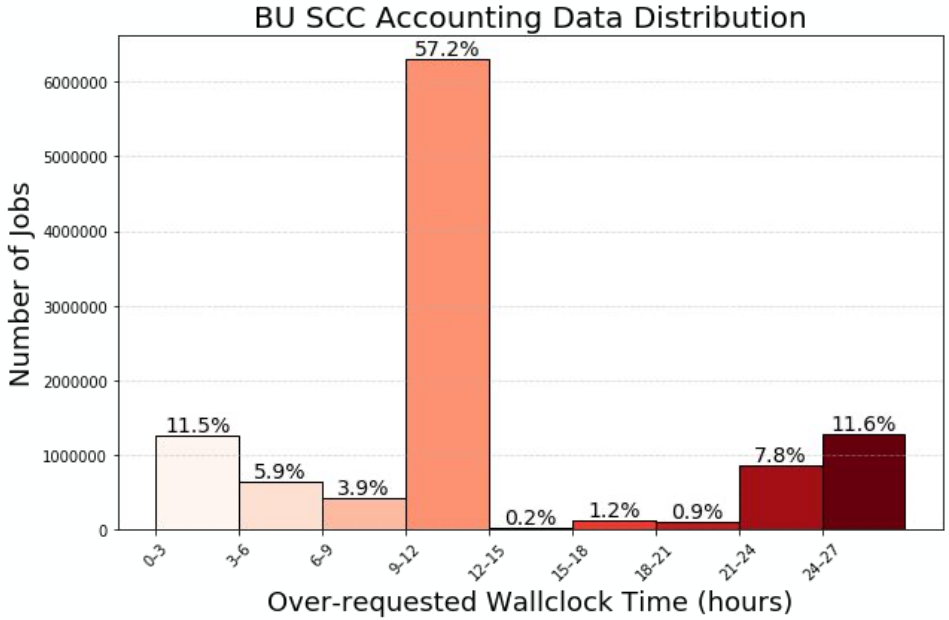


(a) Comparison of CPU allocation and usage



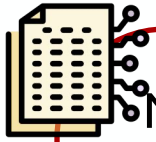
(b) Comparison of memory allocation and usage

[Thonglek et al., CLUSTER'19]



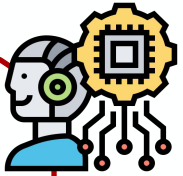
Users tend to request more resources than they need!

State-of-the-Art of Resource Prediction



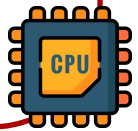
Maximum memory size and execution time prediction with regression models

[Tanash et al., PEARC'21]



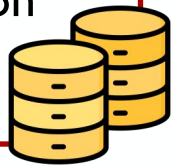
Predicting CPU and memory requirements of batch jobs with an LSTM model

[Thonglek et al., CLUSTER'19]



Using an XGBoost regressor for execution time prediction

[Menear et al., PEARC'23]



Utilizing Tobit Model for eliminating execution time underprediction

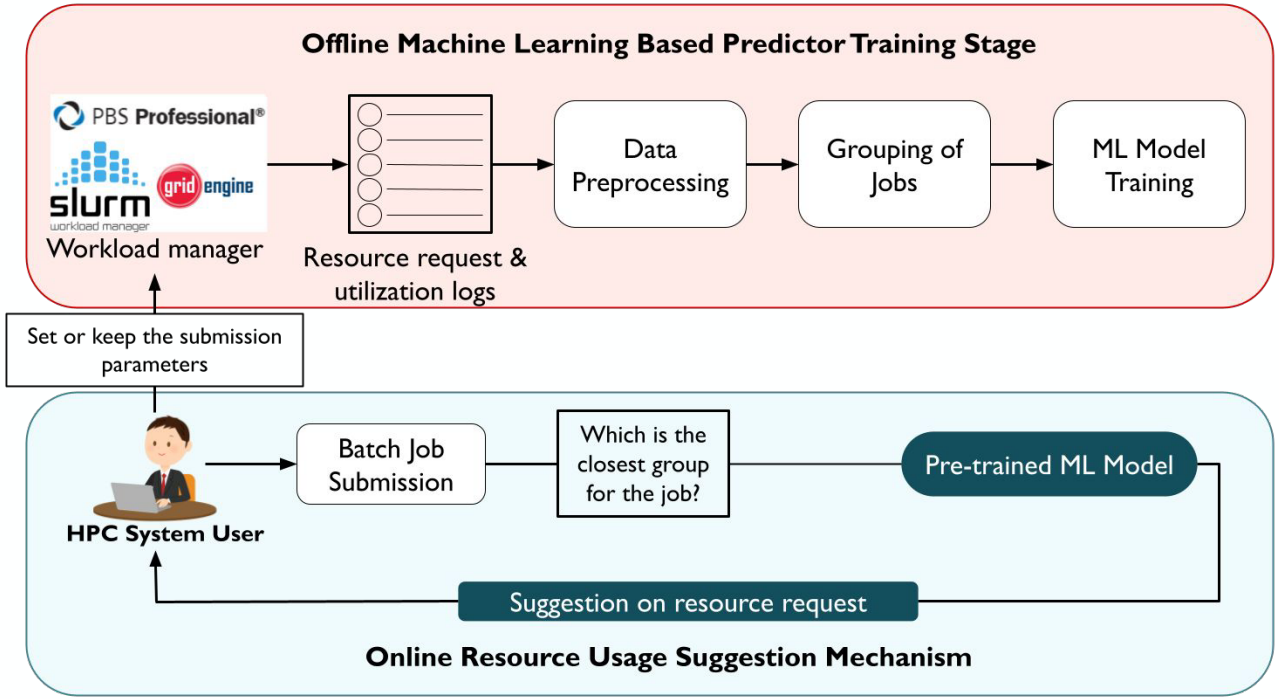
[Fan et al., CLUSTER'17]

State-of-the-Art of Resource Prediction

Limitations:

- Not analyzing overpredictions
- Not addressing underpredictions
- Using resource utilization as an input feature
- Not looking at the similar batch jobs for prediction

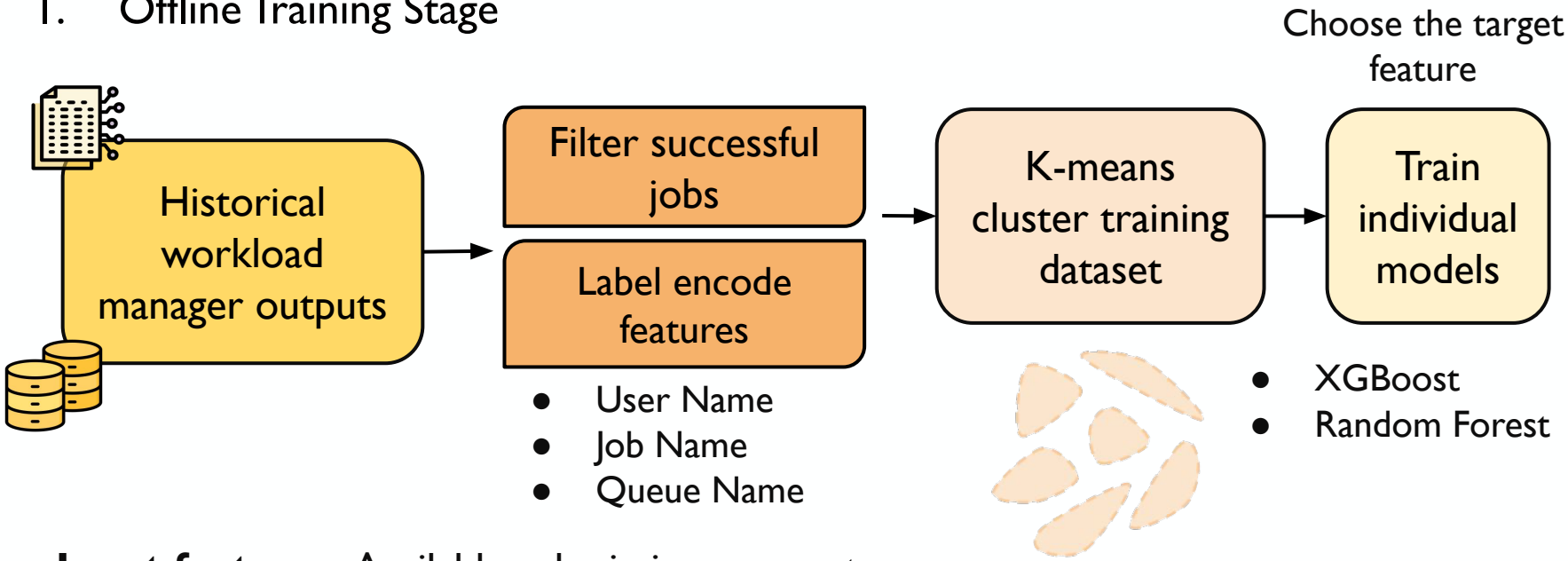
Proposed Framework



Goal: Provide a job grouping-based intelligent framework for HPC batch job resource provisioning to reduce overpredictions and minimize underpredictions.

Framework Design

1. Offline Training Stage



Input features: Available submission parameters

Target feature: Execution time, maximum memory size or number of processors

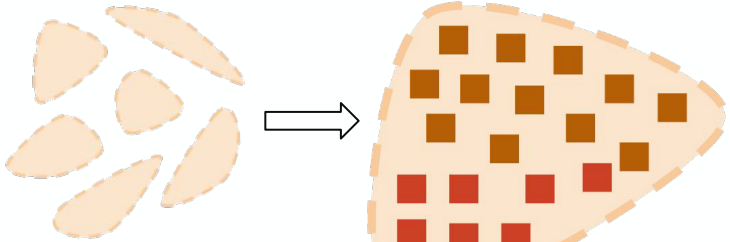
Handling Underpredictions

Problem: Using regression models alone result in 40 to 50% underprediction ratio, which will lead to early job terminations.

→ We use data-driven mechanisms to eliminate the early job terminations in our recommendation framework.

Handling Underpredictions

1. Calculating buffer values per cluster in the offline training stage

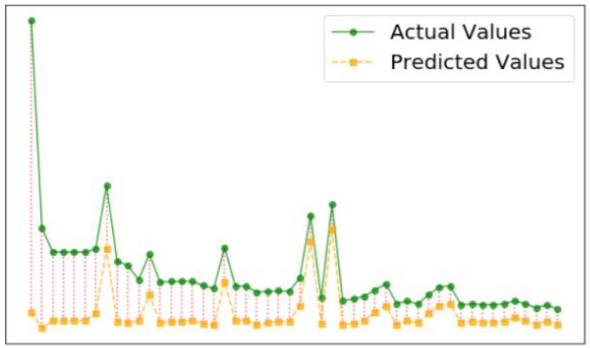


□: **Batch job log outputs**

Training set
Training individual models

Validation set
Making resource predictions

Find **2σ** value of the underprediction amounts



σ = standard deviation

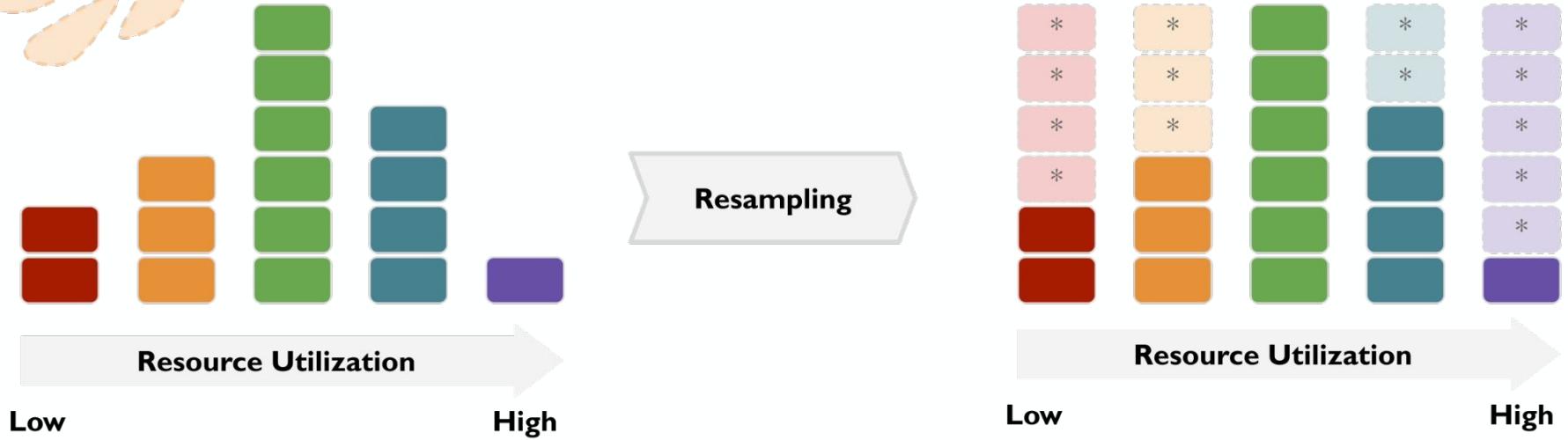
Testing Stage

Add **buffer** value to model predictions

Handling Underpredictions

2. Changing the training dataset distribution by resampling less frequent jobs

Imbalance in bin sizes can bias regression models toward the majority.



Framework Design

2. Testing Stage

```
#!/bin/bash -l
#$ -P peaclab-mon
#$ -N my_job_name
#$ -o ./my_job_output.out
#$ -l mem_per_core=18G
#$ -pe omp 28
#$ -l gpu_c=6.0
#$ -l gpus=2
#$ -l h_rt=4:00:00
python my_python_code.py
```

Find the corresponding cluster

Use that cluster's model to make predictions

slurm workload manager

grid engine

FUJITSU

Altair | PBS Works™

Real world HPC system resource requests and utilization

Experimental Procedure

Dataset Name	Duration	Number of Jobs
NREL Eagle	November 2018 to February 2023	7,388,190
Fugaku	April 2024	338,79
M100	December 2021	79,173
BU SCC	January-December 2023	13,244,729
Sandia	May-October 2024	313,615

Input features: Available submission parameters for each dataset
Target features: Execution time, maximum memory size or number of processors

Experimental Procedure

Prediction Models

Baseline Method: XGBoost Regression Model [Menear et al., PEARC'23]

XGB

RF

Clustering Similar Jobs (Multiple Model Training)

Resampling Train Data for Each Cluster

Adding 2σ

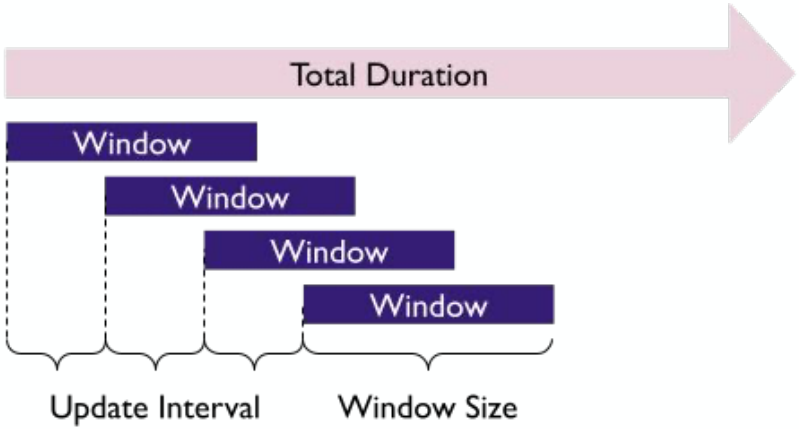
Comparison between 10 models

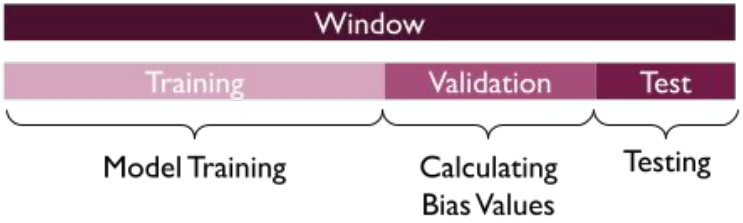
- Underprediction Ratio
- Overestimation Factor

$$\text{Overestimation Factor (OF)} = \frac{\text{Requested Resource of Predicted Utilization}}{\text{Actual Resource Utilization}}$$

[Menear et al., PEARC'23]

Train-Test Time Duration Experiment

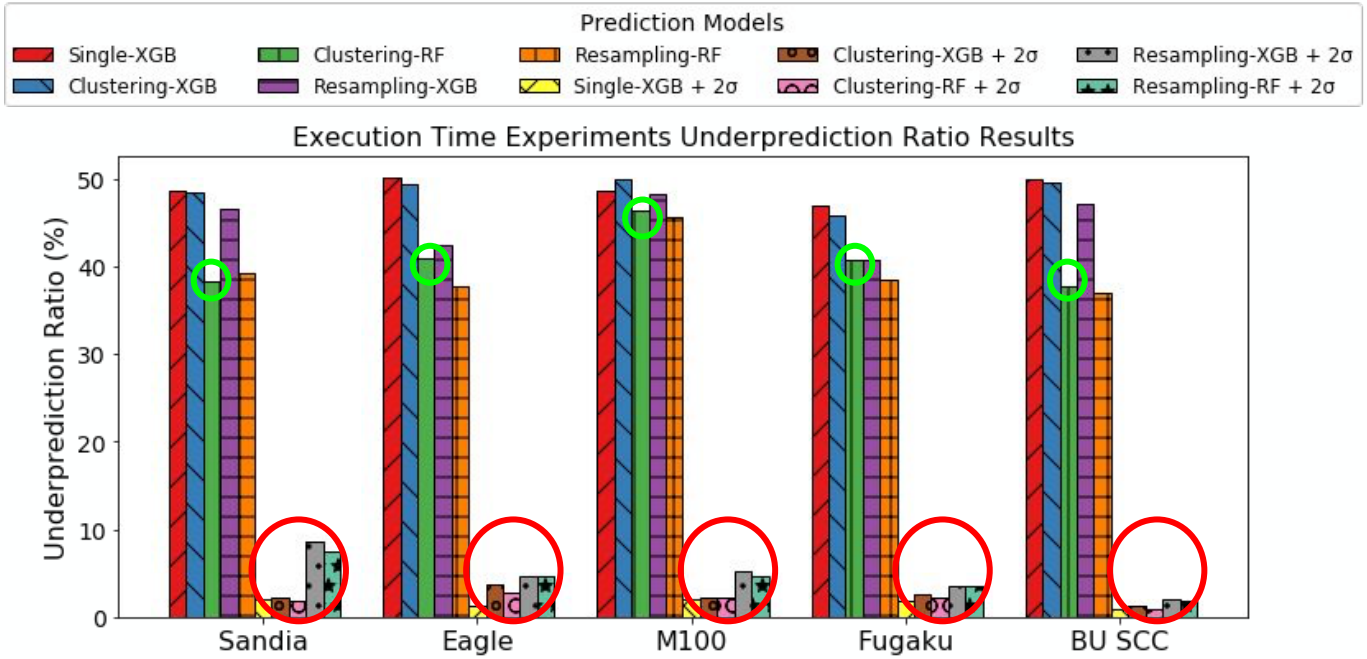




Observing the underprediction in execution time prediction

Dataset	Window Size	Update Interval
NREL Eagle	1 month	1 month
Fugaku	3 days	3 days
M100	3 days	3 days
BU SCC	2 months	1 month
Sandia	1 week	1 week

Execution Time Prediction Experiment

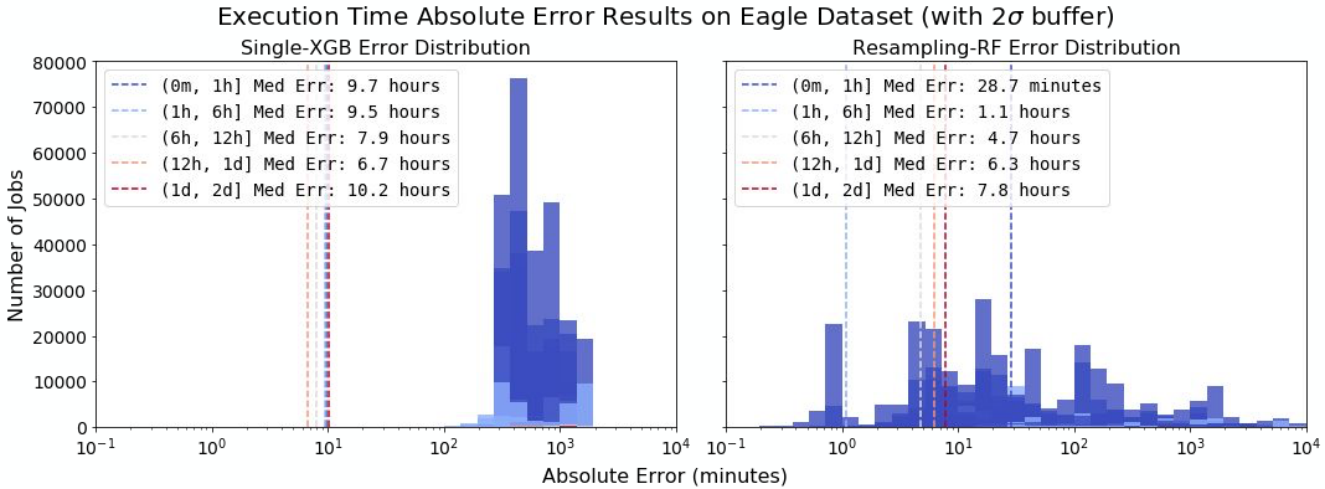


➔ Clustering method with Random Forest regression model improves and adding buffer value further reduces the underprediction rate.

Execution Time Prediction Experiment

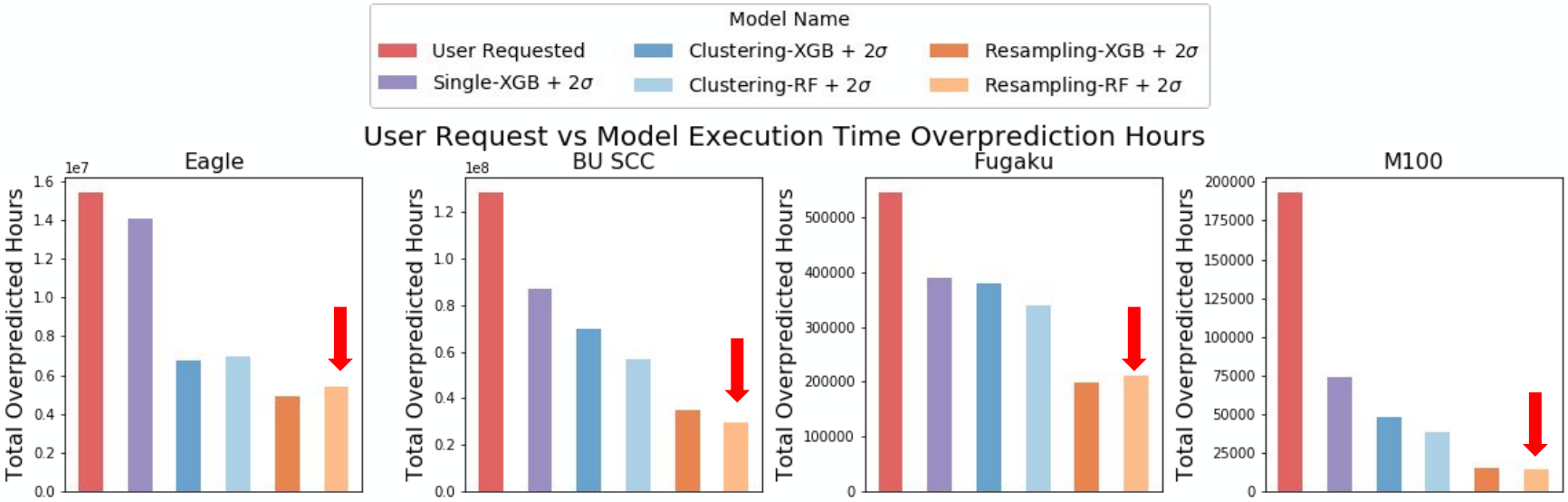
Method	Sandia	Eagle	M100	Fugaku	BU SCC
Single-XGB	48.59%	50.16%	48.59%	46.91%	49.87%
Resampling-RF + 2σ	7.39%	4.65%	4.75%	3.57%	1.82%

Our framework offers lower underestimation rate.



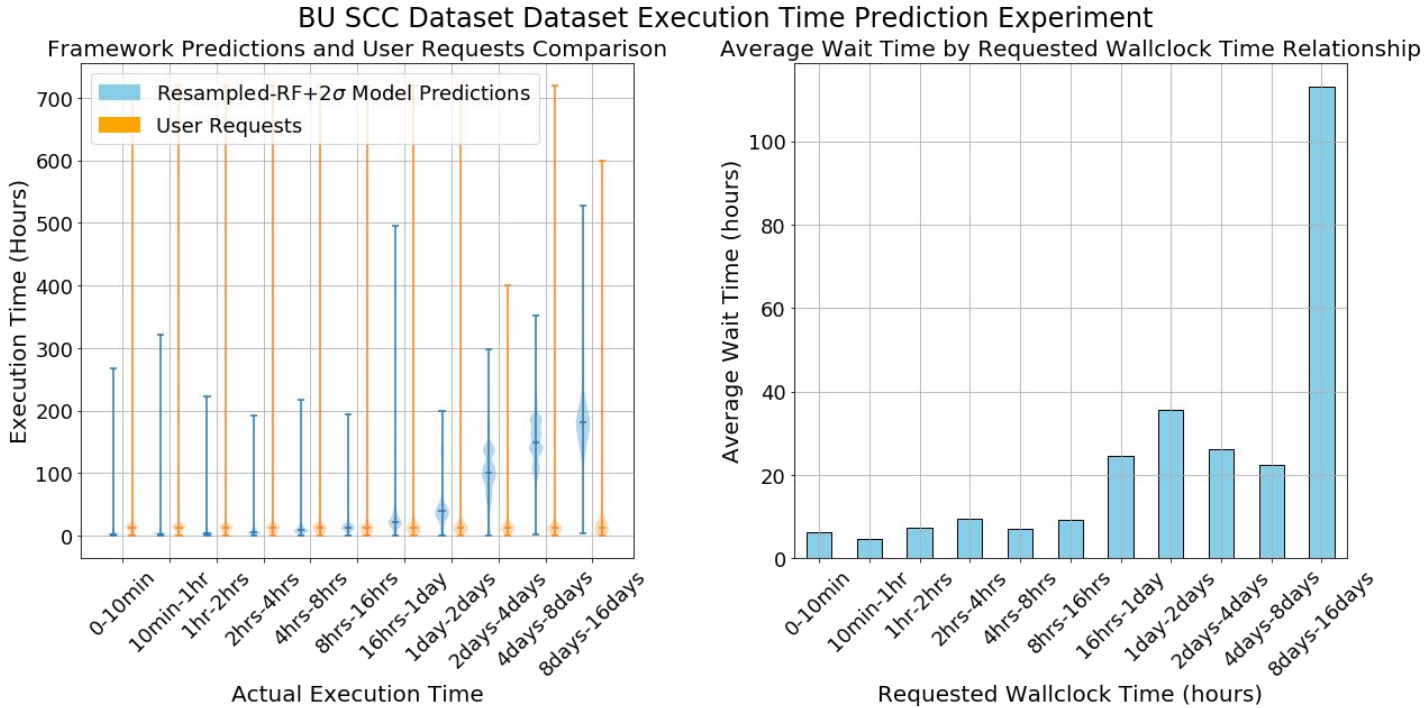
➔ Resampling strategy reduces the mean absolute error in execution time prediction compared to the baseline method.

Execution Time Prediction Experiment



➔ While reducing the underpredictions, we also limit the overpredictions of execution time.

Execution Time Prediction Experiment



→ We can reduce the job wait time compared to the users' over requests in wallclock time.

Predicting Other Resource Types

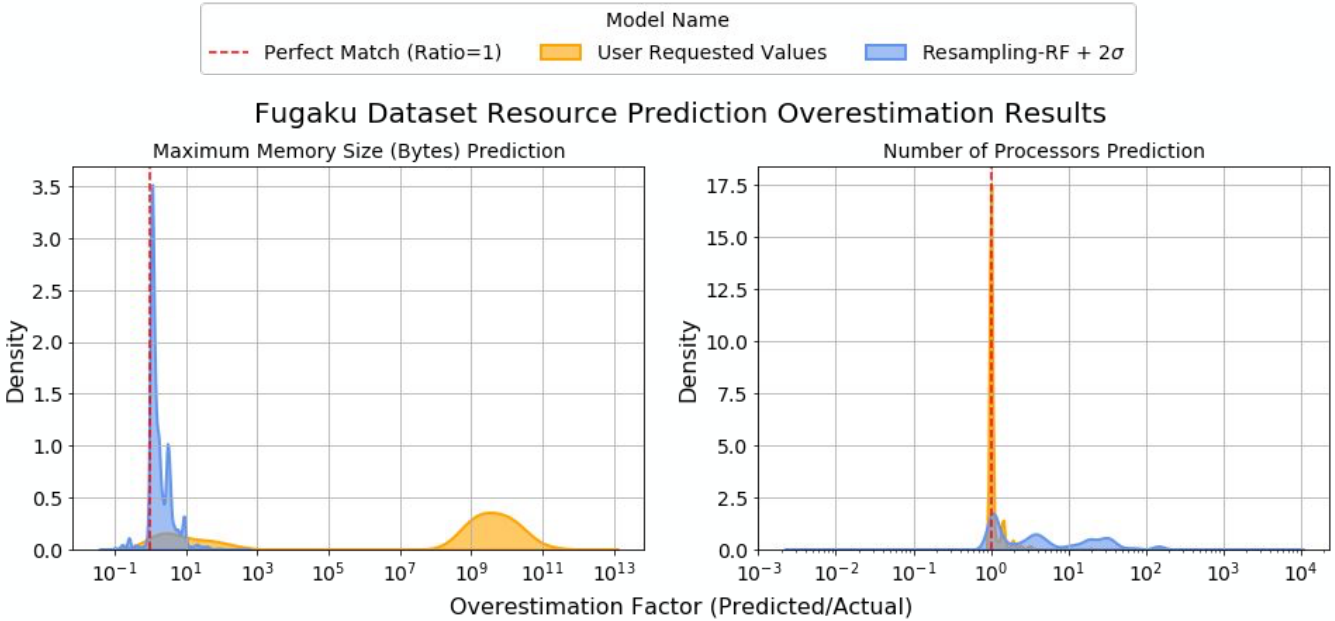
CPU Prediction: Number of processors required for the batch job.

Maximum Memory Size Prediction: `max_rss` value from the workload manager.

Dataset	Fugaku		Sandia	
Predicted Resource Type	CPU	Max Memory	CPU	Max Memory
Single-XGB	25.78%	49.87%	37.42%	48.31%
Resampling-RF + 2σ	2.41%	6.70%	10.32%	9.74%

→ Our framework offers lower underestimation rates for other resource types' predictions as well.

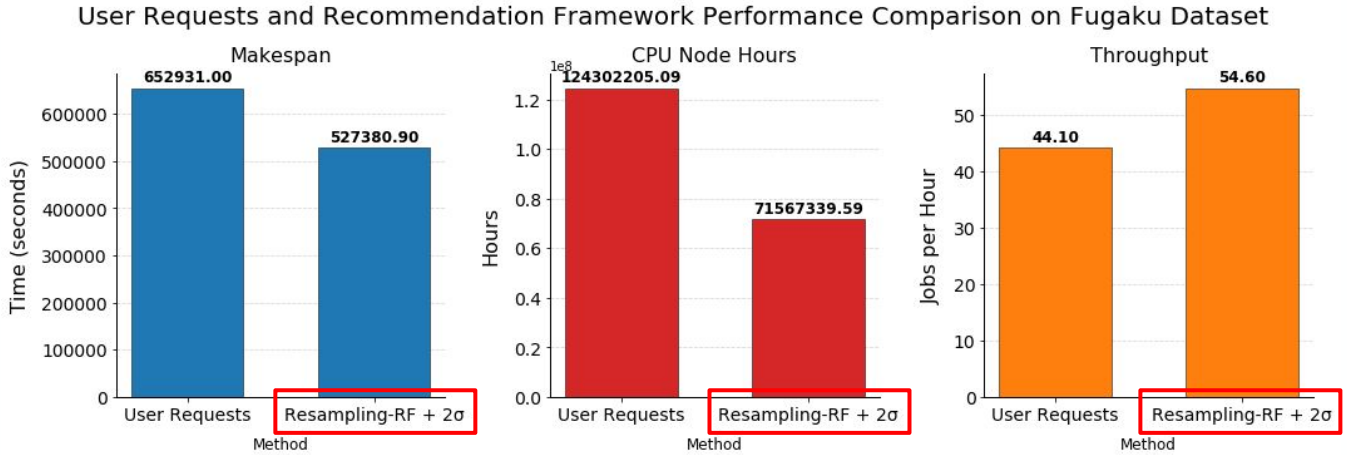
Predicting Other Resource Types



- Fugaku users' overestimation of maximum memory size reaches up to 10^{11}.
- Utilizing an intelligent resource recommender can help reducing the idle memory size.

Future Work

- Using workload manager simulators to assess the framework predictions



Simulation Results from CQSim [Yang et al., SC'13]

- Deploying the framework on a real large scale computing system

IPDPS 2025 • Milano, Italy

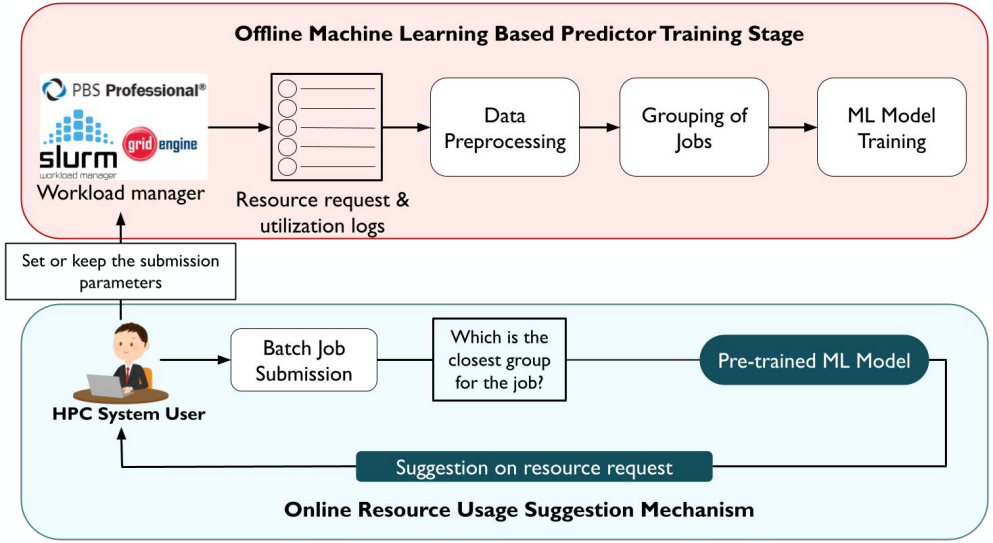
JSSPP

PEACLAB

Sandia National Laboratories

BOSTON UNIVERSITY

- **Intelligent, data-driven models** help with
  - Minimizing under- and over-estimation of batch job resources
  - Improving efficiency and resource utilization in large-scale computing environments
- Our **recommendation framework** supports HPC users to make informed decisions for job submissions



CONCLUDING REMARKS

For questions and feedback: boztop@bu.edu