Bring your own Poster Name: Custom Cost, Loss, And Reward Functions to Train Regression Models for Estimating Execution Resources using HARP.
Team: Swathi Vallabhajosyula (OSU), Rajiv Ramnath (OSU), Joe Stubbs (TACC)

Abstract:
High-performance computing (HPC) is crucial in executing resource-intensive scientific workflows such as genome sequencing, weather predictions, and deep neural network (DNN) training. However, achieving optimal resource utilization during job execution requires selecting the right execution endpoint based on specific resource requirements. Different clusters, such as those at OSC or TACC, have diverse architectures, allocation guidelines, and billing policies, making it challenging for users to fine-tune resource allocations for their applications. To address this issue, we introduce the HARP (HPC Application Resource [runtime] Predictor) solution as part of the ICICLE project (AI4CI). ICICLE aims to democratize AI by creating a national infrastructure that fosters interdisciplinary collaboration, advances foundational AI research, and promotes diversity and inclusion, ultimately enabling transparent and trustworthy AI solutions for addressing societal challenges and national priorities. The AI4CI thrust develops research-based solutions that build artificially intelligent models for smartly optimizing cyberinfrastructure.

HARP enables application profiling and recommends optimal resource allocation configurations, reducing execution costs without compromising workflow execution times. In our previous papers [1-3], we demonstrated the architecture of HARP and its principal components. Now, we are exploring possible extensions for our preliminary framework in the following dimensions:

1. Loss Function: Traditional regression models use error functions like mean square error (MSE) or mean absolute error (MAE) to fit a model against training data. However, for resource allocations, underestimations are more expensive than overestimations since they can lead to job termination. We propose a new loss function that biases models towards underprediction, aiming to overestimate resource requirements to avoid job failures.
2. Metric System: Current evaluation metrics such as MSE, root mean square error (RMSE), or mean absolute percentage error (MAPE) treat under and overestimations equally, potentially ranking systems inaccurately. To address this, we propose a new metric that prioritizes systems reducing underpredictions while minimizing overestimations, offering a more balanced ranking of estimators.
3. Cost Function: To penalize the regression model for underestimating resources, we propose a reward system based on job execution status and re-execution costs. We simulate a "black box re-allocation policy" by doubling the execution time every time a job fails to execute with the suggested allocations. This policy expands available and expandable resources like cores, memory, or walltime until the job successfully executes.
4. Memory Modelling: Memory modeling is a crucial aspect when training deep neural network (DNN) models, as it relies on various factors such as model training parameters, optimizers, and training data. The ZeRO paper [4] introduces a straightforward formula for estimating model parameters and memory optimizations in distributed training. We aim to leverage this equation to estimate memory requirements and identify a suitable system configuration capable of accommodating and effectively training the model from the available options.

To our understanding, we are the innovative developers of an initial framework that begins by simulating executions to build regression models. Diverging from numerous prior methods that relied on queue history and requested allocations to estimate walltime, our approach centers on

capturing consumption and allocations based on precise criteria such as memory requirements. Importantly, our method, which records consumption post-emulation through the SLURM manager, introduces no extra overhead when profiling applications. The HARP code is available for download and installation on Linux-based systems. The newest release of HARP is designed to enable API-based central storage and integration with TAPIS.

References

[1] Vallabhajosyula, Manikya Swathi, and Rajiv Ramnath. “Towards Practical, Generalizable Machine-Learning Training Pipelines to build Regression Models for Predicting Application Resource Needs on HPC Systems.” Practice and Experience in Advanced Research Computing. 2022. 1-5.
[2] Vallabhajosyula and R. Ramnath, “Establishing a Generalizable Framework for Generating Cost-Aware Training Data and Building Unique Context-Aware Walltime Prediction Regression Models,” 2022 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), Melbourne, Australia, 2022, pp. 497-506,
[3] Vallabhajosyula, Manikya Swathi, and Rajiv Ramnath. “ Insights from the HARP Framework: Using an AI-Driven Approach for Efficient Resource Allocation in HPC Scientific Workflows.” Practice and Experience in Advanced Research Computing. 2023
[4] S. Rajbhandari, J. Rasley, O. Ruwase and Y. He, "ZeRO: Memory optimizations Toward Training Trillion Parameter Models," SC20: International Conference for High Performance Computing, Networking, Storage and Analysis, Atlanta, GA, USA, 2020, pp. 1-16, doi: 10.1109/SC41405.2020.00024.
[5] HARP github page: https://github.com/ICICLE-ai/harp