Supervised Learning for Parallel Application Performance Prediction

Andrew Titus[1,2] and Abhinav Bhatele[2] (advisor)

[1]Department of Electrical Engineering and Computer Science - Massachusetts Institute of Technology,
[2]Center for Applied Scientific Computing, Lawrence Livermore National Laboratory

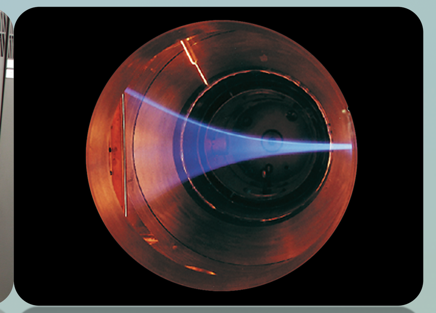
Abstract

We evaluate supervised machine learning methods as tools for prediction of communication time of large parallel applications. Through these methods, we correlate time for different task mappings to the corresponding network hardware counters in the context of two production applications, MILC and pF3D. The results from these machine learning regression algorithms are used to gain insight into the relative importance of different hardware counters or metrics for predicting application performance.

Experimental Approach

I. Generate task mappings, both with random ordering of MPI ranks and with *Rubik*, a mapping tool developed at LLNL [1]

II. Run *pF3D* and *MILC* on 1K and 4K nodes of *Vulcan*, a LLNL-hosted IBM Blue Gene/Q supercomputer, using these mappings to generate network counter data



III. Run 6 different supervised machine learning regression algorithms, provided by the *Scikit-learn* Python package, on the generated network counter data

IV. Evaluate scalability and effectiveness of predictions by these models

Regression Algorithms Used

Decision Trees

Gradient Boosted Regression Trees (GBRT)

Randomized Forests of Decision Trees

Ridge Regression

Bayesian Ridge Regression

Support Vector Machines (SVM)

Model Inputs

15 different *communication features*, derived from hardware network counters, are used as inputs. These are based upon three main categories of metrics:

Bytes passing between nodes on network

Buffer size of network routers

Queue time of data packets on routers

Evaluation of Prediction Success

Success of a model here is determined by *Rank Correlation Coefficient (RCC)*, a metric that relates the number of correctly predicted pair orderings of performance rankings to the actual pair orderings of performance rankings [2]

Formally speaking, if observed ranks of tasks mappings are given by $\{x_1, x_2, \cdots, x_n\}$, and the predicted ranks by $\{y_1, y_2, \cdots, y_n\}$, we define RCC as:

$$concord_{ij} = \begin{cases} 1, & \text{if } x_i >= x_j \ \& \ y_i >= y_j \\ 1, & \text{if } x_i < x_j \ \& \ y_i < y_j \\ 0, & \text{otherwise} \end{cases}$$

$$RCC = \Big(\sum_{0<=i<n} \sum_{0<=j<i} concord_{ij} \Big) / \big(\frac{n(n-1)}{2} \big)$$

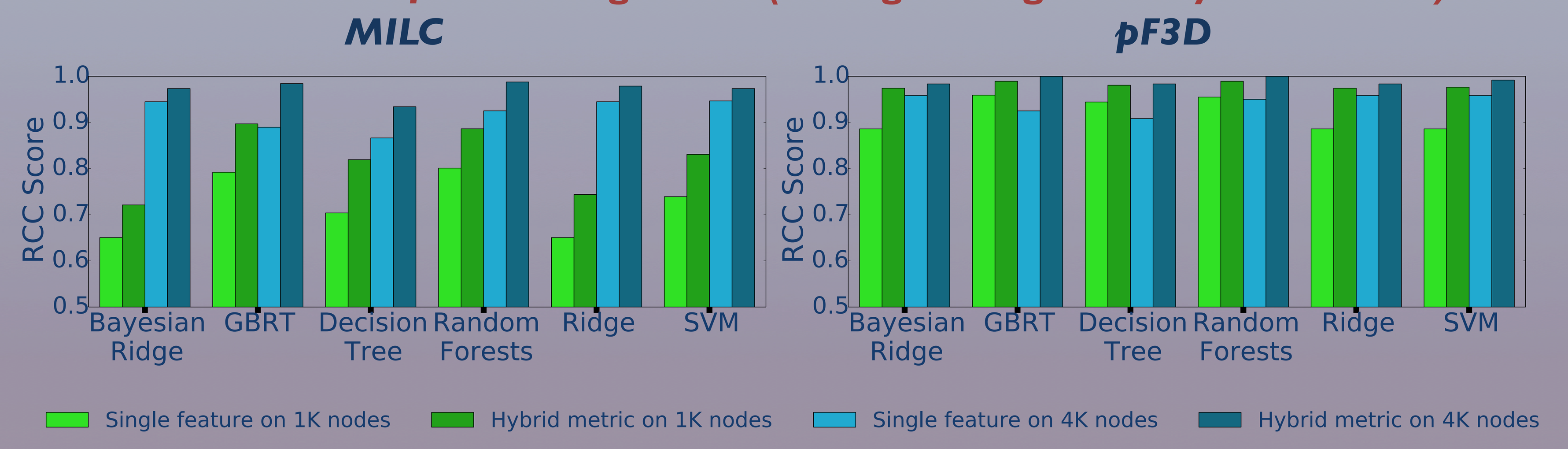
Simplified Example

Mapping	Actual	Predicted
1	10s	15s
2	15s	20s
3	20s	15s
4	25s	30s

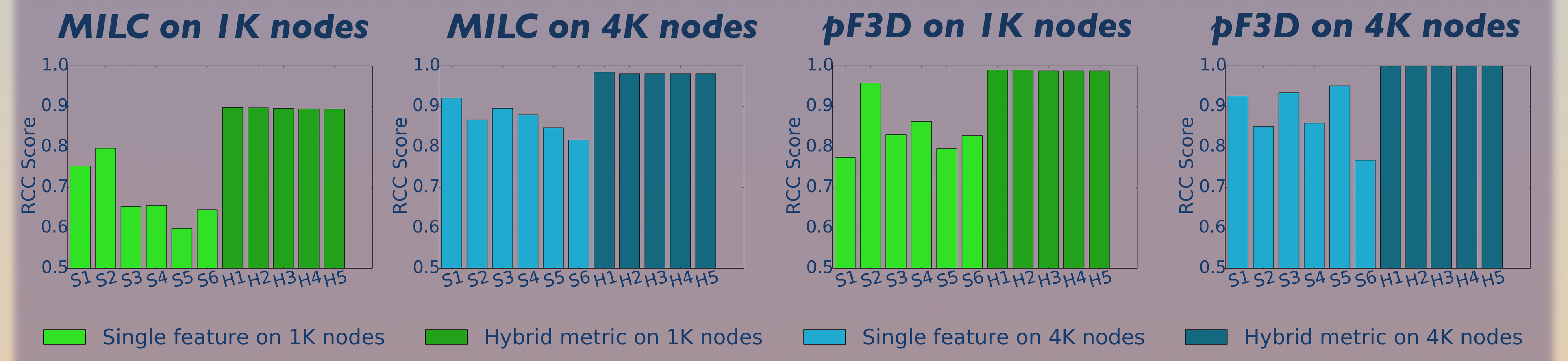
RCC Value:
5 correct orderings/ 6 pair orderings = .833

Single Metric: Predictions that use only one communication feature as input
Hybrid Metric: Predictions that use a combination of communication features as input [2]

Best RCC Values for each algorithm (among all single and hybrid metrics)



Best RCC Values for single and hybrid metrics using GBRT

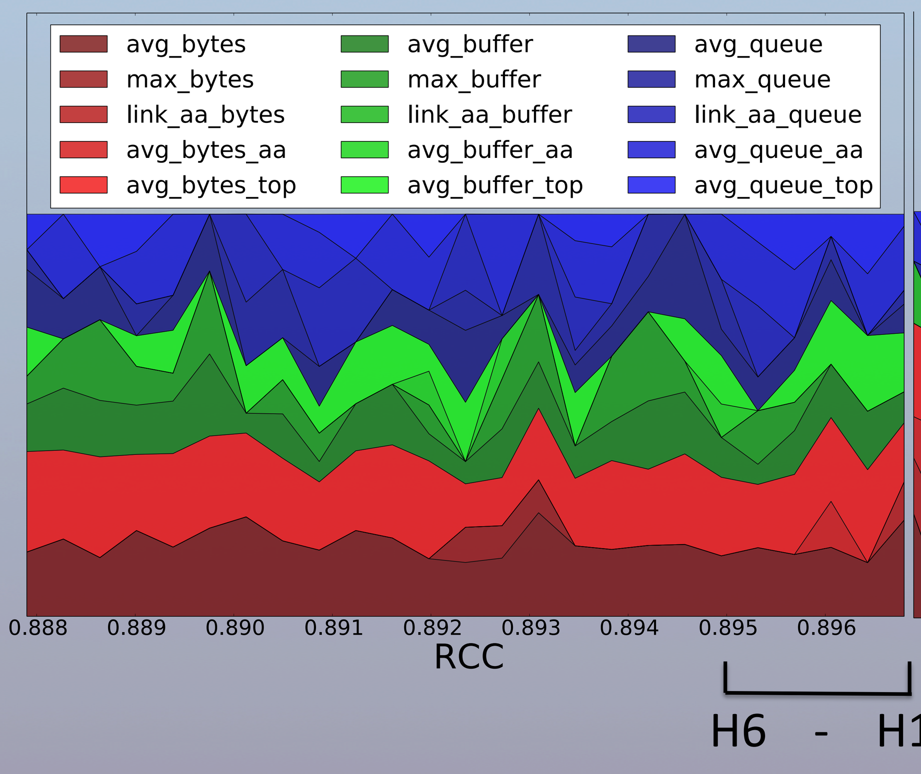


S1: average bytes
S2: maximum bytes
S3: average buffer
S4: maximum buffer
S5: average queue
S6: maximum queue
H1-H6: best hybrid metrics per application and node count

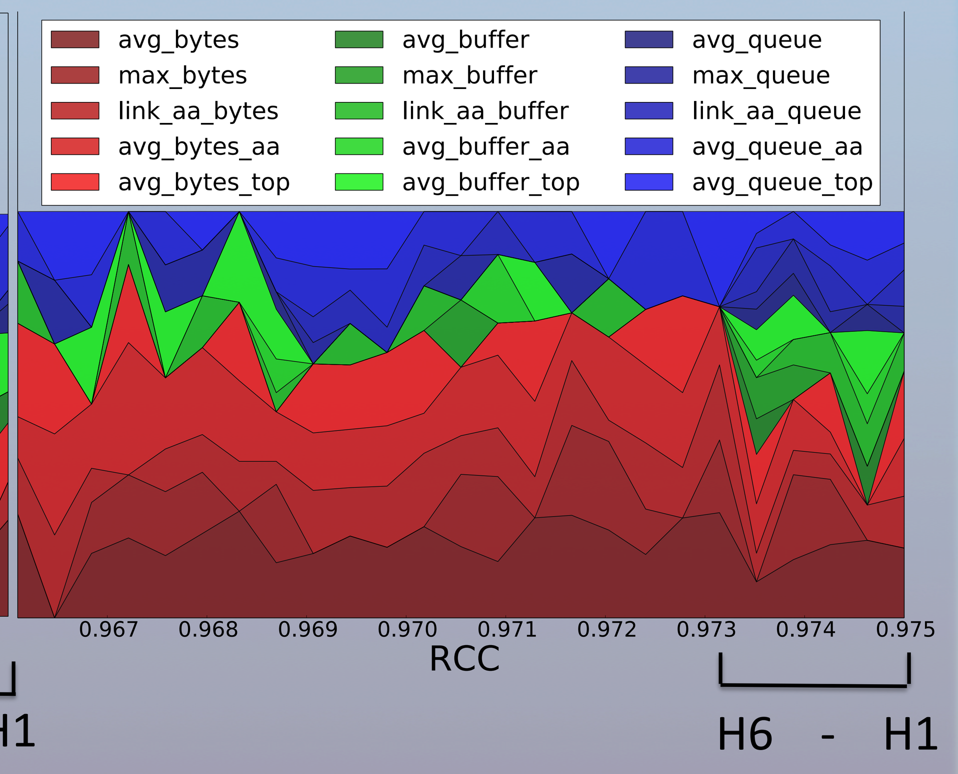
Feature Importance Analysis

Analysis of the relative importance of communication features used as inputs in the best hybrid metrics can provide insight into the causes of network congestion

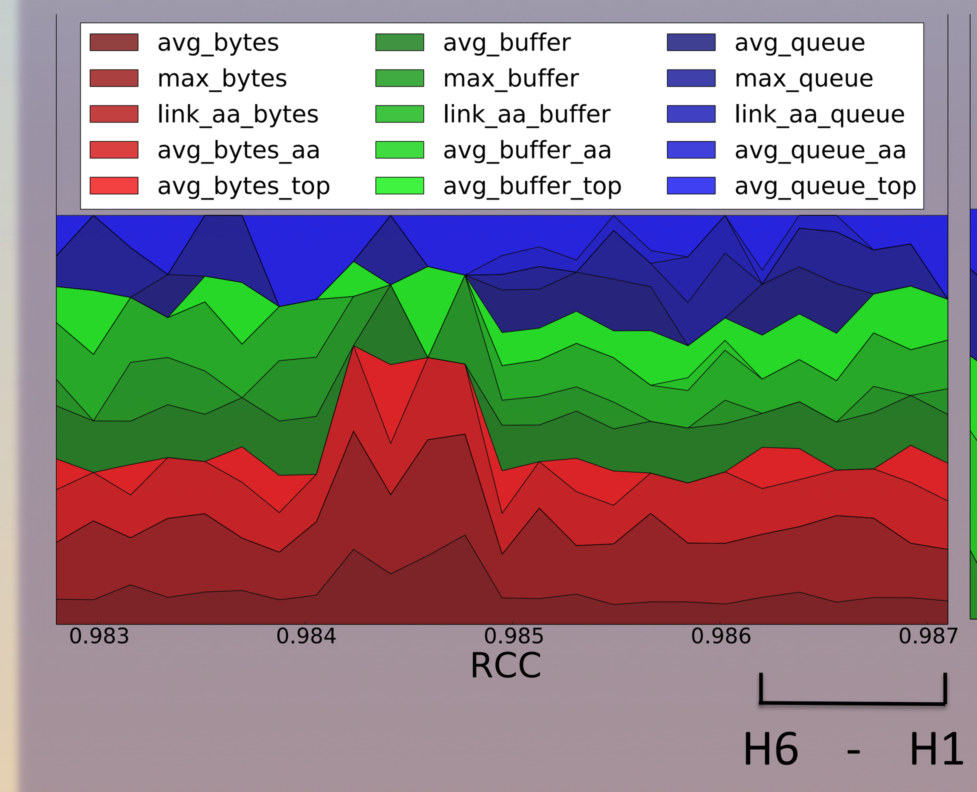
MILC on 1K nodes



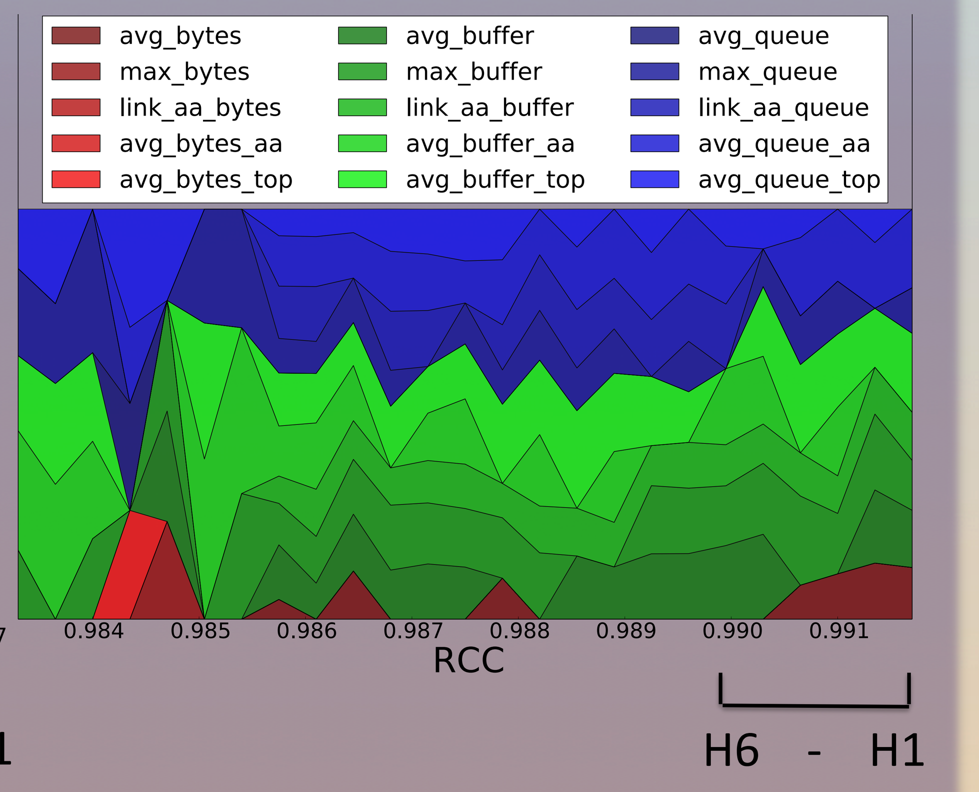
MILC on 4K nodes



pF3D on 1K nodes



pF3D on 4K nodes



Conclusion

We see high correlations between network counters and communication time for production applications

Hybrid metrics tend to consistently have the highest prediction ability with *ensemble methods* (GBRT, Random Forests)

Best metrics for performance prediction typically use many different communication features in various proportions

Future Work

Use of other communication features as model inputs

Evaluation of other machine learning methods

Evaluation of these metrics on other parallel applications

Further study of feature importance

Acknowledgements

Machine learning performed using Scikit-learn Python package
http://scikit-learn.org/stable/

[1] A. Bhatele *et al.* Mapping Applications with Collectives over Sub-communicators on Torus Networks. In Proceedings of SC '12, November 2012. LLNL-CONF-556491

[2] N. Jain, A. Bhatele, M. P. Robson, T. Gamblin, and L. V. Kale. Predicting application performance using supervised learning on communication features. In Proceedings of SC '13, November 2013. LLNL-CONF-635857

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. This work was funded by the Laboratory Directed Research and Development Program at LLNL under project tracking code 13-ERD-055 (LLNL-POST-662676)