PREDICTIVE MODELING OF I/O CHARACTERISTICS IN HIGH PERFORMANCE COMPUTING SYSTEMS

Thomas C. H. Lux

Dept. of Computer Science
Virginia Polytechnic Institute
& State University
Blacksburg, VA 24061
tchlux@vt.edu

Layne T. Watson

Dept. of Computer Science
Dept. of Mathematics
Dept. of Aerospace & Ocean Eng.
Virginia Polytechnic Institute
& State University

Tyler H. Chang
Jon Bernard
Bo Li

Dept. of Computer Science
Virginia Polytechnic Institute
& State University

Li Xu

Dept. of Statistics
Virginia Polytechnic Institute
& State University

Godmar Back
Ali R. Butt
Kirk W. Cameron

Dept. of Computer Science
Virginia Polytechnic Institute
& State University

Yili Hong

Dept. of Statistics
Virginia Polytechnic Institute
& State University

ABSTRACT

Each of high performance computing, cloud computing, and computer security have their own interests in modeling and predicting the performance of computers with respect to how they are configured. An effective model might infer internal mechanics, minimize power consumption, or maximize computational throughput of a given system. This paper analyzes a seven-dimensional dataset measuring the input/output (I/O) characteristics of a cluster of identical computers using the benchmark IOzone. The I/O performance characteristics are modeled with respect to system configuration using multivariate interpolation and approximation techniques. The analysis reveals that accurate models of I/O characteristics for a computer system may be created from a small fraction of possible configurations, and that some modeling techniques will continue to perform well as the number of system parameters being modeled increases. These results have strong implications for future predictive analyses based on more comprehensive sets of system parameters.

SpringSim-HPC, 2018 April 15-18, Baltimore, MD, USA;

Keywords: Regression, approximation, interpolation, performance modeling

1 INTRODUCTION

Performance tuning is often an experimentally complex and time-intense chore necessary for configuring HPC systems. The procedures for this tuning vary largely from system to system and are often subjectively guided by the system engineer(s). Once a desired level of performance is achieved, an HPC system may only be incremenetally reconfigured as required by updates or specific jobs. In the case that a system has changing workloads or non-stationary performance objectives that range from maximizing computational throughput to minimizing power consumption and system variability, it becomes clear that a more effective and automated tool is needed for configuring systems. This scenario presents a challenging and important application of multivariate approximation and interpolation techniques.

Predicting the performance of an HPC system is a challenging problem that is primarily attempted in one of two ways: (1) build a statistical model of the performance by running experiments on the system at select settings or (2) use a performance simulator to run artificial experiments and search for optimal configurations. In this paper the proposed multivariate modelling techniques rest in the first category, however they represent a notable increase in the ability to model more complex interactions between system parameters.

Many previous works attempting to model system performance have used simulated environments to estimate the performance of a system. (Grobelny, Bueno, Troxel, George, and Vetter 2007) (Wang, Butt, Pandey, and Gupta 2009) (Wang, Khasymski, Krish, and Butt 2013) Some of these works cite statistical models as being over-simplified and not capabel of capturing the true complexity of the underlying system. This claim is partially correct, noticing that a large portion of predictive statistical models rely on simplifying the system to one or two parameters. (Snavely, Carrington, Wolter, Labarta, Badia, and Purkayastha 2002) (Bailey and Snavely 2005) (Barker, Davis, Hoisie, Kerbyson, Lang, Pakin, and Sancho 2009) (Ye, Jiang, Chen, Huang, and Wang 2010) These limited statistical models have provided satisfactory performance in very narrow application settings. Many of the aforementioned statistical modeling techniques claim to generalize, while simultaneously requiring additional code annotations, hardware abstractions, or additional application level understandings in order to generate models. The approach presented here requires no modifications of the application, no architecture abstractions, nor any structural descriptions of the input data being modelled. The techniques used are purely mathematical and only need formatted data as input.

Among the statistcal models presented in prior works, (Bailey and Snavely 2005) specifically mentioned that it is difficult for the models to capture variability introduced by I/O. System variability in general has become a problem of increasing interest to the systems and HPC communities, however most of the work has focused on Operating System induced variability. (Beckman, Iskra, Yoshii, Coghlan, and Nataraj 2008) (De, Kothari, and Mann 2007) The work that has focused on managing I/O variability, (Lofstead, Zheng, Liu, Klasky, Oldfield, Kordenbrock, Schwan, and Wolf 2010), does not use any sophisticated modeling techniques. Hence, this paper presents a case study applying advanced mathematical and statistical modeling techniques to the domain of HPC system I/O characteristics. The models are used to predict the mean throughput of a system and the variance in throughput of a system. The discussion section outlines how the exact techniques presented can be applied to any performance metric and any system.

In general, this paper compares five multivariate approximation techniques that operate on inputs in $\mathbb{R}^d$ (vectors of d real numbers) and produce predicted responses in $\mathbb{R}^1$. In order to provide coverage of the varied mathematical strategies that can be employed to solve the continuous modeling problem, three of the techniques are regression-based and the remaining two techniques produce interpolants. The sections below outline the mathematical formulations of each technique and provide computational complexity bounds with respect to the size (number of points and dimension) of input data.

System Parameter	Type	Values
Hypervisor Scheduler	Categorical	CFQ, NOOP, DEAD
Operating System Scheduler	Categorical	CFQ, NOOP, DEAD
IOZone Test Type	Categorical	Fread, Fwrite, RandomRead
File Size	Continuous	64, 128, 256, 512, 1024
Record Size	Continuous	32 64, 128, 256, 512
Thread Count	Continuous	1,2,4,8,16,32,64
Frequency	Continuous	2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1

Table 1: A description of the system parameters being considered in the IOZone tests. There are three categorical settings and four continuous settings. Notice that the ranges of values for continuous settings are different.

1.1 Approximation

Multivariate approximations are capable of accurately modelling complex relationships between the dimensions of any set of points in $\mathbb{R}^n$ with respect to some response in $\mathbb{R}$. The approximations produce a function $f : \mathbb{R}^n \to \mathbb{R}$ which minimizes some metric related to the provided data.

1.1.1 Multivariate Adaptive Regression Splines

1.1.2 Multi-Layer Perceptron Regressor

1.1.3 Support Vector Regressor

1.2 Interpolation

1.2.1 Delaunay

1.2.2 Linear Shepard

The remainder of the paper is broken up into _ parts. *(will fill in the parts and descriptions once they are finalized)*

2 METHODOLOGY

2.1 Data

The summary of the data used in the experiments for this paper can be seen in Table 1.

2.2 Dimensional Analysis

This work utilizes an extension to standard k-fold cross validation that allows for a more thorough investigation of the expected model performance in a variety of real-world situations. Alongside randomized splits, two extra components are considered: the amount of training data provided, and the dimension of the input data. It is important to consider that algorithms which perform well with less training input also require less

Algorithm	Input Dimension	Mean Absolute Error	Mean Relative Error
Delaunay	1	2060267	0.03
	2	2916224	0.07
	3	3001794	0.09
	4	3134779	0.10
LSHEP	1	6353148	0.06
	2	13038882	1.52
	3	5446207	1.13
	4	5133648	1.07
MARS	1	2176368	0.03
	2	6213369	0.61
	3	4024964	0.65
	4	4497414	1.01
MLPRegressor	1	2461097	0.06
	2	8883580	0.81
	3	11190977	2.03
	4	11831721	2.20
SVR	1	19306731	0.96
	2	78213963	18.49
	3	97761266	31.27
	4	113551300	37.2

Table 2: Most models experience a decay in predictive performance as the dimension of the data increases. This is expected because higher dimension input has exponentially more possible patterns to identify. The MLP Regressor and SVR experience the worst decay in performance with increasing dimension.

experimentation. Although, the amount of training data required may change as a function of the number of input dimensions and this needs to be studied as well.

The framework used in this paper will be referred to as a multi-dimensional analysis (MDA) of the IOZone data presented in this study. However, the MDA framework can be applied to other datasets.

1. Cycling the categorical settings
2. Selecting subsets of 1,2,3 up to 4 dimensions
3. Cycling different training : testing ratios (5:95 $\rightarrow$ 95:5)
4. Generating 200 random training : testing splits
5. Ensuring the testing points are on/inside the convex hull of the training.
6. Ensuring the training points are well-spaced.

2.3 Prediction

1. For each file generated from the dimensional analysis, train on the training data, evaluate at the testing data points

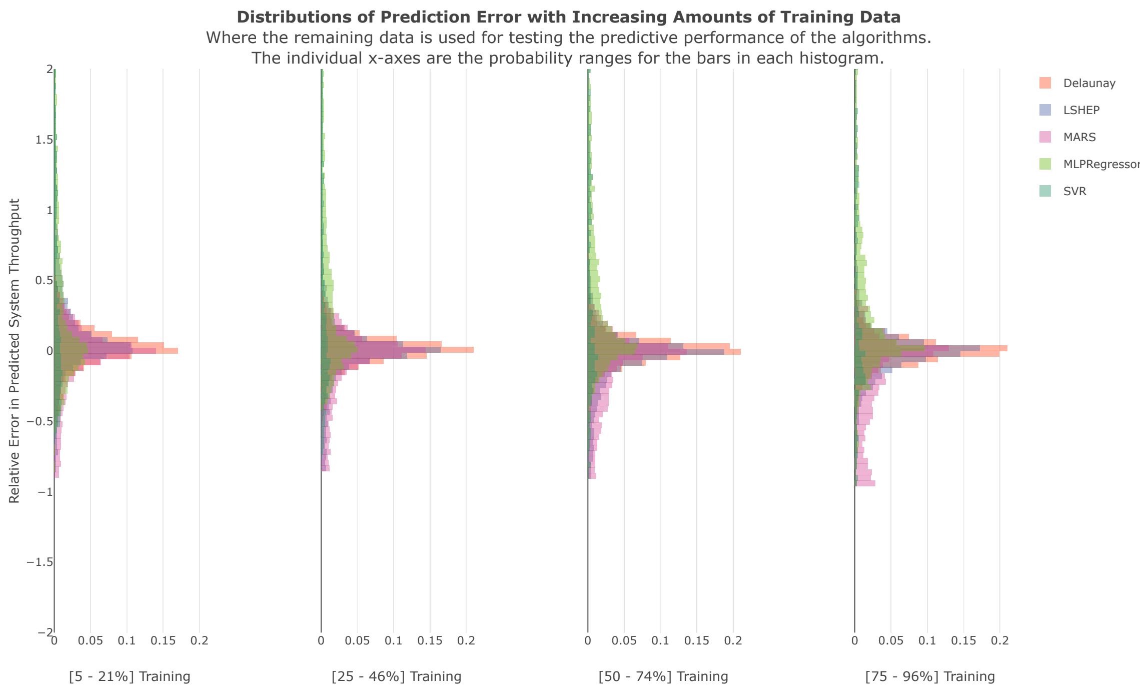

Figure 1: In this figure, it can be seen how the distribution of prediction errors for each algorithm is affected by the quantity of training data provided. Notice that for all amounts of training data, Delaunay has the largest likelihoods around 0 error.

Algorithm	Training Data	Mean Absolute Error	Mean Relative Error
Delaunay	[5-21]%	5274933	0.11
	[25-46]%	2898330	0.08
	[50-74]%	1518477	0.07
	[75-96]%	886346	0.08
LSHEP	[5-21]%	13218469	2.53
	[25-46]%	4936381	0.85
	[50-74]%	2251409	0.26
	[75-96]%	1415034	0.15
MARS	[5-21]%	6543153	0.61
	[25-46]%	4324916	0.80
	[50-74]%	3115943	0.95
	[75-96]%	2591451	0.74
MLPRegressor	[5-21]%	17348982	2.58
	[25-46]%	12504957	2.29
	[50-74]%	5292177	1.17
	[75-96]%	2996253	1.02
SVR	[5-21]%	170906944	49.45
	[25-46]%	99851846	31.28
	[50-74]%	51565405	19.63
	[75-96]%	27125130	12.6

Table 3: All models experience a reduction in error with increasing amounts of training data. This suggests that the data being modeled is pattern-dense, over-fitting is *not* occurring, and that oversimplifications will tend towards worse predictions.

3 RESULTS

3.1 I/O Throughput Mean

3.2 I/O Throughput Variance

3.3 Increasing Dimension

4 DISCUSSION

4.1 Modeling the System

4.2 Quantifying Variability

4.3 Extending the Analysis

5 FUTURE WORK

The most sever limitation to the presented work is the present inability to model the relationship between categorical system parameters. This could be seen as a limitation of the models selected, because only MARS is capable of handling non-numeric variables, or as a limitation of the MDA framework.

REFERENCES

Bailey, D. H., and A. Snavely. 2005. "Performance modeling: Understanding the past and predicting the future". In *European Conference on Parallel Processing*, pp. 185–195. Springer.

Barker, K. J., K. Davis, A. Hoisie, D. J. Kerbyson, M. Lang, S. Pakin, and J. C. Sancho. 2009. "Using performance modeling to design large-scale systems". *Computer* vol. 42 (11).

Beckman, P., K. Iskra, K. Yoshii, S. Coghlan, and A. Nataraj. 2008. "Benchmarking the effects of operating system interference on extreme-scale parallel machines". *Cluster Computing* vol. 11 (1), pp. 3–16.

De, P., R. Kothari, and V. Mann. 2007. "Identifying sources of operating system jitter through fine-grained kernel instrumentation". In *Cluster Computing, 2007 IEEE International Conference on*, pp. 331–340. IEEE.

Grobelny, E., D. Bueno, I. Troxel, A. D. George, and J. S. Vetter. 2007. "FASE: A framework for scalable performance prediction of HPC systems and applications". *Simulation* vol. 83 (10), pp. 721–745.

Lofstead, J., F. Zheng, Q. Liu, S. Klasky, R. Oldfield, T. Kordenbrock, K. Schwan, and M. Wolf. 2010. "Managing variability in the IO performance of petascale storage systems". In *High Performance Computing, Networking, Storage and Analysis (SC), 2010 International Conference for*, pp. 1–12. IEEE.

Snavely, A., L. Carrington, N. Wolter, J. Labarta, R. Badia, and A. Purkayastha. 2002. "A framework for performance modeling and prediction". In *Supercomputing, ACM/IEEE 2002 Conference*, pp. 21–21. IEEE.

Wang, G., A. R. Butt, P. Pandey, and K. Gupta. 2009. "A simulation approach to evaluating design decisions in mapreduce setups". In *Modeling, Analysis & Simulation of Computer and Telecommunication Systems, 2009. MASCOTS'09. IEEE International Symposium on*, pp. 1–11. IEEE.

Wang, G., A. Khasymski, K. Krish, and A. R. Butt. 2013. "Towards improving mapreduce task scheduling using online simulation based predictions". In *Parallel and Distributed Systems (ICPADS), 2013 International Conference on*, pp. 299–306. IEEE.

Ye, K., X. Jiang, S. Chen, D. Huang, and B. Wang. 2010. "Analyzing and modeling the performance in xen-based virtual cluster environment". In *High Performance Computing and Communications (HPCC), 2010 12th IEEE International Conference on*, pp. 273–280. IEEE.