Empowering Machine Learning for the Selection of Robust Algorithms for the Dynamic Scheduling of Scientific Applications

Nitin Sukhija*, Srishti Srivastava*, Florina M. Ciorba†, Ioana Banicescu*, Brandon Malone†

* Mississippi State University  
Department of Computer Science and Engineering  
Mississippi State, MS 39762, USA  
\{nitin@cavs, ss878@, ioana@cs,\}@msstate.edu

† Technische Universität Dresden  
Center for Information Services and High Performance Computing  
01062, Dresden, Germany  
florina.ciorba@tu-dresden.de

† Helsinki Institute for Information Technology  
Department of Computer Science  
0014, University of Helsinki, Finland  
brandon.malone@cs.helsinki.fi

ABSTRACT

With the advancements in computing systems technology, the development and use of complex scientific applications in science and engineering is becoming increasingly prevalent. In general, running such computationally intensive applications on large-scale high performance computing systems exhibits stochastic behavior, due to variability in the execution time of loop iterations and high variability in processor availability. Thus, execution of such applications in computing environments in the presence of unpredictable variations (due to system load, operating system jitter, etc.) requires robust scheduling algorithms to avoid their performance degradation, primarily caused by load imbalance. Dynamic loop scheduling (DLS) techniques are employed to achieve load balancing of scientific applications during their execution on heterogeneous computing systems. The DLS techniques are based on probabilistic analyses, making them inherently robust against a large number of runtime variations in the computing system. Even though many DLS techniques have been developed to improve the performance of scientific applications via load balancing, the scalability and reliability of the scheduling techniques depend on the test beds used to model, test, and evaluate them.

Towards achieving guaranteed optimal performance of the DLS techniques metrics are required to measure their robustness against variations in perturbation factors present in the current and future heterogeneous high performance computing environments. Recently, robustness metrics have been proposed to study the robustness of the DLS methods. The robustness of a DLS technique is defined as its ability to perform well in the presence of perturbations (e.g., possible erroneous inputs), or variations in environmental factors (e.g., variations in processors availability leading to a variation in the system load) [1]. A robust method is preferred over a non-robust method as for a given set of perturbations the performance of a robust method does not decrease below a certain tolerance value. The perturbation factor considered in this work is the fluctuation in system load (Λ), defined as the compound effect of the unpredictable variation in the iteration execution times, and the variance in the delivered computation speed of the processors expressed as the percentage availability of the processors. The fluctuation in system load may be the result of many factors, such as, the variation in processor speeds, architecture, power, and others. A mechanism has been demonstrated to measure the robustness of the DLS methods against system load fluctuation, which is also referred to as a measure of flexibility of the DLS methods, using a flexibility metric [1]. A DLS technique is considered to be most flexible, if it delivers the most optimized performance in the presence of unpredictably fluctuating system load. The flexibility in this case is measured as the distance between the parallel execution time in the ideal execution scenario (where the processors are dedicated to the application throughout the execution) and the parallel execution time obtained when the system load fluctuates during the execution of the application.

Motivation: The flexibility of scheduling algorithms often varies from instance to instance. An instance is an assembly of four attributes: problem size, system size, characteristics of the variations in the application task execution times, and those of the processor availabilities. Since the DLS algorithms employ probabilistic rules, their use results in application runtimes that vary from run to run for a single DLS and among the DLS algorithms. This poses a challenge on predicting which algorithm is most robust in achieving the best performance of a given scheduled application onto a computing system in the presence of variable system load. This motivates the investigation into the use of machine learning techniques to select effective scheduling algorithms for a broad spectrum of scientific applications.

Goal: The goal of the present work is to not only predict the performance of an algorithm on a particular instance but also to enable a dynamic selection of the most robust algorithm for scheduling scientific applications for any new instance.

Problem statement: Given a scientific application (containing N independent loop iterations, whose individual execution times are constant or can vary over time according to the Gaussian, gamma, or exponential distributions resulting in algorithmic variance, T), a collection of DLS algorithms, and a computing system (composed of P heterogeneous processors whose individual availability or speed may not or may vary over time according to uniform or exponential distributions, resulting in systemic variance, d), which algorithm should be employed to guarantee the robust execution of the given
application in the given system in the presence of algorithmic and systemic variances? This problem is referred to as the algorithm selection problem for a portfolio of DLS algorithms (where the portfolio can be the set of algorithms {STATIC, FSC, GSS, FAC, WF, AWF-B, AWF-C, AF}). The most widely adopted solution to this problem is the winner-take-all approach. Applying this approach, the robustness of each DLS algorithm can be measured on a representative set of instances, and the scheduling algorithm having the best robustness value is selected. However, this approach can lead to overlooking of other algorithms that are not robust on certain instances but offer optimal performance on others.

**Approach:** To solve the problem of selecting a scheduling algorithm, we use machine learning algorithms to learn empirical hardness models which act as a predictor of a DLS algorithm’s robustness for a given instance based on the learning acquired through the four attributes of the instances \((N, P, T, A)\) and the algorithm’s past robustness values. Significantly extending previous work, we formally define *empirical robustness prediction models*, which predict the robustness of a DLS algorithm on a given instance. This is a regression problem; consequently, we explore the vast space of regression model classes, their hyperparameters and parameters which solve the regression problem, and select the model which best predicts the robustness of a scheduling algorithm on any instance. The *empirical robustness prediction model* offers the basis for selecting the most robust algorithm from a DLS algorithm portfolio for a given instance based on its characteristics.

The algorithm selection methodology integrates a scheduler and the libraries provided by the SimGrid simulation framework [2] to evaluate the robustness of the DLS methods for scheduling loop iterations on a computing environment given a certain tolerance value \((\tau)\). The robustness of a scheduling algorithm \(S\), defines its performance in the presence of both \(T\) and \(A\). The tolerance value observed via simulation, \(\tau_{sim}\), reflects the robustness of each \(S\) for a particular instance \(c\) represented by the 4-tuple: \((N, P, T, A)\), and it must satisfy the right hand side inequality below:

\[
T_{\text{PAR}}^{\text{ideal}} \leq T_{\text{PAR}} \leq \tau_{\text{sim}} \cdot T_{\text{PAR}}^{\text{ideal}}
\]

The simulated tolerance \(\tau_{sim}\) denotes the simulated impact of variable system availability \((A)\) on the total parallel execution time \(T_{\text{PAR}}\).

**Contribution:** The poster illustrates the following contributions, which have also been studied in a recent work of our research group [3]: (i) Employing state-of-the-art machine learning techniques to explore a large space of empirical robustness prediction models. The learned models predict robustness much more accurately than the previous models; (ii) Selecting DLS algorithms from a portfolio with the developed empirical robustness prediction models. The prediction models enable an algorithm selection on a per-instance basis; and (iii) Experimentally evaluating the performance of the per-instance selections compared to the simpler winner-take-all approach. The results show that algorithm selection based on the empirical robustness prediction models yields more robust performance on large number of instances than the selections made using winner-take-all.

We ran Auto-WEKA on the training set to search both for good model classes and hyperparameters for empirical robustness prediction models. The empirical robustness prediction models \(\tau_{i}\) give a mapping from an instance \(c=(N, P, T, A)\) and an algorithm \(S\) to \(\tau_{\text{pred}}\), the predicted tolerance of \(S\) on those 4 instance attributes. In the algorithm selection process the algorithm with the lowest predicted robustness for \(c\) is selected for use by the portfolio. The algorithm selection process simulates real online behavior because the models are used to make predictions on instances not seen in training.

The methodology for solving the DLS selection problem along with the experiments demonstrating higher guarantees regarding the robust performance of the application using the automatically selected DLS algorithms when compared to the robust performance of the same application using a manually selected DLS algorithm will be presented during the poster session.

**Keywords**
Dynamic loop scheduling; robustness; algorithm selection; empirical prediction models; machine learning; variable system availability; SimGrid.

**ACKNOWLEDGMENTS**
This work is in part supported by the National Science Foundation under grant number NSF IIP-1034897, by the Center for Advanced Vehicular Systems (CAVS), by the German Research Foundation (DFG) in the Collaborative Research Center 912 “Highly Adaptive Energy-Efficient Computing”, and by the Academy of Finland (Finnish Centre of Excellence in Computational Inference Research COIN, 251170).

**REFERENCES**