DEVELOPING A TOOLKIT FOR TASK CHARACTERISTICS PREDICTION BASED ON ANALYSIS OF QUEUE'S HISTORY OF A SUPERCOMPUTER

M. Rezaei[1,a], A. Salnikov[1,2], A. Shiryaev[1]

[1] *Moscow Institute of Physics and Technology, Moscow, Russia*

[2] *Lomonosov Moscow State University, Moscow, Russia*

E-mail: [a] mahdirezaei_ai@yahoo.com

Empirical studies have repeatedly shown that in High-Performance Computing HPC users' resource estimations lack accuracy. Therefore, resource underestimation may remove the job at any step of computing and subsequently allocated resources will be wasted. Moreover, resource overestimation also will waste resources. In this work, to effectively utilize the overall HPC system, we proposed a new approach to predict the required resources such as; number of required CPUs, time slots etc. for newly submitted job. The study focused on predictive analytics tasks including regression and classification. A supervised machine learning system, comprising several models, was trained based on the collection of statistical data including per-job and per-user features collected from the reference queue systems. Results indicated that adding more features to the dataset improves the prediction accuracy. The possibility of designing a plugin to apply our machine learning system in practical applications was studied. A dynamically connected SLURM SPANK plugin was created that adds the "--predict-time" option and takes control on srun and sbatch commands while they are executed. It was found that the plugin enables practical use of our proposed machine learning system.

Keywords: machine learning, predictive analytics, decision making, HPC, job scheduling, SLURM plugin.

Mahdi Rezaei, Alexey Salnikov, Alexander Shiryaev

1. Introduction

High-Performance Computing (HPC) applications are intrinsically computational and data intensive. Because of the dynamicity of resources and on-demand user application requirements, job scheduling in such environments is an NP-Complete and complex problem [1]. Resource allocation is done by job schedulers but the major drawback of job schedulers is that required resources must be known to the scheduler at the time of job submission. Empirical studies have repeatedly shown that users' resource estimations lack accuracy [2], whereas there is no automatic system in HPCs that can effectively allocate resources. The SLURM, a famous job scheduler, has a mechanism to predict only the starting time of a job. However, this mechanism is very primitive and more often the time is overestimated. The purpose of this work is to do predictive analytics on resource allocation using machine learning methods. Using algorithms, including machine learning algorithms, to predict required resources for jobs has been pursued by several previous studies [3-8]. Using historical data is a reasonable method to improve the performance of the schedulers in order to utilize the overall HPC system efficiently [9]. We created also a plugin to study the possibility of applying our system on real clusters. Our proposed plugin is dynamically connected SPANK plugin, that adds the option '--predict-time' and while executing srun and sbatch commands, takes control on them. The block diagram of our proposed system is shown in [fig. 1].

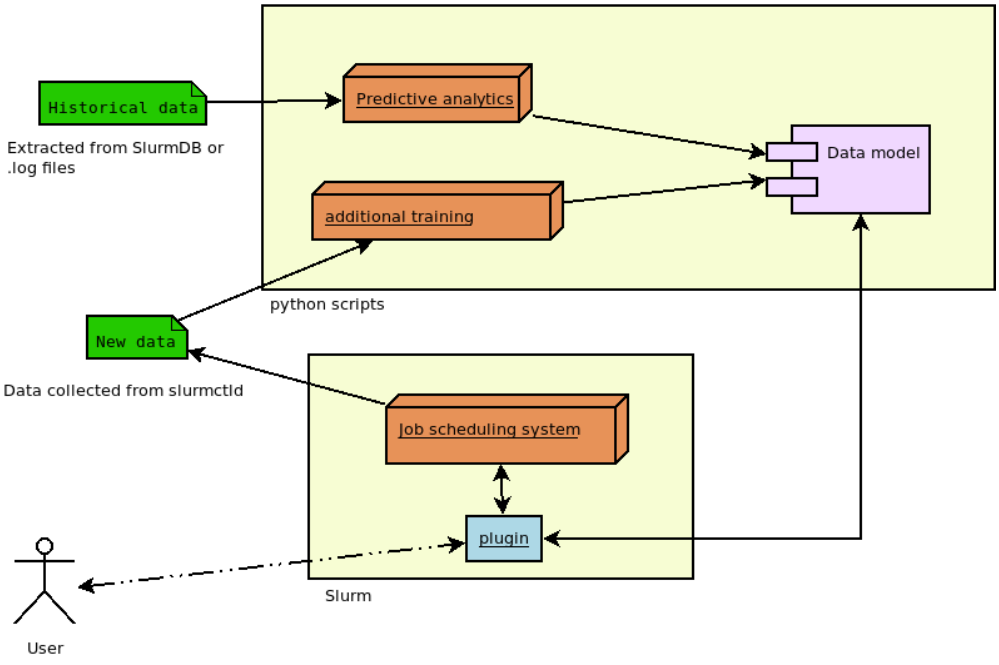

Figure 1. Block diagram of proposed system

2. Supervised machine learning and predictive analytics

Machine learning plays an important role in predictive analytics as algorithms are capable to be trained based on a historical dataset. Dagnino et al. have already discussed the applicability of machine learning in data analytics [10, 11]. Nowadays many of applicable machine learning systems use supervised learning. Different machine learning techniques and methods are available. Therefore, different algorithms can be defined to find the best resource prediction in an HPC system [12]. The study focuses on predictive analytics, including regression and classification tasks.

2.1 Regression algorithms

Regression algorithms have continuous outputs and are used to predict the resources. They are as follows: Multilayer Perceptron (MLP), Random Forest Regression (RFR), Lasso Regression (LR), K-Nearest Neighbor Regression (KNN), Ordinary Least-Squares Regression (OLSR), Support Vector Regression (SVR), Ridge Regression (RR), Polynomial Regression (PR), and CART Regression (CARTR).

2.2 Classification algorithms

Classification algorithms are used when the output is a discrete label and are used to check the failure of jobs. They are as follows: Naive Bayes classifier, Kernel Support Vector Machines (SVM), CART classification.

2.3 Data preparation

To check the efficiency of our system we used collected statistics of Bluegene/P system installed at the Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University named after M.V Lomonosov. This statistic includes information on jobs run for almost 12 months in 2017. It has about 112000 instances. We used 80% of the dataset for training and 20% for testing. The per-job features, presented in Table 1, in first phase were used to train our machine learning. In second phase, 7 metrics known as per-user features, shown in Table 2, were added to our dataset to train our machine learning.

2.4 Machine learning experimental results

Our findings presented in [fig. 2a] show that in general, the predictions by MLP, RFR, KNN, PR, and CARTR are promising. After adding per-user features to the dataset, it was observed that most of the models had improved. RFR, KNN, and CARTR improved a little. However, the performance of MLP and PR drastically improved after adding per-user features to the dataset. Another promising finding was that after adding per-user features to the dataset all the models improved to predict the required time.

Table 1. per-job features

Feature	Type	Description
time_limit	Numeric	The time requested by the user for a job.
num_cpus	Numeric	The number of CPUs requested by the user for a job.
id	String	Task identifier, as identified in our job scheduling system.
name	String	Task name specified by the user.
user	String	User name (extract with getpwuid (euid)).
group	String	User group (extract with getgrgid (egid)).
task_class	String	Class of task. It is assigned automatically by scheduling system or by the user in case if allowed. It is used to split the priorities of tasks into several classes. Each class has its own priority.
state	String	The status of job which is either completed or removed.
required_time	Numeric	The time during which a job is done. This time will be predicted for newly submitted jobs.
required_cpu	Numeric	The number of CPUs used by a task during execution. This number will be predicted for newly submitted jobs.

As [fig. 2b] shows, RFR is the best model and the highest increase in accuracy rate belongs to the SVR model. From [fig. 2c], we can see that CART classification is the best model for classification.

Table 2. per-user features

Feature	Type	Description
used_portion_of_time_limit	Numeric	The degree of reasonableness of execution time requested by the user.
avg_aborted_task	Numeric	Percentage of aborted tasks submitted by the user.
average_congestion	Numeric	The average occupancy of the system by the user.
average_cpus	Numeric	The average number of requested CPUs by the user.
duration	Numeric	The average waiting time of the user in the queue.
wait_time / time_limit	Numeric	The average ratio of time in the queue to the requested time.
average_time_limit	Numeric	The average of time limits set by the user.

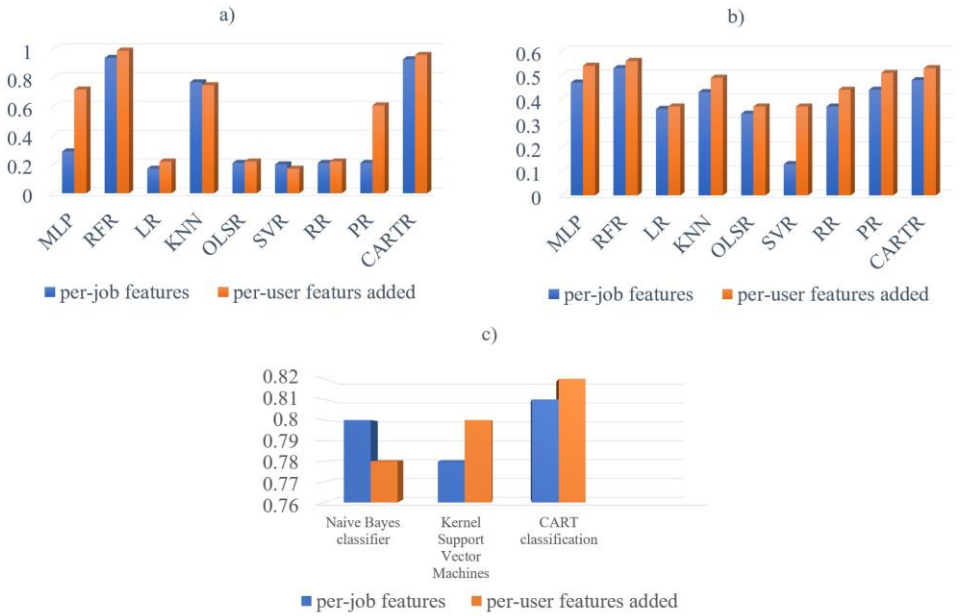

Figure 2. (a) R2 values in the regression problem to predict the number of required CPUs (b) R2 values in the regression problem to predict the required time (c) F1-score of the classification value

3. Plugin

To make our system applicable on the SLURM we need to create a plugin. Therefore, a component was designed to connect to the SLURM. It has the ability to collect statistics, analyze them and based on the analysis create a model. Using this model, the prediction could be done. Plugins may not only complement, but also modify the behavior of SLURM to optimize the system's performance. SLURM has a centralized SLURMctld manager for monitoring resources and tasks. Every compute server (node) has a SLURMd daemon that could be compared with a remote shell: it waits for a job, runs that job, returns state and waits for more work. In addition, SLURM has several utilities for its users. In this work we are mainly interested in 2 utilities: srun - to start a job and sbatch - for placing jobs in the queue. More precisely, sending a batchscript (instructions to SLURM to perform the job) to the SLURM.

3.1 Review of existing solutions

The standard SLURM package has a mechanism to predict only the starting time of a job. However, this mechanism is very primitive and more often the time is overestimated. Moreover, there is a software system for modeling the activity of computing cluster users based on the SLURM, which uses the collection of statistics to simulate the load on a model of computing cluster under the control of SLURM. This software lists several metrics used by the system administrators of clusters. This

approach has been tested on data from computing clusters of the Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University and NIKIET JSC [13]. However, this solution is also not suitable because it does not allow to analyze and prediction. There are other systems [14] for analyzing the efficiency of using a cluster. Nevertheless, trying to connect such systems to the SLURM wastes large amount of resources, while our proposed method only requires the development of a component where analysis system can easily get embedded. In another work [15], various metrics for clusters are studied, where the user will have to use these metrics manually to predict the run time of the job which is not a ready-made solution.

3.2 Statement of technical specifications

The task is done through design, develop and test the functionality of the component (or components) connected to the SLURM queuing system. The component collects statistical data and does analysis on the flow of computational tasks. The component is a system of two modules implemented as:

- Add-ons to SLURM to add and handle the --predict-time option in the srun and sbatch commands to predict the job run time.
- A Linux daemon that handles HTTP requests, trains the model, and etc. (hereinafter - the main application). The application must be able to connect custom algorithms according to a given template.

In addition, special scripts are required to install these modules easily.

4. Conclusion and future work

Our work has led us to conclude that adding new features to the dataset improves prediction accuracy. An innovative solution for the resource allocation problem was found. The possibility of writing a plugin to apply our machine learning system in practical applications was studied. It was found that designing a plugin allows the practical use of machine learning algorithms in decision making. However, it is required to improve the performance of this component. In future work, we will use this component to evaluate our algorithms on a real cluster to find the best method to do prediction.

References

[1] Jeffrey D. Ullman, NP-complete scheduling problems, J. Comput. System Sci. 10 (3) (1975) 384–393.

[2] D. T safrir, Y. Etsion, and D. G. Feitelson, “Backfilling Using System-Generated Predictions Rather than User Runtime Estimates,” IEEE Trans. Parallel Distrib. Syst., vol. 18, no. 6, pp. 789-803, 2007.

[3] A. W. Moore, J. Schneider, and K. Deng, “Efficient Locally Weighted Polynomial Regression Predictions,” in Proc. 14th Int. Conf. Machine Learning, 1997, pp. 236-244.

[4] A. W. Mu'alem, and D. G. Feitelson, “Utilization, Predictability, Workloads, and User Runtime Estimates in Scheduling the IBM SP2 with Backfilling,” IEEE Trans. Parallel Distrib. Syst., vol. 12, no. 6, pp. 529-543, 2001.

[5] W. Smith, I. Foster, and V. Taylor, “Predicting application run times with historical information,” J. Parallel Distrib. Comput., vol. 64, no. 9, pp. 1007-1016, 2004.

[6] R. Albers, E. Suijs, and P. H. N. de With, “Triple-C: Resource usage prediction for semi-automatic parallelization of groups of dynamic image-processing tasks,” in Proc. 23rd Int. Parallel Distributed Processing Symp., 2009.

[7] R. Duan, F. Nadeem, J. Wang et al., “A Hybrid Intelligent Method for Performance Modeling and Prediction of Workflow Activities in Grids,” in Proc. 2009 9th IEEE/ACM Intl. Symp. Cluster Computing and the Grid, 2009, pp. 339-347.

[8] F. Nadeem, and T. Fahringer, “Using Templates to Predict Execution Time of Scientific Workflow Applications in the Grid,” in Proc. 2009 9th IEEE/ACM Intl. Symp. Cluster Computing and the Grid, 2009, pp. 316-323.

[9] Predicting application run times with historical information Warren Smitha, Ian Fosterb, Valerie Taylorc,2004

[10] A. Dagnino, K. Smiley, and L. Ramachandran, "Forecasting Fault Events in Power Distribution Grids Using Machine Learning," Proceedings of the 24th International Conference on Software Engineering and Knowledge Engineering (SEKE'12), San Francisco, CA, USA, pp. 458-463, July 2012.

[11] A. Dagnino and D. Cox, "Industrial Analytics to Discover Knowledge from Instrumented Networked Machines", Proceedings of the 26th International Conference on Software Engineering and Knowledge Engineering (SEKE’14), Vancouver, Canada, July 2014.

[12] Rodrigues, E. R., Cunha, R. L., Netto, M. A., and Spriggs, M. “Helping HPC users specify job memory requirements via machine learning,” in Proceedings of the Third International Workshop on HPC User Support Tools, pp. 6-13, IEEE Press, 2016.

[13] A software system for modeling the activity of users of a computing cluster based on the queuing system SLURM - article // TRUE - Intellectual System for Thematic Research of Scientometric Data URL: http://omega.sp.susu.ru/books/conference/PaVT2015/short.html

[14] Leonenkov S., Zhumatiy S. (2019) Supercomputer Efficiency: Complex Approach Inspired by Lomonosov-2 History Evaluation. In: Voevodin V., Sobolev S. (eds) Supercomputing. RuSCDays 2018. Communications in Computer and Information Science, vol 965. Springer, Cham.

[15] B. Song, C. Ernemann and R. Yahyapour, "User group-based workload analysis and modelling," CCGrid 2005. IEEE International Symposium on Cluster Computing and the Grid, 2005., Cardiff, Wales, UK, 2005, pp. 953-961 Vol. 2, doi: 10.1109/CCGRID.2005.1558664.