Azerbaijan Journal of High Performance Computing, Vol 3, Issue 2, 2020, pp. 223-233
https://doi.org/10.32010/26166127.2020.3.2.223.233

Machine Learning-Based Modeling for Performance Improvement in an Exascale Systems

Etibar V. Vazirov

Department of General and Applied Mathematics, Azerbaijan State Oil and Industry University, Baku, Azerbaijan, etibar.vazirov@asoiu.edu.az

*Correspondence: Etibar V. Vazirov, Department of General and Applied Mathematics, Azerbaijan State Oil and Industry University, Baku, Azerbaijan, etibar.vazirov@asoiu.edu.az

Abstract

The combination of heterogeneous resources within exascale architectures guarantees to be capable of revolutionary compute for scientific applications. There will be some data about the status of the current progress of jobs, hardware and software, memory, and network resource usage. This provisional information has an irreplaceable value in learning to predict where applications may face dynamic and interactive behavior when resource failures occur. In this paper was proposed building a scalable framework that uses special performance information collected from all other sources. It will perform an analysis of HPC applications in order to develop new statistical footprints of resource usage. Besides, this framework should predict the reasons for failure and provide new capabilities to recover from application failures. We are applying HPC capabilities at exascale causes the possibility of substantial scientific unproductiveness in computational procedures. In that sense, the integration of machine learning into exascale computations is an encouraging way to obtain large performance profits and introduce an opportunity to jump a generation of simulation improvements.

Keyword: High Performance Computing, Machine Learning, Exascale Computing System, Artificial Intelligence.

1. Introduction

The model for responding of an application in an acceptable possible time in High Performance Computing Systems is very important for system management. In traditional HPC systems, an execution pattern is designed based on application features and system capabilities. The designer of the computing and processing system obtains this pattern. Elements that form the computing system's manager should be able to set this pattern up at runtime. In the Distributed Exascale Computing Systems, the existence of dynamic and interactive behavior of this kind of system turns the application and the constituent elements of the system changing during the runtime. To model execution patterns and power consumption, the use of machine learning techniques and characterizing performance data are gaining popularity in the

HPC community (Thiagarajan, J. J., Anirudh, R., et al., 2018, May).

There has been an outbreak in the amount of performance data that can be gathered, so many components are identified as potential sources that can impact performance. Hence, the performance analysis of HPC codes requires analyzing enormous amounts of data. They are generated approximately at the rate of tens to hundreds of terabytes a day. The enormous volume and variety of gathered data make it unmanageable in many cases for a human to analyze and get insights. Consequently, the methodologies for automated performance analysis are actively followed in HPC systems (Huck, K. A., & Malony, A. D., 2005, November). It is mostly getting more common to adopt corresponding tools (scikit-learn, caffe, TensorFlow, and so on) from machine learning (ML) for intelligence analysis and pattern invention. This has enabled researchers to build predictive models, conclude complex correlation patterns, determine fields of interest for performance optimization, and detect irregular patterns (Bhowmick, S., Eijkhout, V., Freund, Y., Fuentes, E., & Keyes, D., 2006; Sukhija, N., Malone, B., Srivastava, S., Banicescu, I., & Ciorba, F. M., 2014; Bhatele, A., Titus, A. R., et al., 2015, May; Yeom, J. S., Thiagarajan, J. J., et al., 2016; November; Islam, T. Z., Thiagarajan, J. J., et al., 2016, November).

Furthermore, without having appropriate knowledge about specific properties of the system, such as parallel programming on multi-core (or multi nodes) computing resources in HPC environments and quantity of computing nodes (cores) suitable for the specific application job, users cannot request the runtime estimation of the submitted job for system scheduling. Fundamentally, there are two kinds of runtime in terms of estimation: underestimated and overestimated. In underestimated runtime, the HPC system terminates the job before its completion, while an overestimated runtime results in a longer queuing time. Therefore, the productivity and efficiency of the HPC system slow down in each case (Zhang, H., You, H., Hadri, B., & Fahey, M., 2012). For this purpose, a data-driven approach, which actively observes, analyzes, and logs jobs collected from the large-scale supercomputers for predicting job statuses, has been introduced on HPC systems (Guo, J., Nomura, A., Barton, R., Zhang, H., & Matsuoka, S., 2018, March). Once again, using machine learning techniques, it is possible to analyze job data and then build models in various HPC scientific applications. In that sense, one of the valuable approaches found so far, is using machine learning-based technique, which builds classifiers that, can recognize hidden patterns in the collected data. This technique really makes sense to understand what jobs are running in the system and the number of resources allocated at each node (Guo, J., Nomura, A., Barton, R., Zhang, H., & Matsuoka, S., 2018, March). This productive and efficient approach can be used to provide runtime estimation of the scalable framework proposed in this paper.

2. Why Did We Get for Exascale?

Supercomputers need to have extensive memories, maintain and read vast portions

of data at high speed for being profitable to a large spectrum of applications. Besides, the supercomputer must have a software environment that simplifies the systems' accurate and profitable use. By 2025, International Data Corporation estimates that worldwide data will grow 61% to 175 zettabytes (The Exascale Era Is Coming, And Here's Why It Matters). From small private companies to the largest government labs, everyone will be looking for ways to turn this vast amount of data into meaningful insights, using combinations of modeling, simulation, data analytics, and artificial intelligence. Therefore, the exascale era demands a renovation of digital infrastructure since it can manage a large amount of data and support hybrid workflows putting together simulation, analytics, and machine learning. Several examples can be mentioned to explain why exascale matters today. For instance, the exascale era can make nuclear energy inexpensive and small-scale. Nuclear power can come back into a competitive economic environment with the help of mini modular reactors. So the dependence on fossil fuels and carbon emissions may be reduced significantly enough (The Exascale Era Is Coming, And Here's Why It Matters).

Pollution caused by burning fossil fuels can be minimized using exascale computing as well. Currently, 85% of the world's energy is provided by burning fossil fuels. By applying exascale computing, it is highly expected that it will be possible to improve the efficiency of combustion systems in engines and gas turbines (Why We Need Exascale Computing).

Using exascale computing applications in biology, for dynamic models of metabolism, it can be realizable to predict possible parameter values. This would enable scientists to model organisms that would accomplish some tasks. These patterns might also be profitable in the evolution of treatments for well-known types of infections (Why We Need Exascale Computing).

As it seems from these examples, the application of exascale projects is getting more critical for the future world. In order to be successful at exascale - related applications, improving performance is in high demand. In that sense, this paper proposes an ML training model that predicts runtime failures using application resources and performance features, which plays a key role in acquiring high-level improvement at runtime performance.

3. The Convergence of ML and HPC

The convergence of HPC and learning techniques provide a hopeful approach to substantial performance advancements. As it is known, traditional HPC simulations are reaching the limits of original progress. Improving the effectiveness of simulations has always been necessary all the time. Nevertheless, its importance and significance increase strongly in large-scale systems. First, there is a need for enhancement if preservation is not reliable in terms of computational efficiency at scale.

Applying high-performance computing experiences at exascale, makes the high possibilities for more significant scientific inefficiency in computational campaigns.

Therefore, the algorithms, techniques, and campaign strategies worked at lower scales are not indeed a good fit at greater scales, the integration of ML into computations is a reasonable way to obtain large performance benefits in various application domains. In the paper (Jha, S., & Fox, G., 2019, September), the interface opportunities between high performance simulations and machine learning are discussed. In that sense, several important distinctly different connections between machine learning (ML) and HPC have been identified in this paper. Since the HPC performance challenges and machine learning techniques jointly appear in many scientific research types, the term ML and HPC has been defined with two broad categories (Dean, J., 2017, December): HPCforML and MLforHPC. HPCforML uses HPC to execute and improve ML performance or utilizing HPC simulations and methodologies to train ML algorithms. In this case, they are used to understand experimental data or simulations. When it comes to the term MLforHPC, it uses ML to improve HPC applications and systems, where an enormous amount of data comes from the computation or experimental sources that are coupled to ML or HPC elements. The MLforHPC category covers all aspects of machine learning interacting with computation, in most cases performed as HPC. To learn from simulations and produce learned replacements for the simulations using ML techniques, the term MLaroundHPC has been introduced (Jha, S., & Fox, G., 2019, September).

Since supercomputers are becoming more capable in their development toward exascale performance levels, researchers and scientists can gradually run more detailed and accurate simulations to study computational problems gradually. These powerful simulations demand a large amount of computer time, so they are computationally overpriced and consuming 10 to 50 million CPU hours for a single simulation. For instance, running a 50-million-hour simulation on all 658,784 compute cores on the Cori supercomputer NERSC would take more than three days. To explore wide ranges in parameter space, running thousands of these simulations would be uncontrollable (ExaLearn Project to bring Machine Learning to Exascale).

A new machine learning project called ExaLearn intends to create new tools to help researchers and scientists overcome this challenge by applying machine learning to large-scale experimental datasets and simulations. This project focuses on the emulator models, which are built to provide quick approximations of overpriced simulations.

ExaLearn ML project on the exascale system allows scientists to develop further simulations more cheaply and run much quicker on many fewer processors. For this purpose, the managing team of this project runs thousands of simulations that are computationally high-priced over a large parameter space to train the computer to recognize patterns in the simulation data. Then this will help the computer to create a computationally low-priced model. In order to fill in the blanks between the results of the more expensive models, this model smoothly interpolating between the parameters was initially trained (ExaLearn Project to bring Machine Learning to Exascale). Although training would also take a long time, it is expected that these models will generate new

simulations just in few seconds.

ExaLearn co-design center has been announced by the Exascale Computing Project (ECP), which focuses on exascale machine-learning technologies. This project intends to provide exascale machine-learning software for utilization by ECP applications projects, other ECP co-design centers, and US Department of Energy (DOE) testing centers and computing institutions. To remain competitive in computational science and engineering by making effective use of future exascale systems, the ExaLearn project plays a significant role by its new services and tools. In that sense, training a computer over a large parameter space for recognizing patterns in the simulation data can be useful for the introduced model in this paper to predict failures at runtime using application resources and performance features.

Paper (Netti, A., Kiziltan, Z., et al., 2019, August) proposes and analyzes a fault classification model based on supervised ML appropriate for online deployment in HPC environments. It relies on a gathering of performance measuring metrics that are easily accessible in most HPC systems. It is mentioned that this method can categorize roughly various types of faults, such as software issues and bugs.

Paper (Tuncer, O., Ates, E., et al., 2018) propose a model using ML for the diagnosis of performance issues in HPC systems. However, they do not consider the shortages that lead to errors and failures, which cause an inconsistency of the computation. They deal only with performance outliers that result in tedious runtimes for applications.

In this paper, a supervised ML technique has been proposed to build a training model that predicts runtime failures using application resources and performance features.

4. Related Works

Since, it is intended to build a scalable framework that uses special performance information, it is extremely substantial to observe and analyze application performance variation, leading to early job termination, reduced performance, and waste compute platform resources. To overcome this problem, system administrators must predict and discover the reason for failures leading to performance variation and should take reasonable preventive actions. HPC operators usually monitor system health by observing the performance variations and identifying the associated root causes. For this purpose, they continuously collect the system logs along with performance counters and resource usage data.

Using machine learning algorithms, paper (Tuncer, O., Ates, E., et al., 2017, June) presents a framework that takes advantage of resource usage and performance counter data collected during application runs. This machine learning framework can automatically detect the computing nodes that presented known performance anomalies and identify the anomaly type.

In the automated anomaly determinations based on application performance and resource usage, a critical problem is the uncontrollable volume of data observed at

runtime (Ibidunmoye, O., Hernández-Rodriguez, F., & Elmroth, E., 2015). Different types of measurement reduction methods, such as principal component analysis, have been utilized in these papers to solve the mentioned problem (Fu, S., 2011, December; Guan, Q., & Fu, S., 2013, September; Lan, Z., Zheng, Z., & Li, Y., 2009). On the other hand, the features that are useful for anomaly detection maybe be lost due to using the techniques focused on reducing the dataset.

Some statistical approaches and machine learning algorithms have been used so far by many researchers in order to identify and classify some specific anomalies occurred on the system, such as high network overload (Bhatele, A., Titus, A. R., et al., 2015, May), poor file system performance (Kasick, M. P., Gandhi, R., & Narasimhan, P., 2010, October), issues relevant to temperature (Baseman, E., Blanchard, S., et al., 2016, August), or memory-exceeded errors (Brandt, J., Chen, F., et al., 2010, June). Using these methods allows to detect appropriate specific anomalies with high accuracy, but there is no existing method that provides a comprehensive framework to discover and categorize anomalies occurred in computing nodes.

In the paper (Bhattacharya, T., Brettin, T., et al., 2019), was discussed advancing AI Cancer Research project with exascale high performance computing. In building ML models related to cancer, the main goals are developing AI models for cancer research capable for scaling on next-generation high performance computers and evaluating resiliency and trustfulness in the AI models. The application of AI in cancer research has been enhanced by these primary areas: advances in AI and ML algorithms that enable learning from complex, exascale data and advances in computer architectures allowing the revolutionary acceleration of simulation and machine learning algorithms. These advances help to build special ML models that can provide transformative perceptions from data, including molecular dynamics simulations, next-generation sequencing, imaging, and unorganized clinical text documents (Bhattacharya, T., Brettin, T., et al., 2019). This project integrates various types of generated data, pre-exascale computing resources, and advances in ML models to improve comprehension of basic cancer biology, discover prospective new treatment options, predict results, and in conclusion, determine appropriate treatments for patients with cancer.

Based on the log file analysis, a significant number of researches and studies have been conducted so far (Netti, A., Kiziltan, Z., et al., 2019, August; Kasick, M. P., Gandhi, R., & Narasimhan, P., 2010, October; Baseman, E., Blanchard, S., et al., 2016, August). However, in the exascale HPC environment, different authors' mentioned methods may not be efficient enough. For instance, in paper (Tuncer, O., Ates, E., et al., 2017, June), various applications are taken for disposable offline training using different input sizes and input data. This single training procedure may not produce a generic ML model since this resulting model will be specific to the training's selected applications. Besides, having trained for different input sizes and input data just for once, will affect on final model undoubtedly. From this point of view, the modeling principle introduced in this paper is extremely profitable and effective.

5. Machine Learning-Based Modeling

It is highly expected to observe large scientific inefficiency in computational activities when applying high performance computing experiments at exascale. Concerning the fact, that algorithms, techniques, and experimental strategies worked at lower scales are not satisfactory at larger scales. In such a situation, to provide extensive performance profits in different application domains, the integration of machine-learning algorithms into computations would be an intelligent method.

In this paper, applying application resources and performance features is proposed to predict failures rather than trusting system logs. First, it is conducted offline training by running heterogeneous applications with different input sizes and data. On each node at each application, some indicators measuring performance and resource usage are assigned. When the execution of the application finishes, the information from these indicators is collected to analyze and calculate the statistical features. Depending on the information introduced by the corresponding node, this node is labeled as failure or success. These labels and the features computed per node will be input data to train various machine learning algorithms for building a generic ML model. k-nearest neighbors and random forests can be an excellent example of this kind of ML algorithms. Once this model is ready, the application resource usage and performance indicators at runtime can be observed to extract their statistical characteristics. Now it is time to use the ML models built during the training time for testing the statistical characteristics of runtime.

To maximize the ML model's efficiency, which is going to get built during training time, the input size and data size of running heterogeneous applications can be extended. So the overall system is shown in figure 1:

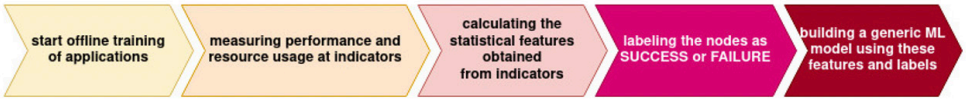

Fig. 1 Structure of ML model

As you see in Figure 1, in the first step, many applications with the same number of nodes are collected for offline training. It is essential to mention that the number of applications and the number of nodes at each application are optional and can be increased because of the unsatisfactory resulting ML model. In the second step, it is time to measure and get information about the nodes' performance and resource usage and put them in a dataset. Since this dataset is a significant resource for extracting valuable data, it has been used in the next step for calculating the statistical features. Once these features are determined and classified, the nodes will be labeled 'success' or 'failure' in step four. In the last step, these features and labels of nodes will be input data to train various machine learning algorithms

for building a generic ML model. The implementation of these five steps can be replicated until a useful ML model is obtained.

In this study, has been stated that by conducting consecutive offline pieces of training over applications, building a very efficient ML model will be used for classifying observations at runtime phase. This operation plays a crucial role in gaining enormous improvement at runtime performance.

6. Mathematical Representation of ML Modeling

Let's suppose there are n applications with k nodes in each one: Aj = {n_1, n_2, …, n_k}, j= 1 to n.

Since it is mentioned that there are indicators for resource usage and performance measurement, then they can be introduced like below functions:

$f_j(n_i) = p_i$, i = 1 to k, j = 1 to n

$g_j(n_i) = r_i$, i = 1 to k, j = 1 to n

where f_j ang g_j are indicator functions for resource usage and performance measurement, respectively. Similarly, p_i and r_i are the corresponding statistical features for these measures. So it is possible to derive a general indicator function vector $m_j = f_j$ o g_j and general statistic feature vector $z_i = (p_i , r_i)$. In that sense, if index i is considered as a label of node and zi as a feature of vector computed per node, then they will be input data for building ML models during offline training. The key aspect behind this approach is collecting training data (z_i , y_i) for $z_i \in T \subset D$, where T is a set of training vectors, D is the full data set, z_i and $y_i = f(z_i)$ are input and output, respectively. This f function mapping the inputs to the output is in charge of modeling procedures. The purpose of the supervised learning approach of ML is to find a deputy function h for f such that the difference between f (z_i) and h(zi) is minimal for all $z_i \in T$ (Malakar, P., Balaprakash, P., et al., 2018, November).

Above, it is mentioned that this kind of ML algorithms can be seen as an example of k-nearest neighbor regression (knn). When a prediction is necessary for a testing point x* , then knn finds k nearest training points and gives back the mean of the corresponding k outputs as the predicted result. Typically, k and the distance measurement are the parameters specific to the learning task and determined by the user (Malakar, P., Balaprakash, P., et al., 2018, November).

7. Discussion

To deal with the revolutionary computation of scientific applications with exascale architectures and predicting failures of the different components, this paper proposes a special ML strategy that can be very useful for performance improvement. Since, it is necessary to have a huge amount of ML training data and resources for accuracy, conducting adjustable offline training over a large number of applications is the best starting point. Moreover, it has been followed by measuring, calculating, labeling, and finally making a decision to predict the best

satisfactory model in this strategy. At this point, it is good to mention that integrating ML algorithms efficient for the distributed exascale systems, using this strategy may contribute huge profits to medicine, global industry, nuclear energy, and many other worldwide fields. This kind of enormous projects would derive very significant challenges of using ML in Exascale projects such as pollution caused by burning fossil fuels, cancer treatment in an advanced phase, using mini modular reactors for making nuclear energy cheap, and so on. Nevertheless, it seems possible to overcome this kind of hard challenges by applying reinforcement learning of ML in the frame of proposed techniques in the future.

8. Conclusion

Due to the complexity and size of HPC systems, the prediction of central issues of poor performance is a significant and challenging. In this paper, building an efficient ML model has been proposed, which can be used for classifying observations at runtime and predict the failures. For this purpose, offline training over many applications with the same number of nodes is conducted to gather information about the performance and resource usage of the nodes into a dataset. This dataset is used for calculating the statistical features and consequently label the nodes as 'success' or 'fail.' The left job is the training ML algorithm, which accepts these labels and features as input data for building a generic ML model. It is expected that this final model will be efficient at avoiding failures that force the application performance dramatically. Besides, by applying reinforcement learning of ML in the future, it is possible to make this model much stronger to keep it powerful against any improved exascale HPC system.

References

Bhatele, A., Titus, A. R., et al. (2015, May). Identifying the culprits behind network congestion. In *2015 IEEE International Parallel and Distributed Processing Symposium* (pp. 113-122). IEEE.

Bhowmick, S., Eijkhout, V., Freund, Y., Fuentes, E., & Keyes, D. (2006). Application of machine learning to the selection of sparse linear solvers. *Int. J. High Perf. Comput. Appl. (submitted)*

Dean, J. (2017, December). Machine learning for systems and systems for machine learning. In *Presentation at 2017 Conference on Neural Information Processing Systems.*

ExaLearn Project to bring Machine Learning to Exascale. Retrieved from: https://insidehpc.com/2019/03/exalearn-project-to-bring-machine-learning-to-exascale/

Fu, S. (2011, December). Performance metric selection for autonomic anomaly detection on cloud computing systems. In *2011 IEEE Global Telecommunications Conference-GLOBECOM 2011* (pp. 1-5). IEEE.

Guo, J., Nomura, A., Barton, R., Zhang, H., & Matsuoka, S. (2018, March).

Machine learning predictions for underestimation of job runtime on HPC system. In *Asian Conference on Supercomputing Frontiers* (pp. 179-198). Springer, Cham.

Huck, K. A., & Malony, A. D. (2005, November). Perfexplorer: A performance data mining framework for large-scale parallel computing. In *SC'05: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing* (pp. 41-41). IEEE.

Ibidunmoye, O., Hernández-Rodriguez, F., & Elmroth, E. (2015). Performance anomaly detection and bottleneck identification. *ACM Computing Surveys (CSUR), 48*(1), 1-35.

Islam, T. Z., Thiagarajan, J. J., et al. (2016, November). A machine learning framework for performance coverage analysis of proxy applications. In *SC'16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (pp. 538-549). IEEE.

Jha, S., & Fox, G. (2019, September). Understanding ML Driven HPC: Applications and Infrastructure. In *2019 15th International Conference on eScience (eScience)* (pp. 421-427). IEEE.

Netti, A., Kiziltan, Z., et al. (2019, August). Online fault classification in hpc systems through machine learning. In *European Conference on Parallel Processing* (pp. 3-16). Springer, Cham.

Sukhija, N., Malone, B., Srivastava, S., Banicescu, I., & Ciorba, F. M. (2014). A learning-based selection for portfolio scheduling of scientific applications on heterogeneous computing systems. *Parallel and Cloud Computing, 3*(4), 66-81.

The Exascale Era Is Coming, And Here's Why It Matters. Retrieved from: https://www.forbes.com/sites/forbestechcouncil/2019/10/24/the-exascale-era-is-coming-and-heres-why-it-matters/#1200517fbce0

Thiagarajan, J. J., Anirudh, R., et al. (2018, May). Paddle: Performance analysis using a data-driven learning environment. In *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)* (pp. 784-793). IEEE.

Tuncer, O., Ates, E., et al. (2017, June). Diagnosing performance variations in HPC applications using machine learning. In *International Supercomputing Conference* (pp. 355-373). Springer, Cham.

Tuncer, O., Ates, E., et al. (2018). Online diagnosis of performance variation in HPC systems using machine learning. *IEEE Transactions on Parallel and Distributed Systems, 30*(4), 883-896.

Why We Need Exascale Computing. Retrieved from: https://www.huffpost.com/author/acmblog-222

Yeom, J. S., Thiagarajan, J. J., et al. (2016, November). Data-driven performance modeling of linear solvers for sparse matrices. In *2016 7th International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS)* (pp. 32-42). IEEE.

Zhang, H., You, H., Hadri, B., & Fahey, M. (2012). HPC usage behavior analysis and performance estimation with machine learning techniques. In *Proceedings of*

the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA) (p. 1). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp)

Submitted: 08.06.2020
Accepted: 27.11.2020

.