Computing, Telecommunication and Control, 2024, Vol. 17, No. 3, Pp. 9–21.
Информатика, телекоммуникации и управление. 2024. Том 17, № 3. С. 9–21.

Applied problem solving with machine learning
Решение прикладных задач методами искусственного интеллекта

Research article
DOI: https://doi.org/10.18721/JCSTCS.17301
UDC 004

EXO-INTELLIGENT HYBRID SUPERCOMPUTER PLATFORMS FOR SHARED-USE CENTERS

V.S. Zaborovsky, L.V. Utkin , V.A. Muliukha ✉

Peter the Great St. Petersburg Polytechnic University,
St. Petersburg, Russian Federation

✉ vladimir.muliukha@spbstu.ru

Abstract. The article discusses the possibilities of increasing the real performance of hybrid supercomputer platforms consisting of different types of processor nodes (CPU, GPU, FPGA) operating in the mode of shared-use computational resources. The conceptual difference of the proposed approach from widespread supercomputing cluster platforms can be metaphorically expressed as "Less Moore, more brain." The considered approach shifts the focus of technology development from classical methods of increasing the performance of HPC platforms by adding new hardware multi-core computing components to more complex exo-intelligent solutions that use inductive (internal) and conceptual (external) data to implement machine learning methods for the purpose of optimally distributing available hardware resources between different classes of user applications. The proposed three-level architecture of hybrid computing platforms opens up new opportunities both for efficient scaling of user program execution processes, and for reification of descriptions of new algorithms by generating corresponding texts of computer programs, as well as interpreting the results obtained based on the use of statistical information, the carrier of which is censored data characterizing the experience of executing user applications in the mode of shared use of hybrid computational resources.

Keywords: high performance hybrid computing systems, machine learning, scheduler, survival function, explainable artificial intelligence

Acknowledgements: The research was financially supported by the Ministry of Science and Higher Education of the Russian Federation within the framework of the state assignment "Development and research of machine learning models for solving fundamental problems of artificial intelligence in the fuel and energy complex" (FSEG-2024-0027).

Citation: Zaborovsky V.S., Utkin L.V., Muliukha V.A. Exo-intelligent hybrid supercomputer platforms for shared-use centers. Computing, Telecommunications and Control, 2024, Vol. 17, No. 3, Pp. 9–21. DOI: 10.18721/JCSTCS.17301

Научная статья
DOI: https://doi.org/10.18721/JCSTCS.17301
УДК 004

ЭКЗО-ИНТЕЛЛЕКТУАЛЬНЫЕ ГИБРИДНЫЕ СУПЕРКОМПЬЮТЕРНЫЕ ПЛАТФОРМЫ ЦЕНТРОВ КОЛЛЕКТИВНОГО ПОЛЬЗОВАНИЯ

В.С. Заборовский, Л.В. Уткин , В.А. Мулюха ✉

Санкт-Петербургский политехнический университет Петра Великого,
Санкт-Петербург, Российская Федерация

✉ vladimir.muliukha@spbstu.ru

Аннотация. В статье рассматриваются возможности повышения реальной производительности гибридных суперкомпьютерных платформ, состоящих из процессорных узлов различных типов (CPU, GPU, FPGA), работающих в режиме совместного использования вычислительных ресурсов. Концептуальное отличие предлагаемого подхода от широко распространенных суперкомпьютерных кластерных платформ можно метафорически выразить как “Меньше Мура, больше мозга”. Рассматриваемый подход смещает фокус развития технологий с классических методов повышения производительности HPC-платформ путем добавления новых аппаратных многоядерных вычислительных компонентов на более сложные экзо-интеллектуальные решения, использующие индуктивные (внутренние) и концептуальные (внешние) данные для реализации методов машинного обучения с целью оптимального распределения доступных аппаратных ресурсов между различными классами пользовательских приложений. Предложенная трехуровневая архитектура экзо-интеллектуальных вычислительных платформ обладает новыми широкими возможностями как для эффективного масштабирования процессов выполнения пользовательских программ, так и для овеществления описаний новых алгоритмов путем генерации соответствующих текстов компьютерных программ, а также интерпретации полученных результатов на основе использования апостериорной статистической информации, носителем которой являются цензурированные данные, характеризующие опыт выполнения пользовательских приложений в режиме совместного использования гибридных вычислительных ресурсов.

Ключевые слова: высокопроизводительные гибридные вычислительные системы, машинное обучение, интеллектуальный диспетчер, функция выживаемости, объяснимый искусственный интеллект

Финансирование: Исследование выполнено при финансовой поддержке Министерства науки и высшего образования Российской Федерации в рамках государственного задания «Разработка и исследование моделей машинного обучения для решения фундаментальных задач искусственного интеллекта в топливно-энергетическом комплексе» (FSEG-2024-0027).

Для цитирования: Zaborovsky V.S., Utkin L.V., Muliukha V.A. Exo-intelligent hybrid supercomputer platforms for shared-use centers // Computing, Telecommunications and Control. 2024. Т. 17, № 3. С. 9–21. DOI: 10.18721/JCSTCS.17301

The source of coming-to-be for exiting things is that into which destruction, too, happens according to necessity

Anaximander of Miletus, 610–546 B.C.

Introduction

Currently, the development and application of digital artificial intelligence systems are experiencing a period of rapid development [1–3], changing technological reality, clarifying the fundamental principles of computer science and supplementing computer programming paradigms with new mechanisms for the

interaction of natural and artificial intelligence [4]. It is well known that methods of digital modeling of physical processes and various types of intellectual activities associated with complex mathematical calculations are carried out using digital computers and are performed many times faster and with greater accuracy compared to human capabilities. At the same time, the cost of electronic memory systems is rapidly decreasing and the performance of digital microprocessors, called "graphics" accelerators, is steadily increasing, contributing to a shift in the emphasis of the development of computer technologies [4, 5] from digital modeling methods of complex physical processes towards probabilistic models, processing of graphic information and the formation of multimodal texts explaining the results of calculations [6, 7]. The ongoing changes caused an expansion of ideas about algorithms based on the ideas of computability of functions, enumerability and solvability of sets [8, 9], which stimulated the search for solutions to applied problems of a probabilistic nature, presented using semantic invariants, large linguistic models or topological data analysis methods. The processes of exponential acceleration and diversification of computer technologies were associated with the development of pre-trained transformer models (Generative Pre-trained Transformer, GPT), which have built-in attention mechanisms [10], which de facto became a bifurcation point in the evolutionary development of computer science in general. In the dialectical spiral of improving computer technologies, there has been a transition from digital platforms for the implementation of the Pythagoras's ancient formulation that "everything is number" to the reification of the meaning of the sacred phrase that "In the beginning was the Word" [John 1:1]. The bifurcation turn that occurred in the development of computer science from the concept of programming to the learning paradigm opens up new prospects for the use of digital intelligent technologies in various areas of human activity. One of the areas of application of machine learning methods that has not yet received sufficient attention is the intelligent control and "self-learning" of the computing platforms themselves, including the tasks of planning access to multimodal data stores and explaining the results of calculations.

The search for new ways to increase the real performance of computing platforms has become especially relevant in modern conditions, when the so-called "Moore's law", which for many years regulated of chip's performance and energy consumption, stopped working[1], and the time frame for the creation of quantum computers has still not been determined [11]. The article discusses the possibilities of using machine learning technologies at the "tactical level" to increase real productivity based on planning application program processing processes using resource managers of hybrid supercomputer platforms consisting of nodes of various types (CPU, GPU, FPGA) and operating in shared-use resource modes. The difference between the platforms being developed and well-known solutions based on supercomputer clusters is the presence in them of an "introverted" machine learning system for the application processes dispatcher. The hierarchical architecture of such a platform consists of three levels:

1) planning the load of computing nodes, taking into account the parameters of user requests and a labeled set of data on the results of previously performed calculations;

2) machine learning models used to assess the influence of internal and external (exo) factors on the state of the cluster's computational resources and the success of application programs;

3) explanations/interpretation of the results of performing applied tasks.

The article substantiates an attempt to materialize the ideas of digital intellectualization, for which a description of computing processes is introduced based on the "survival" function of various classes of applied tasks being performed. This processing makes it possible to quantitatively characterize the real performance of a computing platform not by counting the number of machine operations performed, but by counting the number of successfully completed applied tasks in relation to the total number of executable programs. The introduced function allows us to evaluate the measure of "usefulness" of the dispatcher's work in terms of the probability of successful execution of certain classes of applied tasks, subject to maximizing the load of all hybrid nodes of the computing cluster. To construct the survival function of applied tasks of a specific platform, it is proposed to use information about the "credit history"

[1] Moore's law – Wikipedia, Available: https://en.wikipedia.org/wiki/Moore%27s_law#cite_ref-30 (Accessed 09.10.2024)

of computing in relation to various types of application programs and categories of registered users. The data used to calculate this function includes the values of applied task descriptors, the number and type of computing nodes, the number of successfully or unsuccessfully completed tasks during the time interval allocated for task execution by the dispatcher, etc. The article shows how, using machine learning algorithms for models that predict the amount of time interval sufficient to successfully complete tasks and methods for explaining the results obtained (eXplainable Artificial Intelligence, XAI), it is possible not only to identify, but also to quantify various factors influencing probabilistic characteristics of successful completion of applied computing processes. The results of explanations "computed" in this way, along with the results of executing the applied tasks themselves, are available not only to users, but also to the computational resource manager, which allows increasing the level of real productivity for the entire exo-intelligent supercomputer platform. In this case, technologies for replenishing the functionality of existing computers and exo-mechano/energy platforms with additional exo- intelligent capabilities are in demand and relevant. The use of a combination of high performance computing (HPC) technologies and big data processing algorithms with intuitive principles of finding solutions and planning actions under conditions of uncertainty can be considered as a process of conceptual validating idea of "digital of intelligence". The essence of the idea is that any observable or measurable physical process must a priori have some formal description that allows for the implementation of a number of exo- intelligent operations, namely:

- copy results in the form of a symbolic code, which opens up new opportunities for storing and transferring knowledge;
- process information at high speed and constantly improve processing algorithms through machine learning processes.

However, on the way to implementing the hypothesis, both fundamental and technological problems arise: modern digital systems and software technologies are not capable of modeling processes that do not have a formal mathematical description, for example, cognitive processes. Without going into theological details, it can be noted that the need for the hypothesis of digital intelligence presupposes the existence of an a priori formal description of the observed processes and, as a consequence, the existence of an environment capable of not only generating such a description, but also implementing mechanisms for translating (reifying) this description into a real physical process.

The article examines all the questions formulated above using the example of the hybrid computing platform of the Polytechnic supercomputer, operating in the mode of a shared-use resource center. The article proposes a description of complex computing platforms using the survival function [8] of applied tasks, which characterizes the probability of successful execution of applied tasks based on the use of censored data [11] on the functioning of the computing system as a whole under the control of a resource manager.

The approach proposed in the article to describing computational processes in hybrid computing platforms is based on the principle of multiple machine learning models (an analogue of the physical principles of complementarity) used to estimate the time interval sufficient for the successful completion of various classes of applied tasks [8]. The pre-trained models are used by the dispatcher to quickly schedule the load of computing nodes in such a way as to increase the likelihood of successful completion of all applied tasks. The principal ability to solve machine learning problems in such system is based on the experimentally established fact of statistical stability (Fig. 1) of the observed processes, characterizing the proportion of applied tasks, the completion of which is marked by the system dispatcher with a success code of "0".

The data presented in Fig. 1 reflect the objectively existing features of hybrid supercomputer platforms operating in the mode of shared-use centers. These features are available for mathematical description in order to model the processes occurring and explain the results obtained. The obtained explanations of the results are local in nature and are based on an assessment of the influence coefficients of various

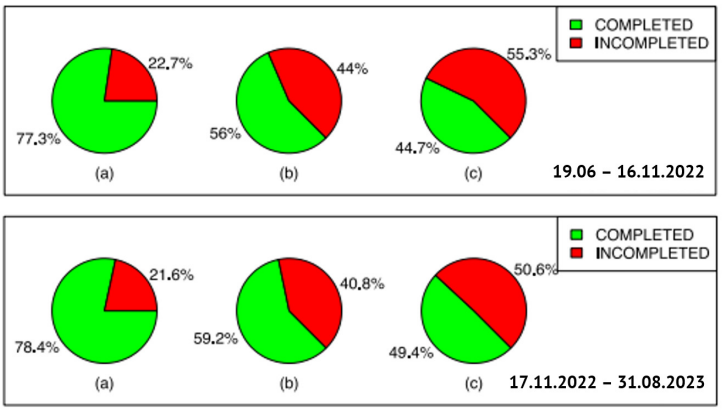

Fig. 1. (a) number of user tasks; (b) time spent; (c) CPU time

factors characterizing the operation of the supercomputer platform as a whole from the point of view of the successful execution of various classes of applied tasks. Taking into account the above, the article proposes a constructive extension of the measure of real performance of a supercomputer platform as the proportion of successfully solved problems to the total number of applied tasks completed during the observation time interval. The application of this performance indicator in practice comes down to solving the inverse problem of algorithmic modeling of "big data", taking into account the locality of the digital and linguistic variables used [12], as well as various attention mechanisms [9], including various methods of information retrieval, for example, based on the so-called inverted index algorithms. Replenishing the architecture of a computer platform with machine learning mechanisms leads to the fact that exo-intelligent systems no longer "suffer" from large volumes of processed information, but, on the contrary, acquire fundamentally new opportunities to increase their real productivity by accumulating experience in solving applied tasks in combination with "inverse" methods. » transformation of calculation results into process planning algorithms and explanations of the results obtained.

It should be noted that classical methods for solving inverse problems, developed by academician A.N. Tikhonov [13], are effective for situations, where possible relationships between solutions of computational problems and input data have an analytical representation, for example, in the form of physical laws. In the situation in which machine learning methods are proposed to used, the analyticity description of such representation is not available due to the combinatorial complexity of the process of selecting "inverse" solutions and the uncertainty of the formal description of the input data corpus. Taking into account this obstacle, the article presents a new approach to the regularization of solutions, obtained using machine learning methods and based on local interpretable model of the observable computational processes. Such interpretable model characterizes survival functions of various classes of user tasks that should be similar to those functions that were obtained experimentally on over large observation intervals [14].

Polytechnic SCC as a Machine Learning Hybrid Platform

The high-performance hybrid supercomputer platform of the "Polytechnic supercomputer center" (Polytechnic SCC) is a highly complex technical system that simultaneously performs a quadrillion (a number with 15 zeros) machine operations per second, which is used to solve various types of user tasks and algorithms that solve computational problems of various complexity classes. Supercomputer users, whose number is steadily growing every year, are interested in the successful result of their calculations, and not interested in the peak performance, which is nominally expressed in the number of floating-point arithmetic operations per second. All registered users get access to SCC resources from the dedicated server using the terminal client protocol. SCC resources are managed using the Slurm dispatcher (Fig. 2): the

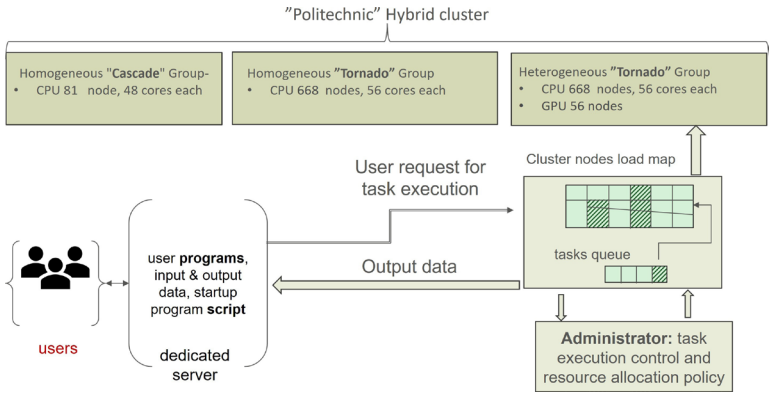

Fig. 2. Structure and data flows of the Polytechnic SCC

user requests some resource (processor cores, memory, etc.), placing his task in the queue; the system, based on the user's priorities and the current filling of the queue, selects the moment of task launch. A queue is a sequence of tasks that must be solved on a specific computational resource (a group of nodes). At the same time, each node at the current time can be occupied by only one task of a user. Thus, the node is assigned to the exclusive use of the hosted tasks, and other tasks on the busy node will not be executed.

All tasks can be divided into classes depending on the areas of expertise: Astrophysics, Bioinformatics, Biophysics, Energetics, Geophysics, IT, Mechanical engineering, Mechanics, Physics and Radiophysics. In Fig. 3 each of the classes is indicated by its own color and occupied one or several nodes of hybrid cluster: IT tasks are black, Astrophysics tasks are turquoise, Bioinformatics tasks are red, Biophysics tasks are pink, Energetics tasks are green, Geophysics tasks are yellow, Mechanical engineering tasks are blue, Mechanics tasks are gray, Physics tasks are lime, Radiophysics tasks are orange, Uncertain tasks are purple, and white indicates node idleness.

By analyzing Fig. 3, we propose two slogan that explain relationship between machine learning (ML) and HPC platform, namely "A frontier ML system is HPC", and vice versa "HPC is a frontier ML system". These slogans have not become a truism yet, but they clearly reflect the ideas of computer evolution tendency – frontier HPC systems become not only the driving force of digital transformation process, but also an active part of machine learning ecosystem. The Fig. 3 clearly shows that during runtime SCC activities some cluster nodes are idle for a long time, since there are no applied tasks in the queue that can use free computational resources upon user request. The Fig. 4 reflects present of different tasks completed status.

Fig. 4 shows the execution status of various user tasks. Although the cluster nodes are formally busy, the analogue of the metric known as efficiency in such a cluster is quite low, which clearly indicates the need to find ways to improve real productivity. Explanation of why the user task "survived" or why it was not completed within the time specified by the scheduler can speed up the SCC performance, user's understanding of the specifics of supercomputer platform and can be used to predicate duration time which is needed to successfully complete all of user tasks. Obviously, the predictive ability for the dispatcher is an exo-intelligent function that increases the likelihood of successful completion of an applied task, as well as "preparing" the cluster's computational resources for processing new tasks. There are several well-known models used in survival analysis. First, we should mention the nonparametric Kaplan–Meier approach for estimating survival functions, which describe the properties of individual components that influence the assessment of the quality of functioning and the risks of changing the behavior of complex systems. But survival functions estimator gives only the average view of tasks features and does not take into account the individual parameters $\mathbf{x}$ of each task. Therefore, based only on survival functions it is difficult to evaluate how the specific task features impact on real performance of

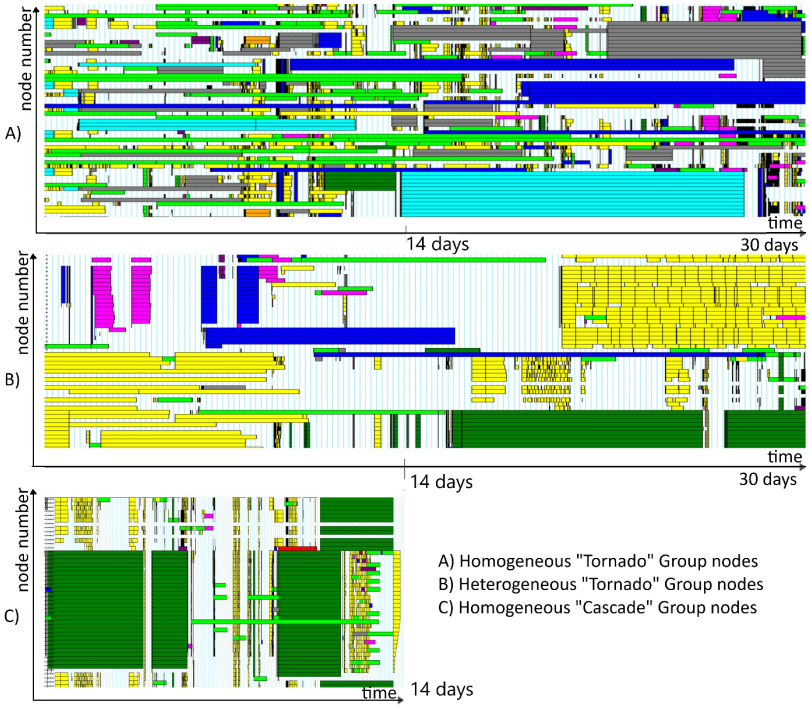

Fig. 3. Uneven loading of cluster nodes

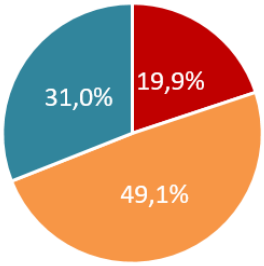

Fig. 4. Completed status of different tasks: orange sector (49,1%) – runtime errors;
blue sector (31%) – task completed successfully; red (19,9%) – tasks removed by scheduler

SCC. The approach which allows to estimate the survival and cumulative hazard feature depending the vector **x** parameter of each task is based on Cox model that is defined as

$$h(t|\mathbf{x}) = h_0(t)\exp^{(\mathbf{x}\mathbf{b}^T)}, \quad (1)$$

where $h_0(t)$ is an arbitrary baseline hazard function and $\mathbf{b} = (b_1, \ldots, b_m)$ is an unknown vector of regression coefficients or model parameter.

As follows from Fig. 5, to solve the problem of optimizing the structure of the user tasks queue, we need to use machine learning scheduler system that is implemented to predict the execution time period, taking into account the current load of the SCC nodes and previous experience. Intelligent scheduler can identify:

- spread of the estimated execution time value,
- imbalance of tasks across the ranges of duration time to which they belong,
- insufficient amount of information in the available factors.

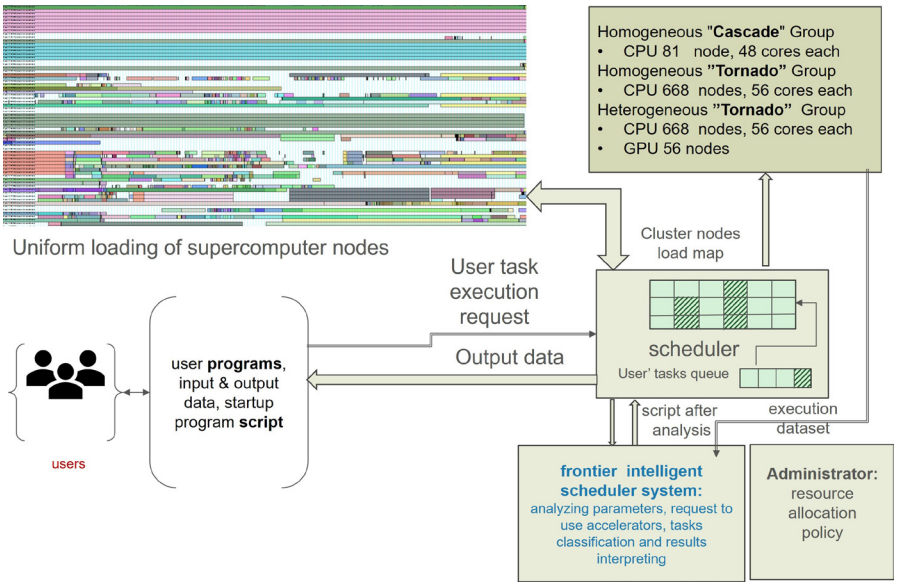

Fig. 5. "Less Moore, more brain": Polytechnic SSC – frontier intelligent platform

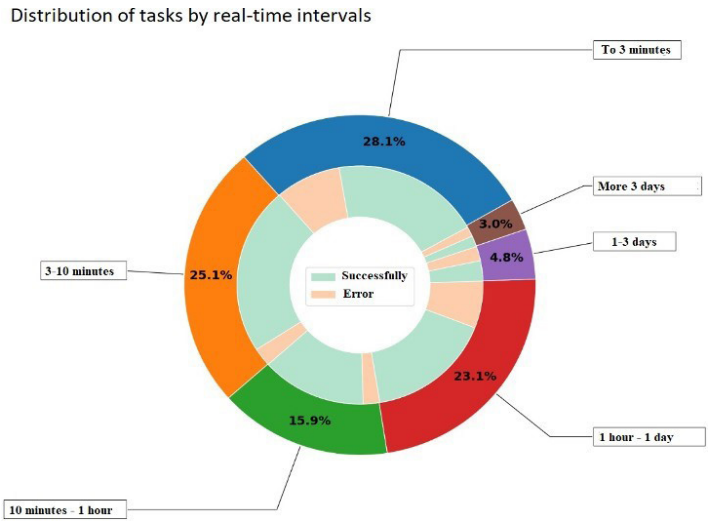

Fig. 6. The distribution of tasks in accordance with their completion time

The survival function analysis processes depicted in Fig. 6 shows that there are no real obstacles to the cluster node load being evenly distributed, because more than 89% of all tasks with actual duration between 0 and 10 minutes indicate an approximate execution time between 1 hour and 15 days. The distribution of tasks in accordance with their completion time is shown in Fig. 6. Therefore, users significantly overestimate the time it takes to complete applied tasks. User's behavior as well as tasks feature strongly affect task execution time and should be considered as a main factor of machine learning scheduler system.

It should be noted that the implementation of the intelligent component does not affect the work of the scheduler Slurm. The module for assessing the time of successful task execution is built in between the user and the scheduler and adjusts the parameters of the launched task in accordance with the forecast of the execution time. This ensures a denser queue load, reduces the average waiting time and increases the probability of successful completion of the computing task in the specified time.

Survival function in SCC real performance analysis

We represent a dataset D of survival analysis as a set of triplets of the form $(\mathbf{x}_i, \delta_i, T_i)$, where $\mathbf{x}_i = (x_{i1}, \ldots, x_{im})$ is the feature vector which contains all available information about the i-th user task represented by m features; T_i is the i-th task completion time. In contrast to the standard regression analysis, proposed model includes additional component δ_i which is the indicator function, so that $\delta_i = 1$, if we observe a successful task completion, and $\delta_i = 0$, if the i-th task has not been successfully completed. In the first case $(\delta_i = 1)$, time T_i corresponds to the time between the baseline time and the time of the successful task completion. This case corresponds to the uncensored observation. In the second case $(\delta_i = 0)$, T_i is the observation time, i.e. the moment when the task is terminated due to several reasons, and we have the censored observation. The aim of survival analysis is to predict the completion time of a new task characterized by the feature vector $\mathbf{x}$ by using the training dataset consisting of n examples $(\mathbf{x}_i, \delta_i, T_i)$, $i = 1, \ldots, n$. Basic functions of survival analysis are the survival and hazard functions. The survival function, denoted as $S(t|\mathbf{x})$, reflects probability that the task $\mathbf{x}$ is not completed up to the time t, so $S(t|\mathbf{x}) = \Pr\{T > t|\mathbf{x}\}$. The hazard function $h(t|\mathbf{x})$ is the rate of the task completion at time t given that no tasks completed before time t. It can be written as follows:

$$h(t|\mathbf{x}) = \lim_{\Delta t \to 0} \frac{\Pr\{t \le T \le t + \Delta t | T \ge t|\mathbf{x}\}}{\Delta t} = \frac{f(t|\mathbf{x})}{S(t|\mathbf{x})}, \tag{2}$$

where $f(t|\mathbf{x})$ is the density function of the task completion.

The hazard function can also be expressed though the survival function as follows:

$$h(t|\mathbf{x}) = \frac{d}{dt} \ln S(t|\mathbf{x}). \tag{3}$$

Hence, we can express the survival function through the cumulative hazard function $H(t)$ as follows:

$$S(t|\mathbf{x}) = \exp(-H(t|\mathbf{x})). \tag{4}$$

It can be seen from above examples that the functions depend on the vector $\mathbf{x}$.

The survival function $S(t|\mathbf{x})$ is computed as follows:

$$S(t|\mathbf{x}) = (S_0(t)) \exp(\mathbf{x}\mathbf{b}^T). \tag{5}$$

Here $S_0(t)$ is the baseline survival function. It is important to point out that $S_0(t)$ as well as $h_0(t)$ do not depend on $\mathbf{x}$ and are estimated using the Kaplan–Meier estimator. The main peculiarity of the Cox model is the linear combination of features. On the one hand, it simplifies the model and allows us to use it in the interpretation of the model predictions. On the other hand, it restricts the Cox model use because real datasets usually have a more complex structure. Various modifications have been proposed to generalize the Cox model by replacing the linear relationship with some non-linear functions, for example, with neural networks or the support vector machine. A goal of the machine learning algorithm is to predict the probabilistic characteristics of the task completion, including the survival function and the task completion time. In the case when the dataset of examples is restricted, it is possible to apply model of random survival forest as one of the efficient methods dealing with the limited number of training data. The random survival forest consists of Q survival trees define by vector $\mathbf{x}$ that consists of m features which include the most important for scheduler data: UserID (the ID of a user), GroupID (the ID of the group on behalf of which the task was queued), NumNodes (the number of nodes requested or allocated for the task), NCPUs (the total number of processors allocated to the task), NumTasks (the total

number of subtasks in the task), CPUs/Task (the ratio of the total number of processors to the number of subtasks), ReqB:S:C:T (the number of different hardware components requested for the task), Socks/Node (the desired ratio of the number of sockets to the number of compute nodes), NtasksPerN:B:S:C (the number of subtasks required to run on a specific number of hardware components), CoreSpec (the number of cores reserved), MinCPUsNode (the minimum ratio of the number of processors to the number of nodes), MinMemoryNode (the minimum ratio of memory in MB to the number of nodes), JobID (the task identification number), Priority (he task priority determined by the SLURM scheduler), etc. It should be noted that the completion time T_i of different tasks is changed in a large interval of time. It can be seen from this distribution that the number of "long" tasks is rather small in comparison with tasks completed in a short time. This causes a difficulty for random survival forest because survival trees are mainly trained on the "short" task and do not take into account the "long" duration. In order to overcome this problem, we propose to cluster all training dataset into K groups which are separated by using the completion time T as well as the feature vector $\mathbf{x}$. The completion time is more important factor in comparison with the feature vector because it determines the distribution of tasks. Therefore, we propose to introduce weights w_0 and w_1 of the completion time as well as the feature vector $\mathbf{x}$, respectively, so that $w_0 + w_1 = 1$, $w_0 > w_1$. In such weighted K-means clustering procedure, the distance between the centroid $\mathbf{c}$ of points from a cluster C_k and the vector $\mathbf{x}$ can be easily computed. One of the important problems to optimize the real performance is to explain why the user task is characterized by the obtained survival function or the expected time to the task completion. This problem can be referred to the XAI direction. Methods of XAI try to answer the question, which feature of an example significantly affect a prediction of a machine learning model. Most methods are explained locally, that is, they explain a prediction of a single example, and assume that the analyzed machine learning model is a black box which means that we know only its input and output data. A lot of explanation methods are based on local approximating the unknown prediction function at a point by means of the linear function of features because coefficients of the function can be regarded as quantitative impacts on the prediction.

In this case, the idea of the task description in terms of natural language requires the development of more complex and efficient structures based on the attention mechanism and transformers now attract strong attention. It should be noted that transformers proposed to solve the survival analysis problems can be solid platform for this. However, as a rule, transformers do not take the peculiarities and context of the problems, which are solved within completion time optimization problem. Therefore, productive approach is to combine the random survival forests and the transformer [12]. However, this approach cannot be directly used in survival analysis of user tasks in SSC. New approaches are needed to develop an efficient multi-modal transformer-based system based on estimation of system survives function (Fig. 7).

Perspectives

Obviously, multimodal transformers outline of the description of different applied tasks can be added to the machine learning ecosystem so that the user can conduct a dialogue with the supercomputer in terms of meaningful queries in the context of specific tasks features, using the capabilities of a pre-trained transformer (Fig. 8), which generates the source code of the task in response to the meaningful request. In such system user can analyze the accuracy of the formulated queries, conducting a meaningful interaction with a supercomputer, that inevitably improves his qualifications and task understanding. This approach has two advantages. First, it allows making the clustering procedure more flexible and enhance real cluster performance. Second, it allows supplementing the explanation procedure and developing multimodal transformer to solve the survival problems in context of scheduler efficiency. Moreover, the idea of the task description in terms of natural language requires the development of more complex and efficient SCC architecture based on machine learning mechanism. It is important to note that one of the perspective directions is to adapt the trained system to changes in the supercomputer structure, for example, to new additional computer blocks, which can be supplemented in runtime mode.

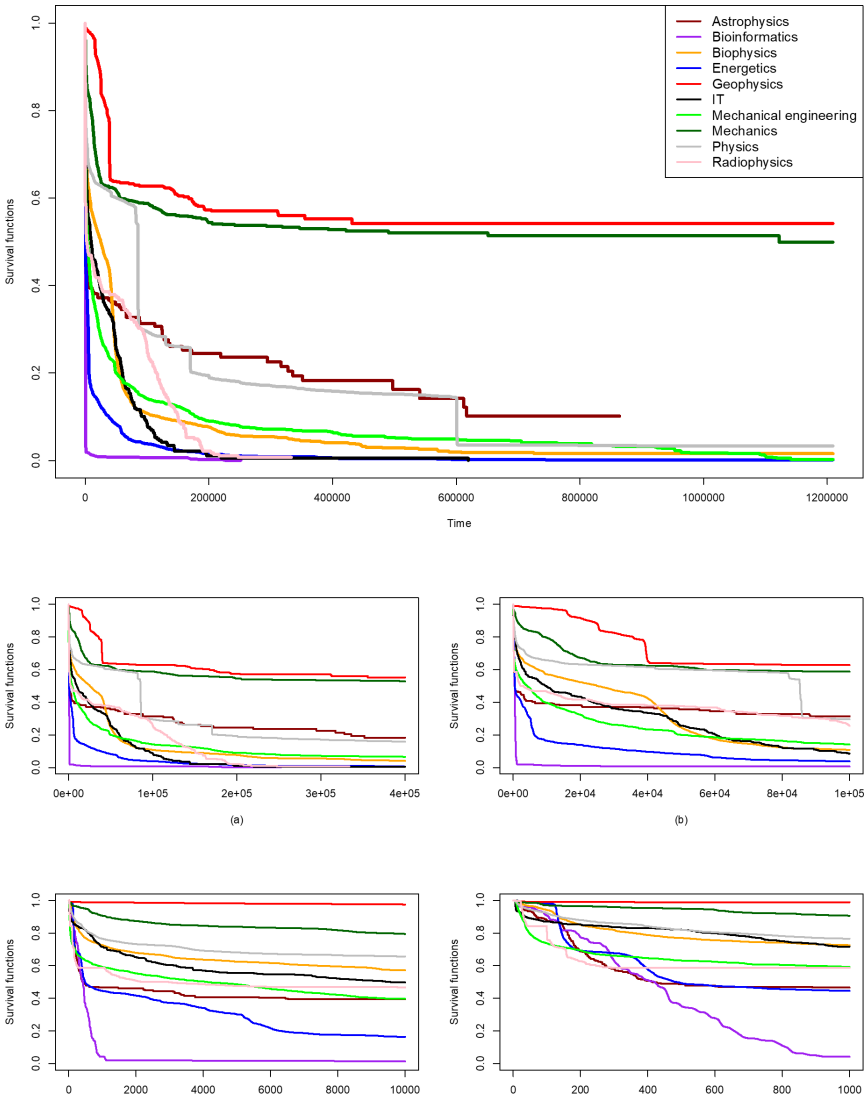

Fig. 7. Experimental survival task distribution functions: (a) the area of task execution times not exceeding 4×10^5 sec.; (b) not exceeding 1×10^5 sec.; (c) not exceeding 1×10^4 sec.; (d) not exceeding 1×10^3 sec.

The structure of a hierarchical hybrid HPC platform with an exo-intelligent subsystem for predicting the execution time of applied tasks and a built-in multimodal transformer that generates source code for new programs based on user requests is shown in Fig. 8. The results of applying machine learning methods to a model that estimates the execution time of an applied task and is used to improve the efficiency of the dispatcher are shown in Fig. 9.

From the Fig. 9 follows that the number of successfully completed applied tasks in Hierarchical Architecture of frontier Polytechnic SCC increased compared to the result of the system without subsystem estimating the time required to successfully compute user tasks (see Fig. 4).

Conclusions

The article discusses new way to increase the real performance of hybrid HPC platforms operating in the mode of shared-use computational resources. The conceptual difference of the proposed approach can be metaphorically expressed as “Less Moore, more brain.” The use of machine learning approach shifts the focus of methods of increasing the performance of HPC platforms from hardware components to more complex exo-intelligent solutions that use inductive (internal) and conceptual (external) data

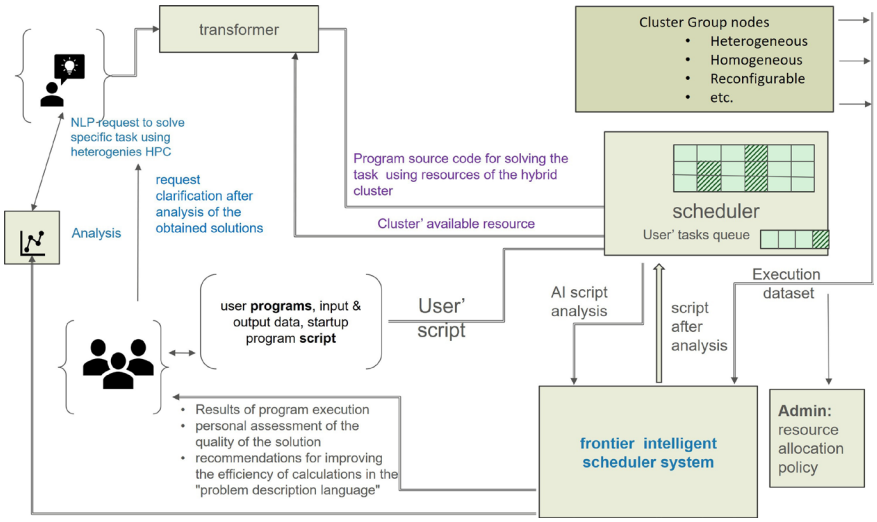

Fig. 8. Hierarchical architecture of frontier SCC "Polytechnic"

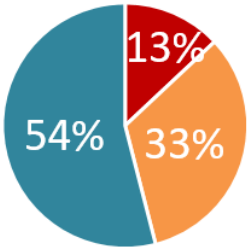

Fig. 9. Status of mechanics tasks: blue sector (54%) – task completed successfully; orange sector (33%) – runtime errors; red (13%) – tasks removed by scheduler

to implement machine learning methods for the purpose of optimally distributing available hardware resources between different classes of user applications. The proposed three-level architecture of hybrid computing platforms opens up new opportunities both for efficient scaling of user program execution processes, and for reification of descriptions of new machine learning algorithms by setting up an appropriate survival model, as well as interpreting the results of calculation based on analysis of statistical information.

REFERENCES

1. **Alaa A., van der Schaar M.** Limits of estimating heterogeneous treatment effects: Guidelines for practical algorithm design. Proceedings of the 35th International Conference on Machine Learning, PMLR 80, 2018, pp. 129–138.

2. **Hu S., Fridgeirsson E.A., van Wingen G., Welling M.** Transformer-based deep survival analysis. Proceedings of Machine Learning research, PMLR 1, 2021, pp. 1–17.

3. **Konstantinov A.V., Utkin L.V., Lukashin A.A., Muliukha V.A.** Neural attention forests: Transformer-based forest improvement. Proceedings of the Seventh International Scientific Conference "Intelligent Information Technologies for Industry" (IITI'23), 2023, Vol. 776, pp. 158–167. DOI: 10.1007/978-3-031-43789-2_14

4. **Lu J., Behbood V., Hao P., Zuo H., Xue S., Zhang G.** Transfer learning using computational intelligence: A survey. Knowledge-Based Systems, 2015, Vol. 80, pp. 14–23. DOI: 10.1016/j.knosys.2015.01.010

5. **Pachón-García C., Hernández-Pérez C., Delicado P., Vilaplana V.** SurvLIMEpy: A python package implementing SurvLIME. Expert Systems with Applications, 2024, Vol. 237, part C, article no. 121620. DOI: 10.1016/j.eswa.2023.121620

6. **Pölsterl, S., Navab N., Katouzian A.** An efficient training algorithm for kernel survival support vector machines, 2016. DOI: 10.48550/arXiv.1611.07054

7. **Ribeiro M.T., Singh S., Guestrin C.** "Why should I trust You?" Explaining the predictions of any classifier, 2016. DOI: 10.48550/arXiv.1602.04938

8. **Utkin L.V., Satyukov E.D., Konstantinov A.V.** SurvNAM: The machine learning survival model explanation. Neural Networks, 2022, Vol. 147, pp. 81–102. DOI: 10.1016/j.neunet.2021.12.015

9. **Wang P., Li Y., Reddy C.** Machine learning for survival analysis: A survey. ACM Computing Surveys (CSUR), 2019, Vol. 51, no. 6, pp. 1–36. DOI: 10.1145/3214306

10. **Wang Z., Sun J.** SurvTRACE: Transformers for survival analysis with competing events. Proceedings of the 13th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics, 2022, pp. 1–9. DOI: 10.1145/3535508.3545521

11. **Kanamori Y., Yoo S.-M.** Quantum computing: Principles and applications. Journal of International Technology and Information Management, 2020, Vol. 29, no. 2, pp. 43–71. DOI: 10.58729/1941-6679.1410

12. **Konstantinov A.V., Utkin L.V.** Interpretable machine learning with an ensemble of gradient boosting machines. Knowledge-Based Systems, 2021, Vol. 222, article no. 106993. DOI: 10.1016/j.knosys.2021.106993

13. **Tikhonov A.N.** Metody resheniya nekorrektnykh zadach [Methods for solving incorrect problems], Moscow: Nauka, 1979.

14. **Kovalev M.S., Utkin L.V., Kasimov E.M.** SurvLIME: A method for explaining machine learning survival models. Knowledge-Based Systems, 2020, Vol. 203, article no. 106164. DOI: 10.1016/j.knosys.2020.106164

15. **Cox D.R.** Regression models and life-tables. Journal of the Royal Statistical Society: Series B (Methodological). 1972, Vol. 34, no. 2, pp. 187–202. DOI: 10.1111/j.2517-6161.1972.tb00899.x

INFORMATION ABOUT AUTHORS / СВЕДЕНИЯ ОБ АВТОРАХ

Zaborovsky Vladimir S.
Заборовский Владимир Сергеевич
E-mail: vlad2tu@yandex.ru

Utkin Lev V.
Уткин Лев Владимирович
E-mail: lev.utkin@gmail.com
ORCID: https://orcid.org/0000-0002-5637-1420

Muliukha Vladimir A.
Мулюха Владимир Александрович
E-mail: vladimir.muliukha@spbstu.ru
ORCID: https://orcid.org/0000-0002-3583-7324

Submitted: 05.07.2024; Approved: 27.09.2024; Accepted: 04.10.2024.
Поступила: 05.07.2024; Одобрена: 27.09.2024; Принята: 04.10.2024.