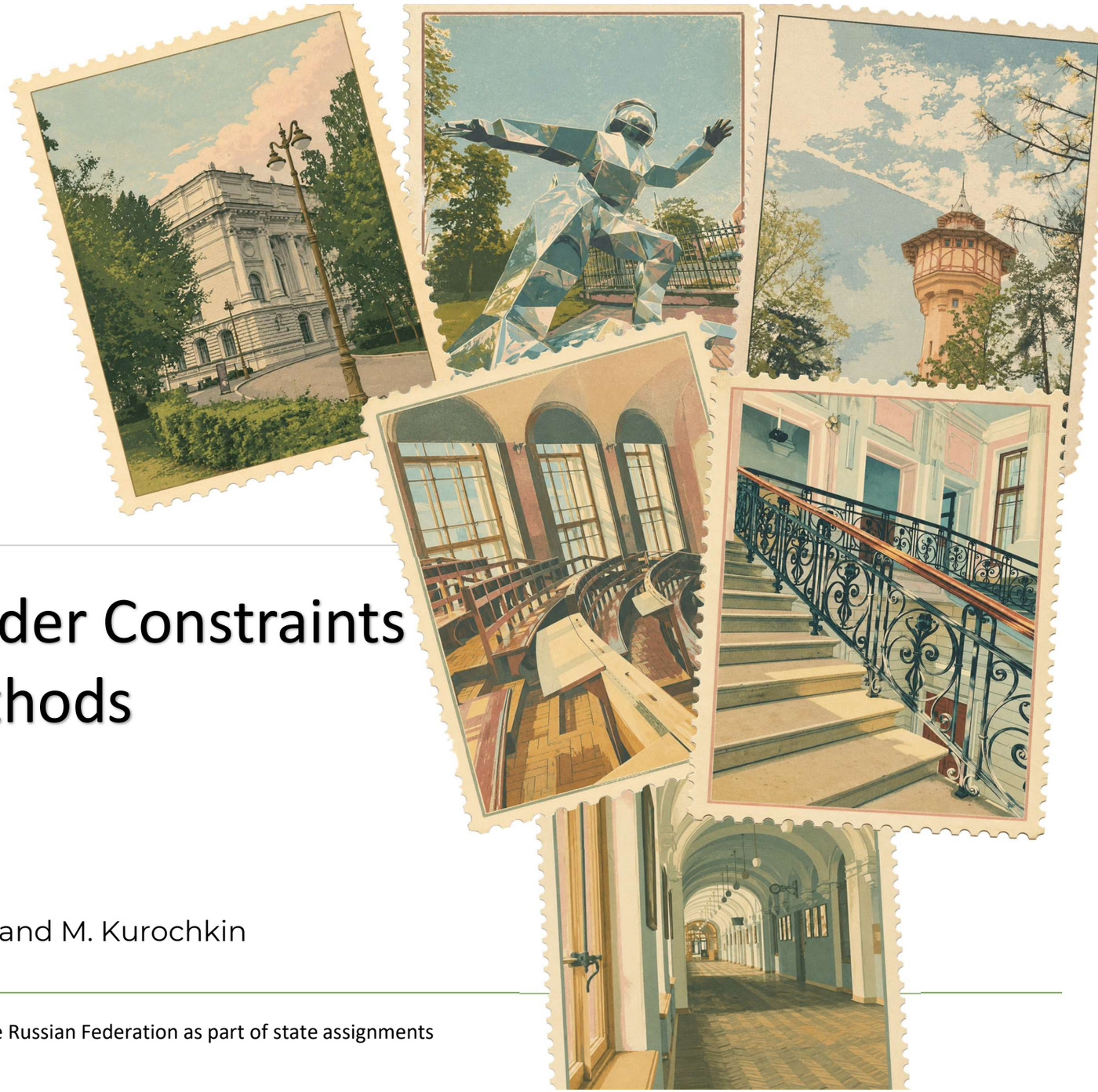
POLYTECH

Peter the Great St. Petersburg Polytechnic University



Solving Applied Problems under Constraints Using Machine Learning Methods

Vladimir Muliukha, A. Vostrov, D.Motorin, V. Glazunov, and M. Kurochkin

The research is partially funded by the Ministry of Science and Higher Education of the Russian Federation as part of state assignments (topic FSEG-2024-0027).

POLYTECH

Practical ML under constraints

- ML is transforming medicine, finance, manufacturing, and HPC.
- Real projects face:
    - scarce high-quality labeled data;
    - limited compute and energy;
    - strict latency and performance requirements;
    - integration with existing production systems;
    - trust, explainability, and user transparency.
- How do we deliver practical ML when resources are constrained?

Toolbox for constrained ML:

- **Small data:** transfer learning, few-shot learning, prototypical/matching networks.
- **Model efficiency:** quantization, pruning, knowledge distillation.
- **Fast inference:** early-exit architectures, asynchronous pipelines.
- **Edge/Green AI:** lower energy and memory footprint.
- **Production integration:** avoid disrupting existing workflows; use asynchronous processing.

Two case studies overview

Case 1 — 3D shape comparison for intellectual property protection

- Constraint: small labeled dataset, speed, invariance.
- Method: chord-length histogram descriptor + Siamese network.

Case 2 — Supercomputer task scheduling

- Constraint: inaccurate runtime estimates, no user workflow changes, SLURM integration.
- Method: asynchronous SurvNAM-based runtime prediction.

Common pattern:

- separate heavy preprocessing/modeling from the user-facing interaction;
- use expert/user feedback;
- integrate without breaking production.

POLYTECH

Case 1: Problem

- 3D models can be submitted as protected industrial objects.
- Need to compare 3D shapes and find close analogues.
- Applicants may change:
    - position;
    - rotation;
    - scale.
- Requirements:
    - invariance to position, rotation, scale;
    - fast comparison;
    - no need to transfer original model files;
    - work with limited labeled data.

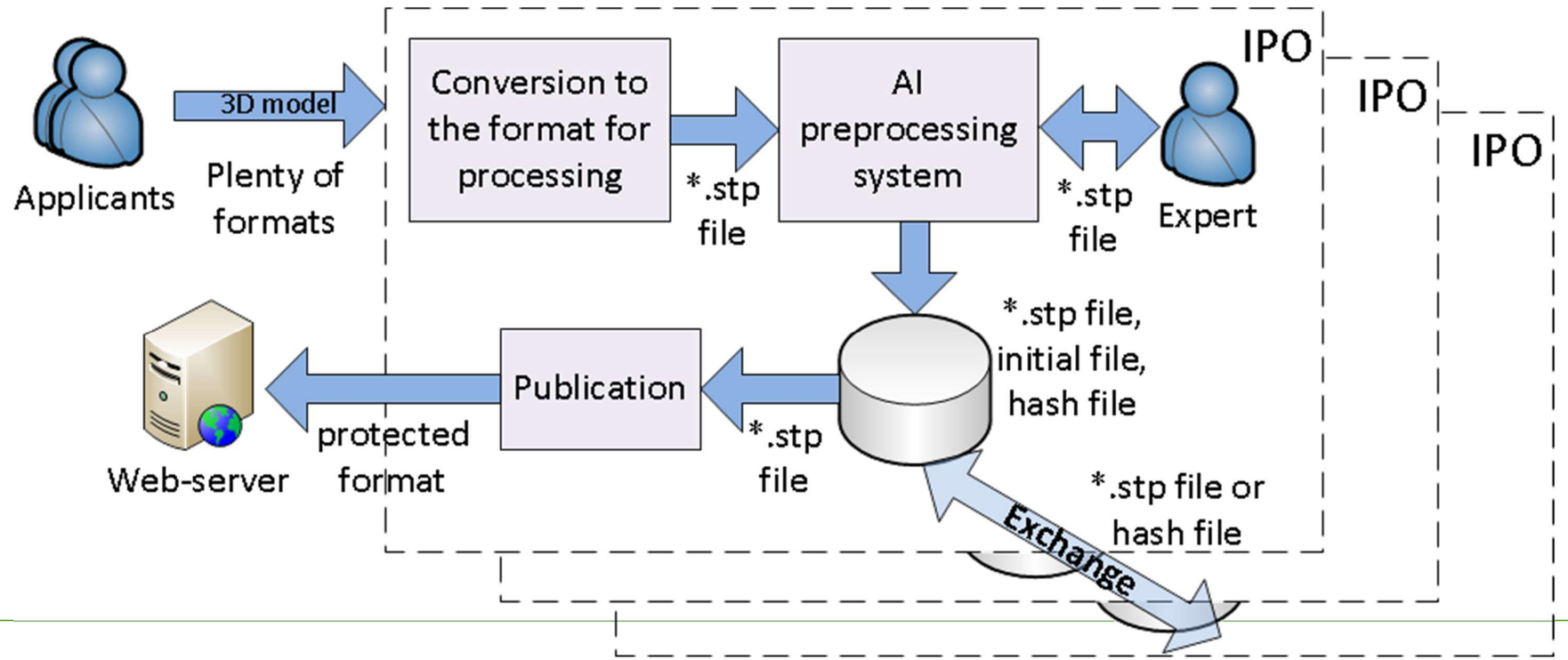
Case 1: Stages of 3D Model IPR Process

The 3D model represented format should reflects certain procedures and specific methods at each stages of application's life cycle, including:

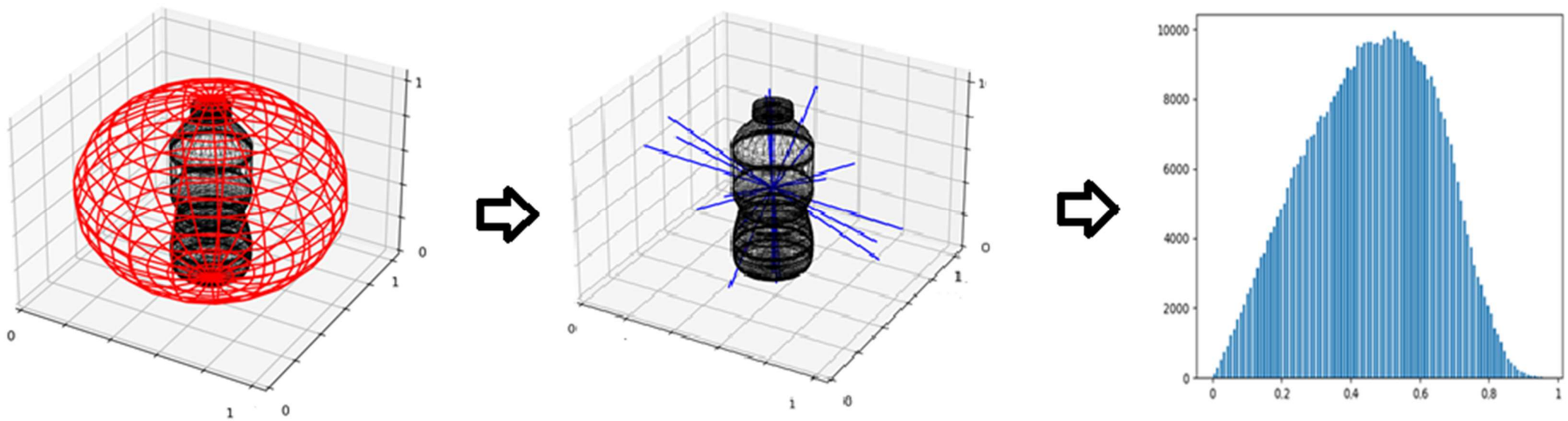
- **Creating** 3D model of IP object (at this stage applicants ***use existing CAD/CAE software*** development tools that support many different formats and standards)
- **Filing** of an application to IPO (***maximum acceptance*** of applications with ***uniqueness of conversion*** to a chosen format/formats)
- **Processing and Storage** (***unified format*** and ***uniform methods*** for automated application analysis)
- **Publication** of patent data (publication of data in an ***open and accessible format, excluding an extraction*** and using the model)
- **Further use or sale** (the ***most complete format*** with all the necessary information for using the model)

Case 1: Road Map of IP Application

Filing	Processing	Storage	Publication	Exchange
STL, STEP IGES, OBJ X3D, DWF, DWG, U3D X3D, 3DM	<STEP, initial file, hash/descriptor file for searching>		Protected format like 3Dpdf	STEP or hash/descriptor file for searching



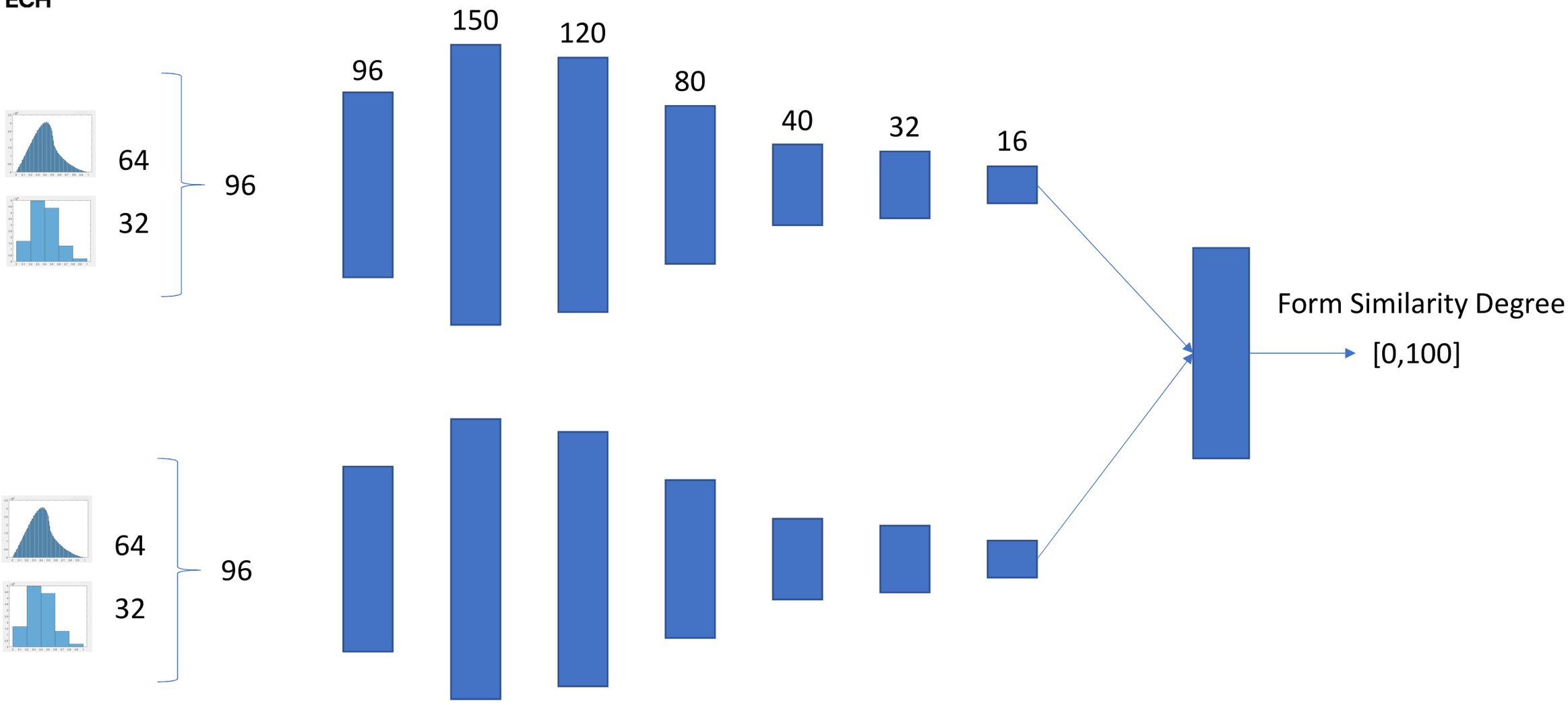
Case 1: Distance histogram as the shape invariant



The sequence of transformation steps from a three-dimensional figure to a distance histogram

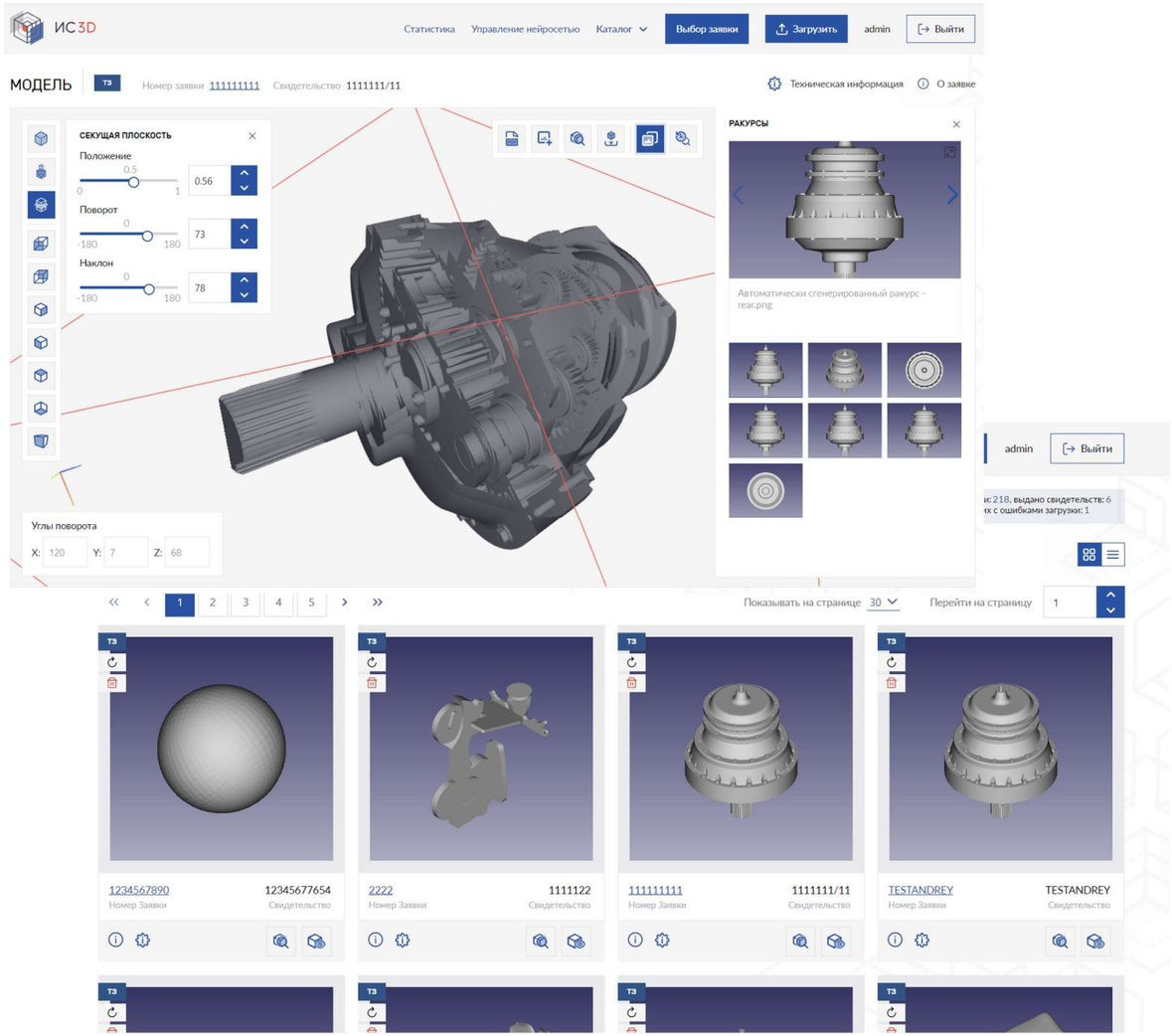
The combined vector of histograms with 32 and 64 bins serves as a shape descriptor for 3D objects

Case 1: Siamese neural network architecture



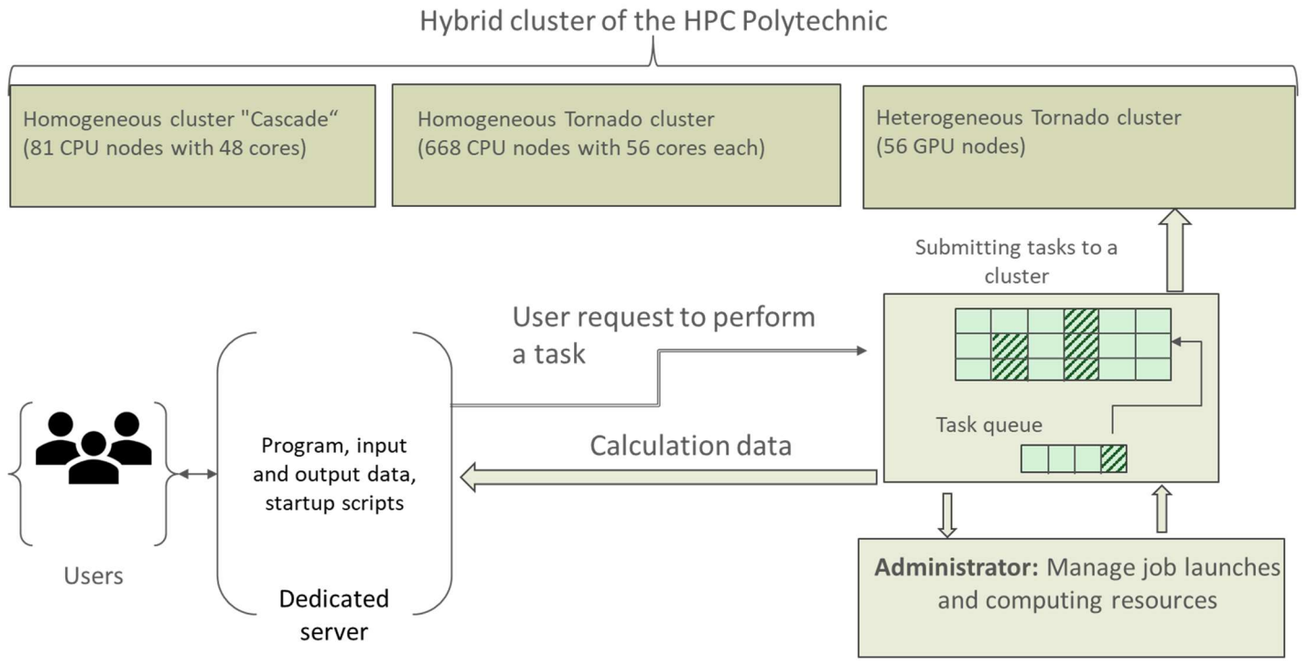
Case 1: Appearance of the program interface

- The descriptor is calculated in the background when a model is loaded, prior to it being sent to the expert.
- The expert performs only a comparison of the vectors that characterize the model's shape.
- When fine-tuning the Siamese network, there is no need to recalculate the descriptor; such a recalculation is only required if the method used to generate the descriptor itself is changed—a process that can also be performed in the background.



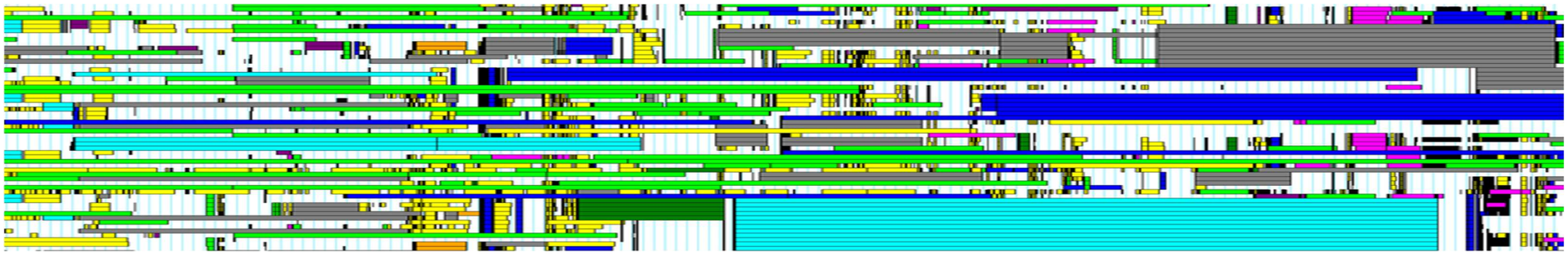
Case 2: Problem

- SPbPU supercomputer center: shared-use, heterogeneous clusters, SLURM scheduler.
- Users submit tasks with CPU, memory, and runtime requirements.
- Main problem: users often provide inaccurate runtime estimates or leave defaults.
- Poor estimates reduce schedule density and increase waiting time.
- Constraints:
  - do not change user workflow;
  - do not add delays to SLURM;
  - guarantee task execution;
  - preserve user-specified limits.



Case 2: An example of supercomputer resource loading

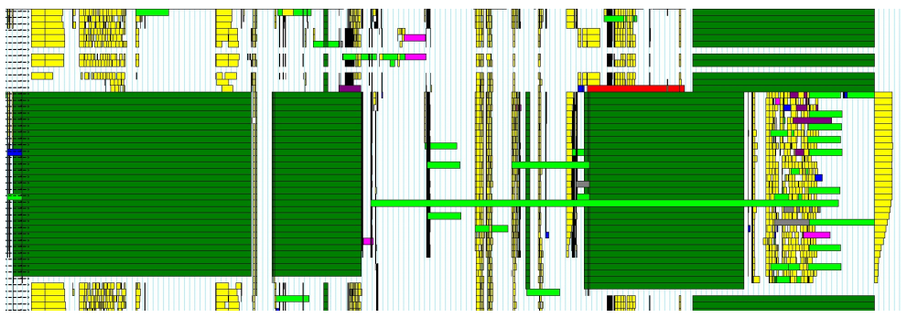
Homogeneous cluster "Tornado"



Heterogeneous cluster "Tornado"



Homogeneous cluster “Cascade”



14 days

- IT – Black
- Astrophysics – Turquoise
- Bioinformatics – Red
- Biophysics – Pink
- Energy – Green
- Geophysics – Yellow

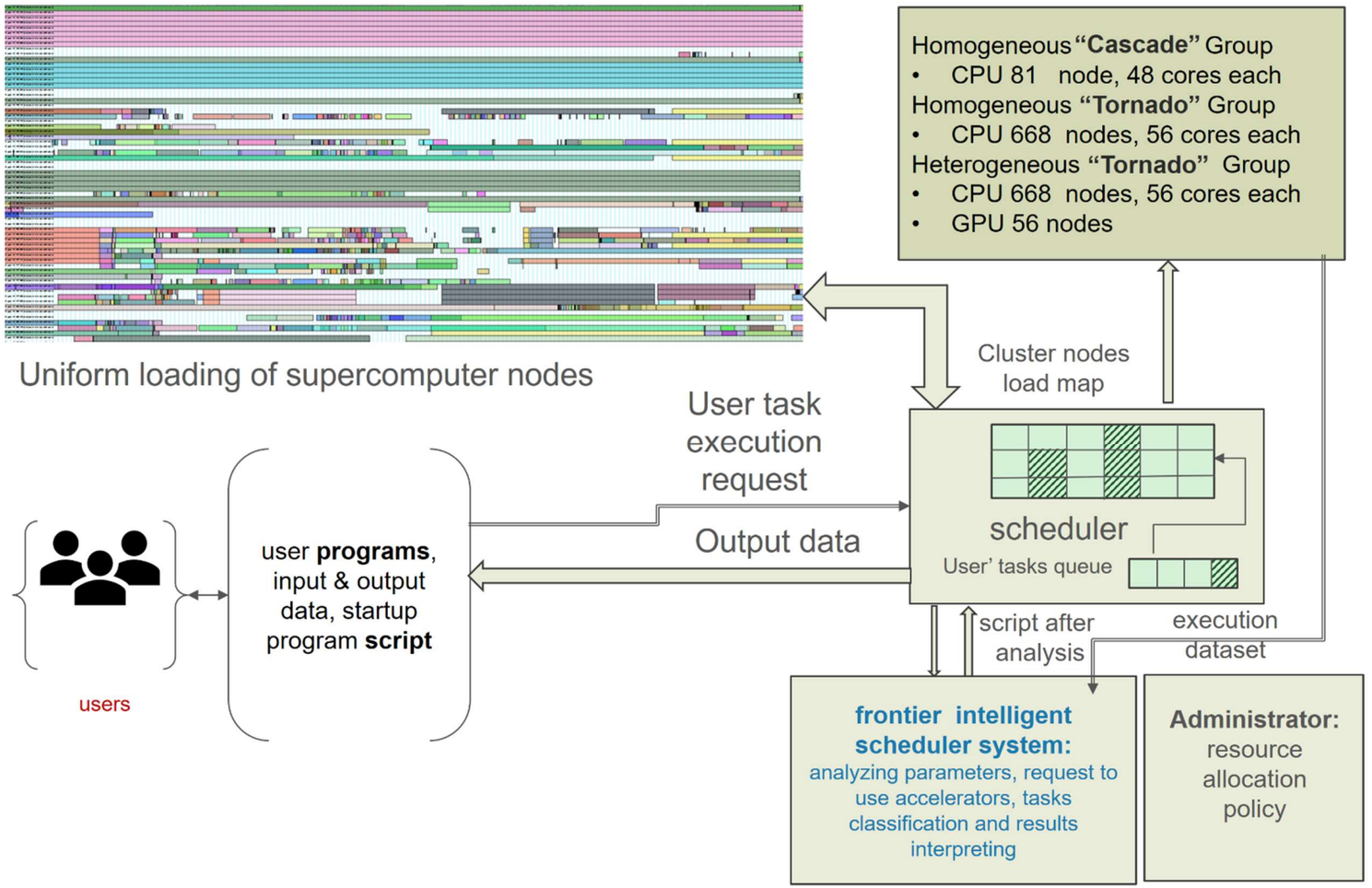
30 days

- Mechanical Engineering – Blue
- Mechanics – Gray
- Physics – Lime
- Radiophysics – Orange
- Undefined Problems - Purple

Case 2: Proposed solution

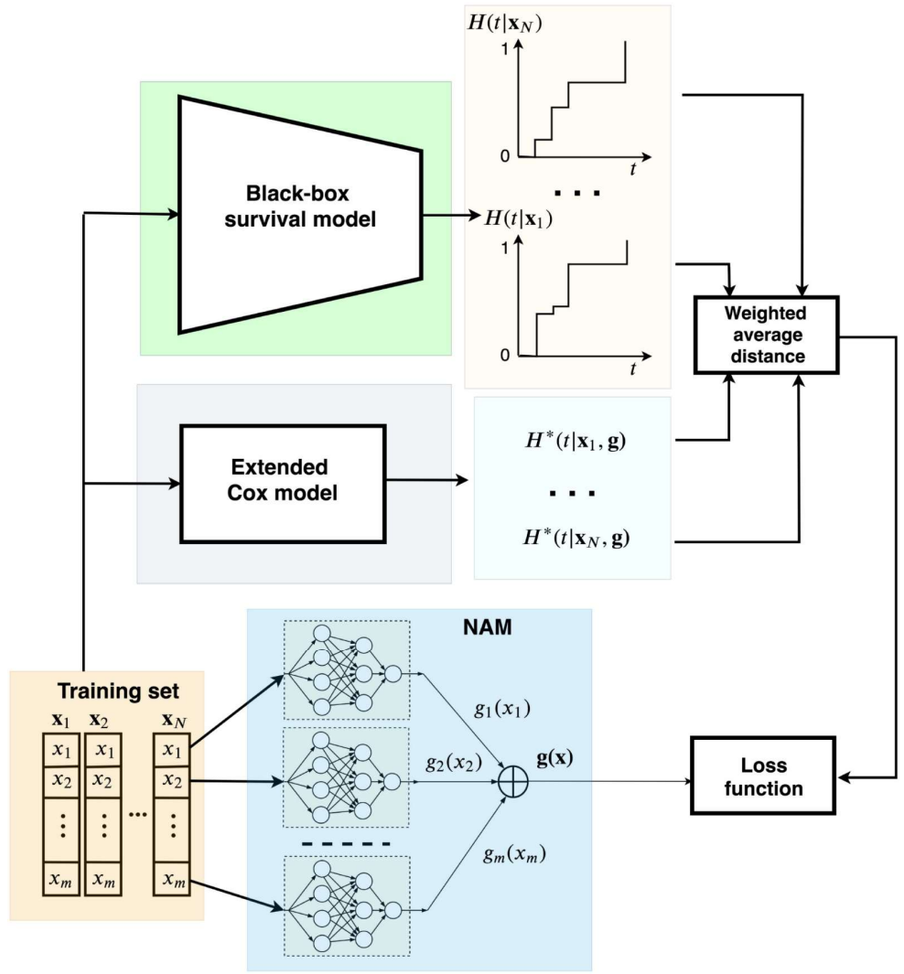
Design principles:

- **User transparency**: module works invisibly.
- **Execution guarantee**: all tasks remain in the queue.
- **Conservative scheduling**: SLURM uses min(predicted, user-specified) duration.
- **Asynchronous operation**: module processes tasks already queued in SLURM's database.
- **Iterative extension**: prediction can be extended up to the user-specified limit.
- **Continuous improvement**: actual execution times are tracked.

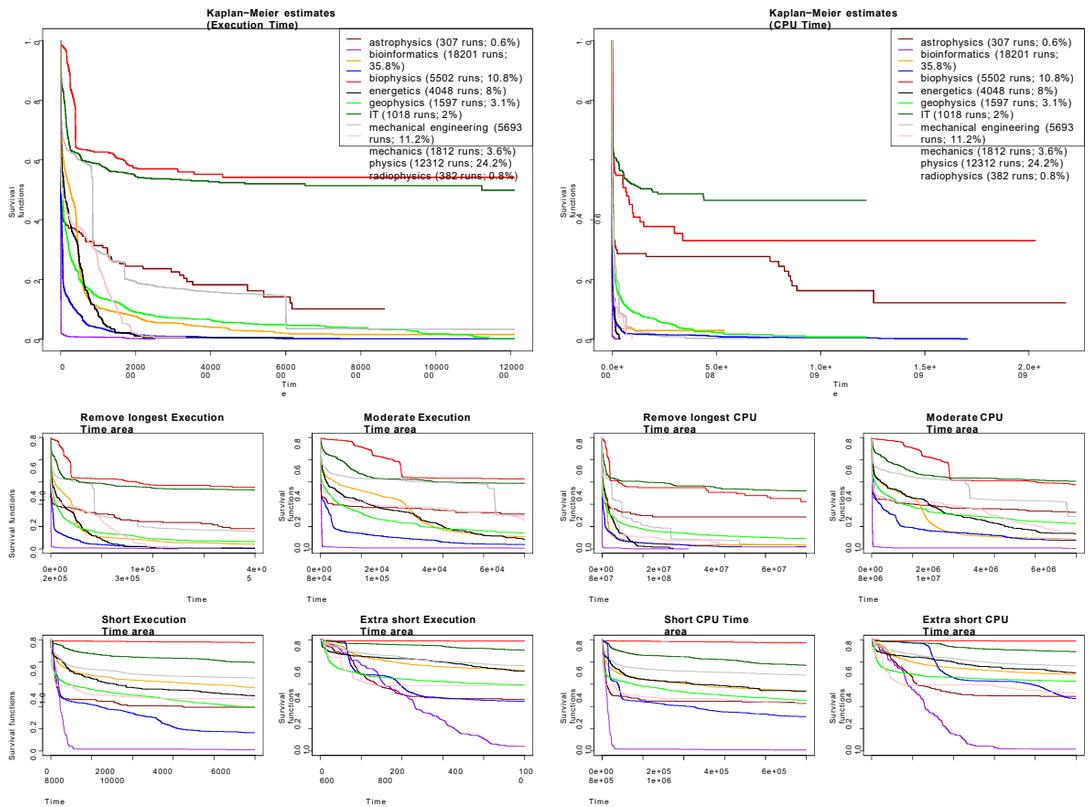


Case 2: SurvNAM model

- SurvNAM is a method designed to help **interpret the decision-making process** of "black-box" models used in survival analysis.
- It addresses the question: **which factors have the greatest impact** on a prediction, and how do these factors interact?
- The core approach involves building a surrogate model. It is **based on a Neural Additive Model** (NAM), which is trained to replicate the predictions of the "black-box" model.
- The surrogate model itself is constructed as an extension of the **Cox proportional hazards model**; instead of assuming a simple linear relationship between covariates (the characteristics of an entity), it **employs a Generalized Additive Model** (GAM). This allows for a more accurate approximation of complex, non-linear dependencies.



Case 2: Experimental results



SurvNAM was designed for censored data, which fits task runtime prediction.

Ensemble of lightweight models trained on randomized data subsets.

Benefits:

- lower training resource demand;
- faster training than a monolithic model;
- asynchronous inference.

Prediction uses task data and user history.

These functions define the distributions of task completion times across different user groups.

If the failure function remains significantly high at the final time point, it indicates that a large proportion of tasks have either not been completed or have ended unsuccessfully (i.e., they are censored).

Case 2: Experimental results

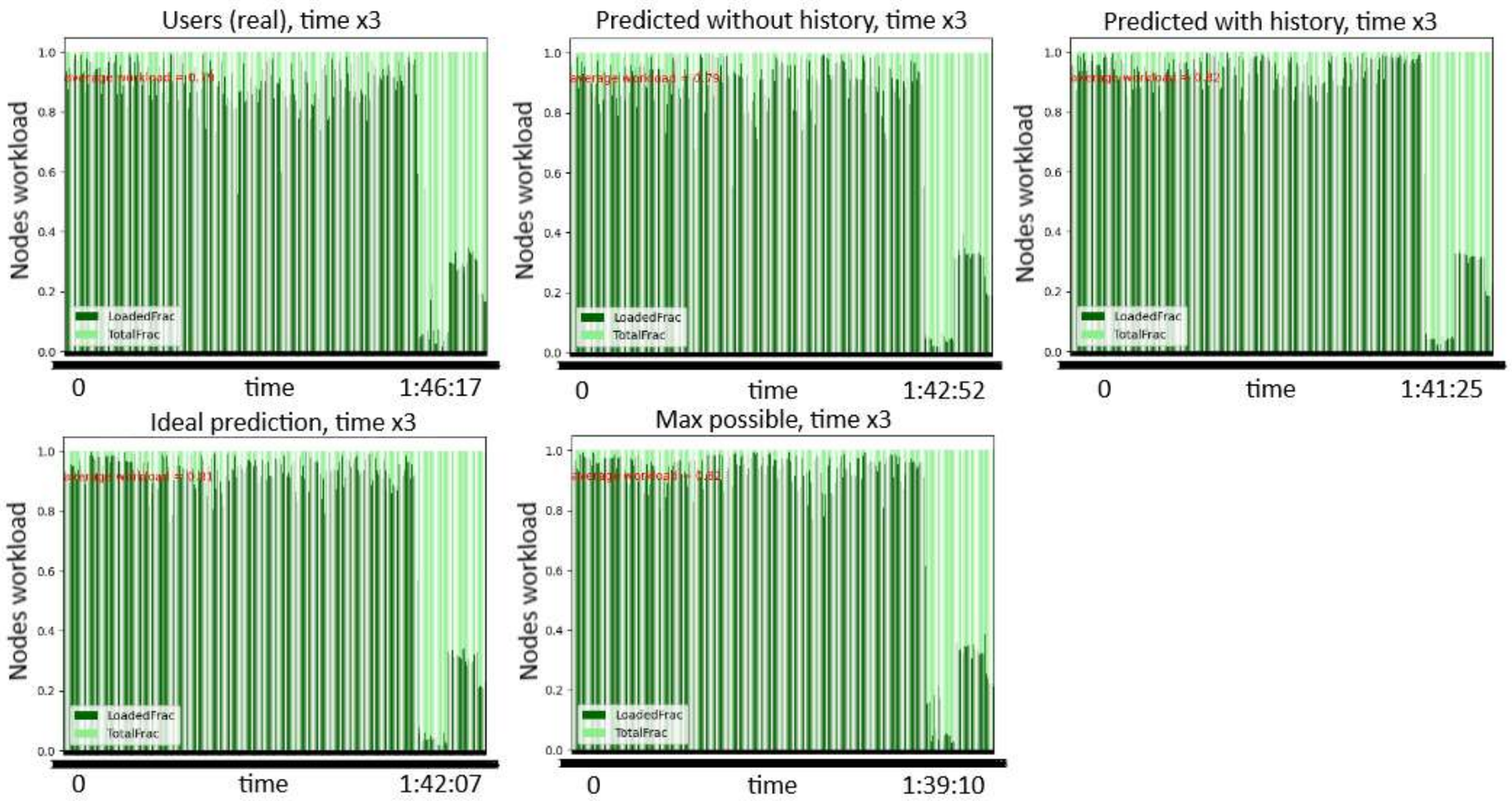
Simulation is based on 2 years of real supercomputer operation data;

five runtime estimate types:

- user estimate;
- model prediction without user history;
- model prediction with user history;
- ideal match;
- maximum possible time.

Results:

- model with user history improves schedule density;
- average task waiting time reduced by 5–10%;
- reduced equipment downtime due to early task completion;
- better planning of computational resource load.



Conclusion

- This paper considers both the main existing methods for implementing machine learning models under constraints, and the implementation of such methods in two specific practical problems solved by the SPbPU team
- In the first case
    - the requirements for invariance to the position, orientation and size of the model are solved by developing a **special shape descriptor;**
    - the smallness of the training sample and the problem of achieving performance are solved by applying a **Siamese neural network** to the developed descriptor;
    - **expert assessments** allow additional training of the proposed model and improving the quality of the assessment.
- In the second case
    - an ensemble machine learning model was used, working with data from both the tasks themselves and the users who run them;
    - the use of the **asynchronous mode** of predictive analytics of task execution time allowed to avoid additional delays in the work of the SLURM dispatcher;
    - proposed approach **increase the schedule density**, reducing equipment downtime due to early completion of tasks and reducing the average waiting time for tasks.

POLYTECH

Thanks for your attention!



Vladimir Muliukha, director of Higher School of AI Technologies, SPbSPU

Vladimir.Muliukha@spbstu.ru