ML Service with PanDA and iDDS

Tadashi Maeno (BNL NPPS)
on behalf of PanDA, iDDS and HPO teams

NPPS/Omega/EDG joint meeting
on AI/ML in BNL Physics
16 April 2021

Contents

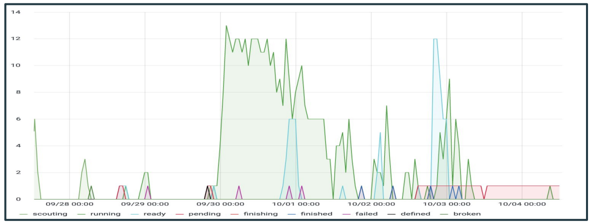
Introduction

- The goal of the ML service project is to provide a service to users for ML-related activities with PanDA and iDDS
  - Scalability and resource integration through PanDA ecosystem
  - Leveraging new capabilities brought by iDDS
    - Decoupling of data delivery and execution
    - Description of workflow with directed acyclic graph (DAG)
    - Orchestration of workflow management system and data management system
  - Modern user auth and interface
- Functions
  - Hyperparameter optimization
  - Parallel training of multiple ML models
  - Elastic distributed training
  - Feedback loops to refine new iterations based on the results of old iterations
  - Task graph for advanced workflows
  - Visualization

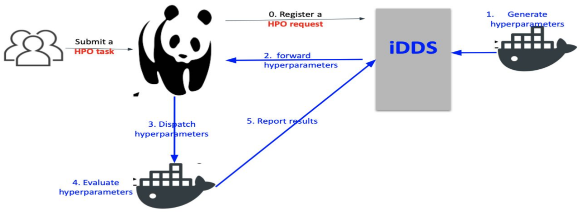
Intelligent Data Delivery Service (iDDS)

(Wen Guan)

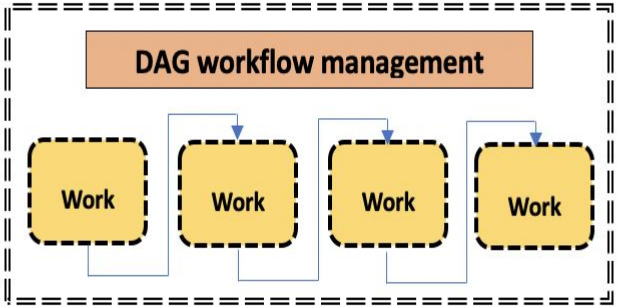
- ➢ Joint ATLAS and IRIS-HEP project launched in 2019
- ➢ Designed to intelligently deliver needed data and workload in a fine-grained way
- ➢ Usecases
  - – Data Carousel
    - Jobs start when its own input is ready, no wait for the full dataset to be transferred
    - In production in ATLAS since May 2020
    - Solved the issues with the delayed start of data processing on tape
  - – HPO (Hyper Parameter Optimization)
    - To provide a fully-automated platform for hyperparameter optimization on top of geographically distributed GPU resources on the grid, HPC, and clouds
    - Advertised to ATLAS ML users, not specific to ATLAS
  - – DAG-based workflow management
    - High-level workflows specified by DAGs driving workload scheduling where successive jobs start off once all dependent jobs are done
      - ➢ Cascade of chains for multi-step processing with thousands of jobs per step
      - ➢ Release jobs incrementally for different steps to avoid long waiting time
    - Using DOMA PanDA and iDDS instances for Rubin Observatory (LSST) exercise



iDDS tasks accounting (by status)



iDDS HPO



iDDS DAG

Hyperparameter Optimization Service

Hyperparameter Optimization (HPO)

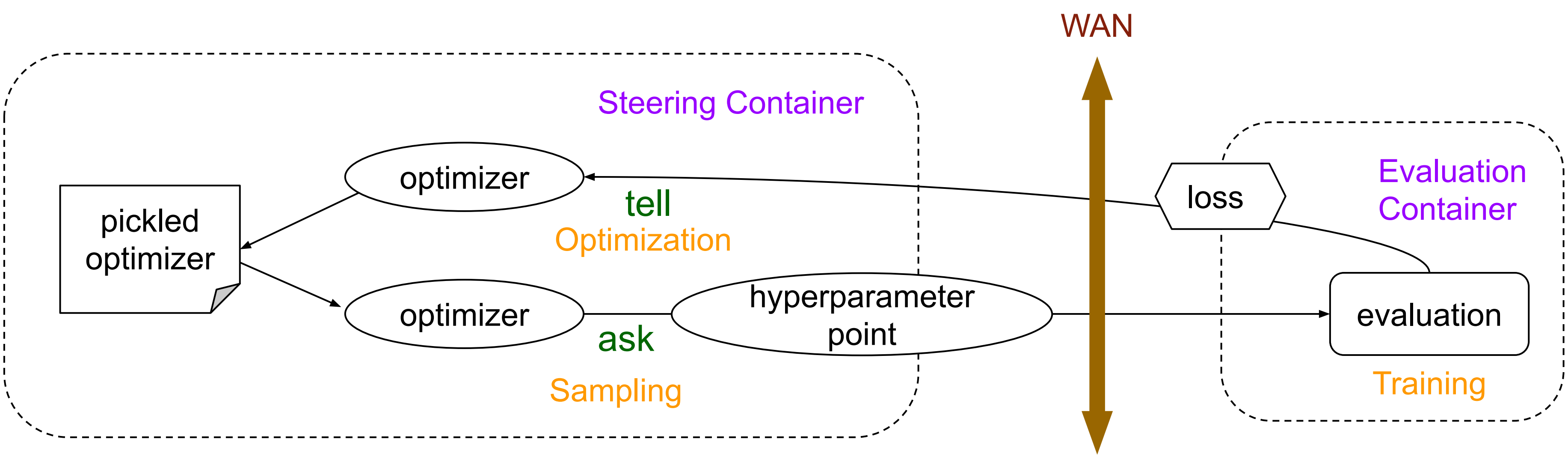
- The problem of choosing a set of optimal hyperparameters for a ML model
  - A hyperparameter = A parameter whose value is used to control the learning process
  - Parameter scan in a search space → a whole training session for each parameter point → computationally intensive
- Usage of GPU resources is crucial
  - Well optimal for linear algebra operations that play a key-role in ML training
  - PanDA's capability to easily integrate heterogeneous resources
- HPO in existing ML packages
  - Single-function-call pattern
    - A kind of a blackbox that manages computing resources behind the scene
    - Not suitable to work with PanDA since PanDA has its own resource management mechanism
  - Ask-and-tell pattern
    - Asynchronous execution of sampling, training, and optimization steps
    - Purely point searching, no resource management

"The ask-and-tell pattern"

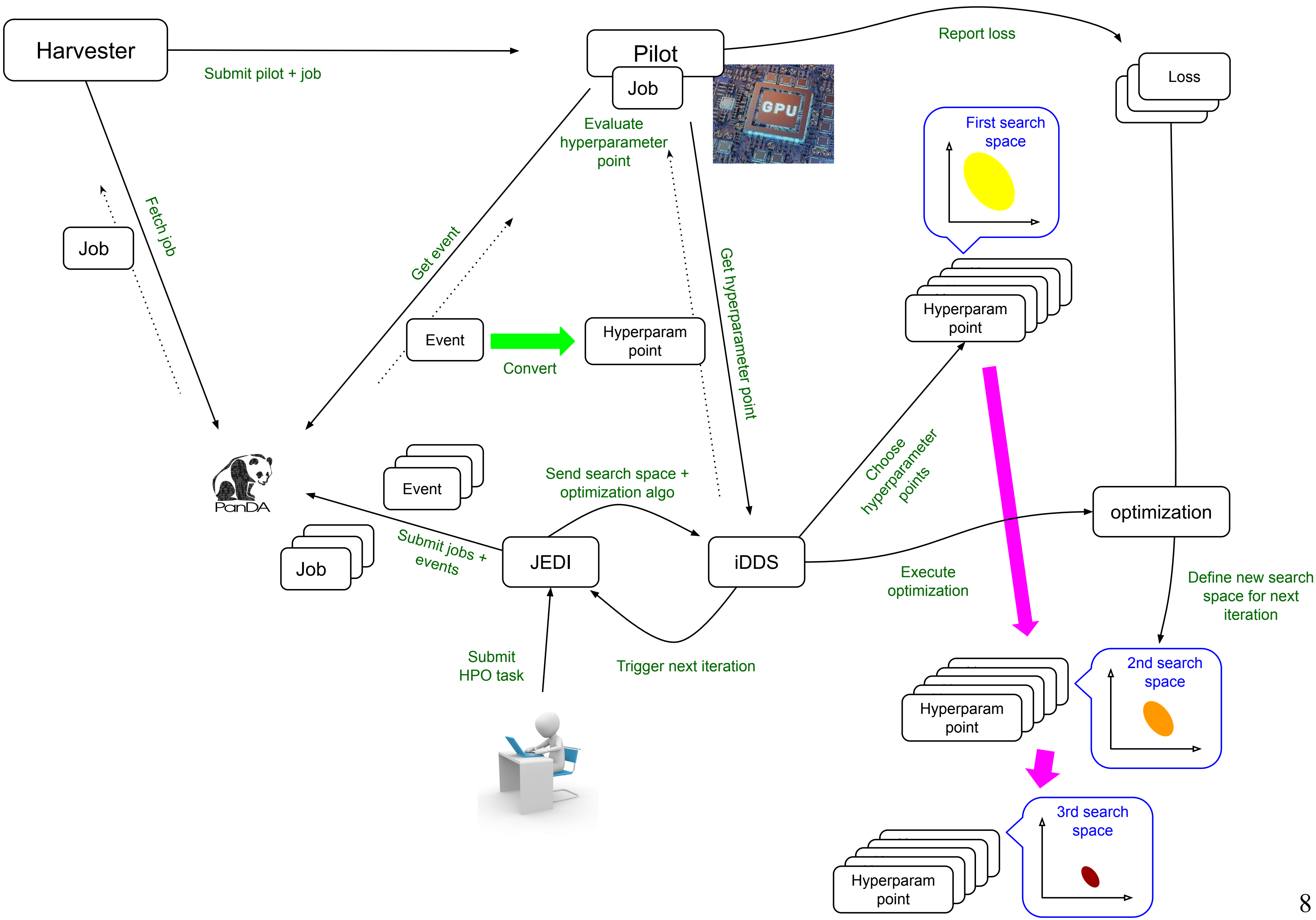
```
while ~ opt.stop
  x = ask(opt)
  y = f(x)
  opt = tell(opt, x, y)
end
```

Ingredients of HPO Workflow

- Two types of containers
  - Steering container - optimisation on central iDDS server
    - Generate next HP points with customised method
    - A wide range of optimization algorithms are supported
  - Evaluation container - training at remote grid (GPU) sites
    - Submodule payload contains a ML model definition and user-specific training
- Checkpointing
  - Periodically upload checkpoints to Grid
  - Download the checkpoint when the same job is retrying
  - Resume training if checkpoint is found

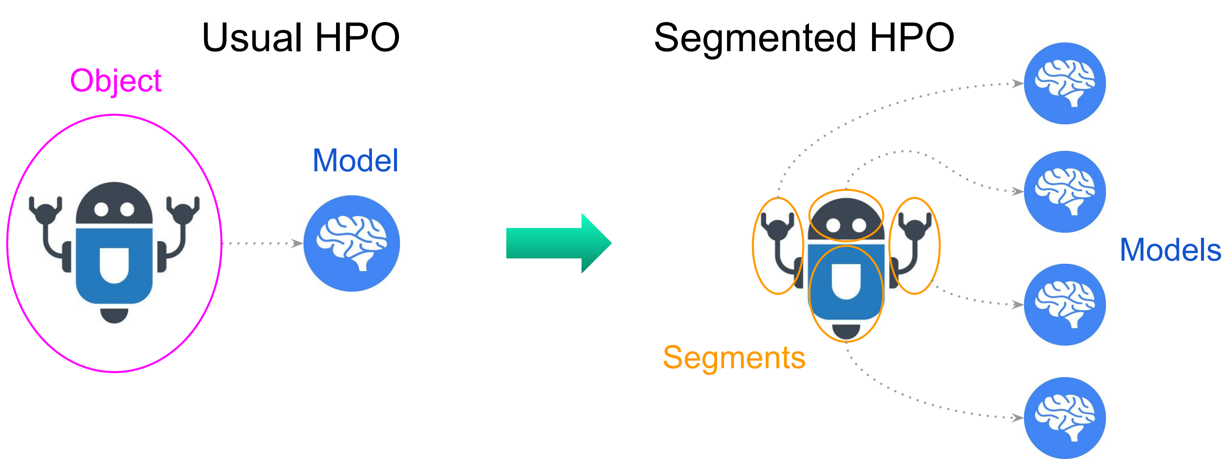


HPO Service with PanDA and iDDS

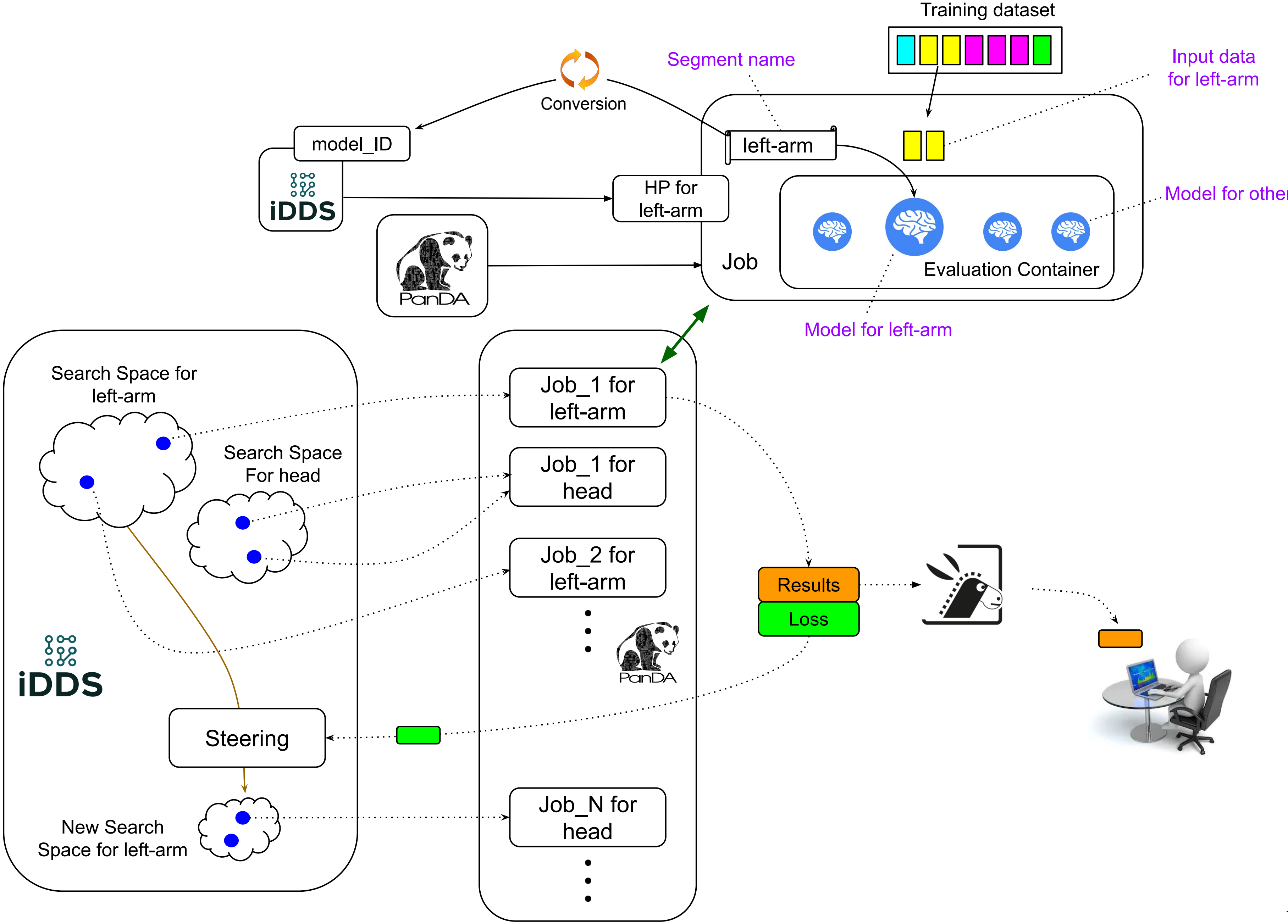


Segmented HPO

- ➢ Some ML payloads can be logically broken down
  - E.g. break-down of a single physics search session into multiple search sessions targeting different physics regions/entities
- ➢ A real ATLAS example: FastCaloGAN, a calorimeter image generation model
  - 300 GANs = 300 models = 100 η slices x 3 PIDs
  - 100 GPU-days to train 300 GANs
  - 300 individual tasks in the usual workflow → Bookkeeping nightmare
- ➢ Segmented HPO
  - A single HPO task to optimize all ML models in one-go
  - Concurrent training of multiple models, and a smaller training dataset for each model
    - Fast turnaround
    - Execution of workloads on more distributed resources



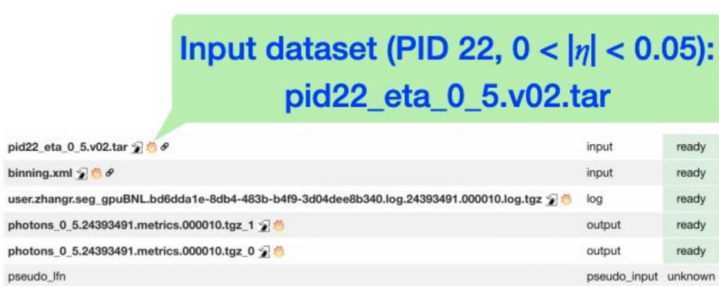
Segmented HPO Workflow

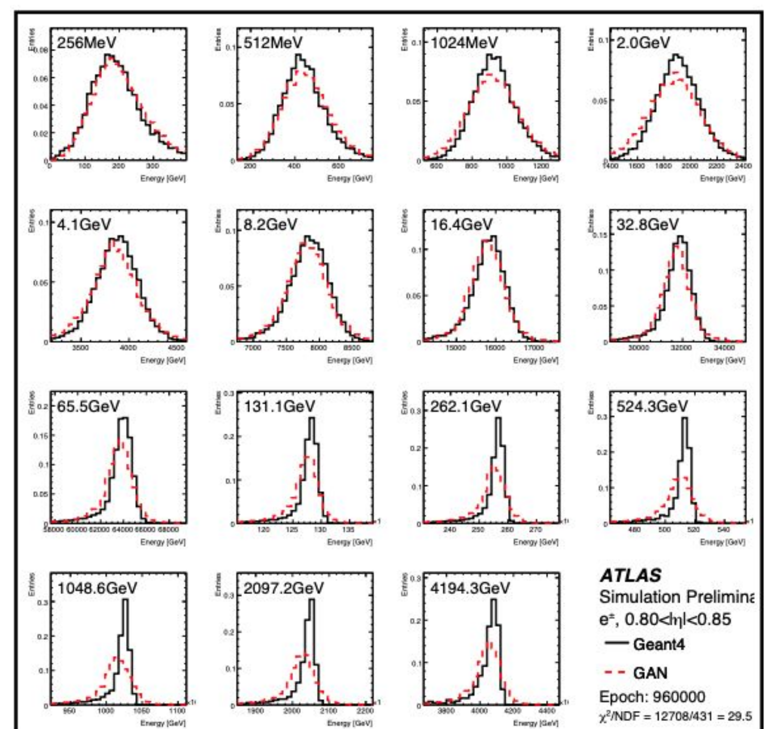


(Rui Zhang)

Test Results with FastCaloGAN

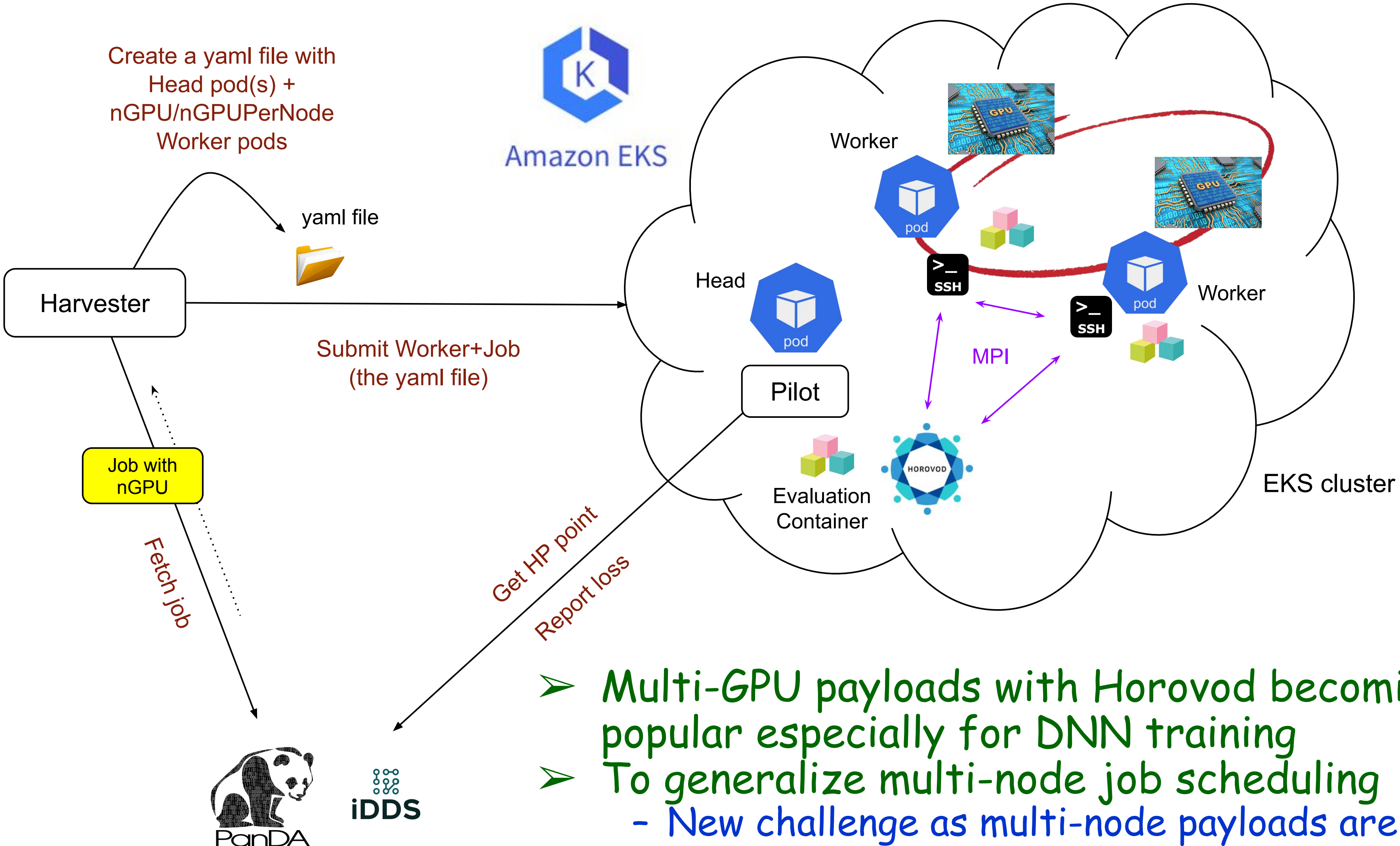
- Tested with 15 GANs (15 segments)
  - 3 particle types × 5 η slices
  - Grid search as still needs offline analysis of training results
- Staged only relevant data for each GAN rather than sending the whole data in the training dataset
  - Minimized data motion
- Reasonable results shown in the plots on the right from a 10K-epoch job running on the BNL GPU site
- Foreseen full automation with more advanced search algorithm once the procedure of the offline analysis is well established





Ongoing Development Activities

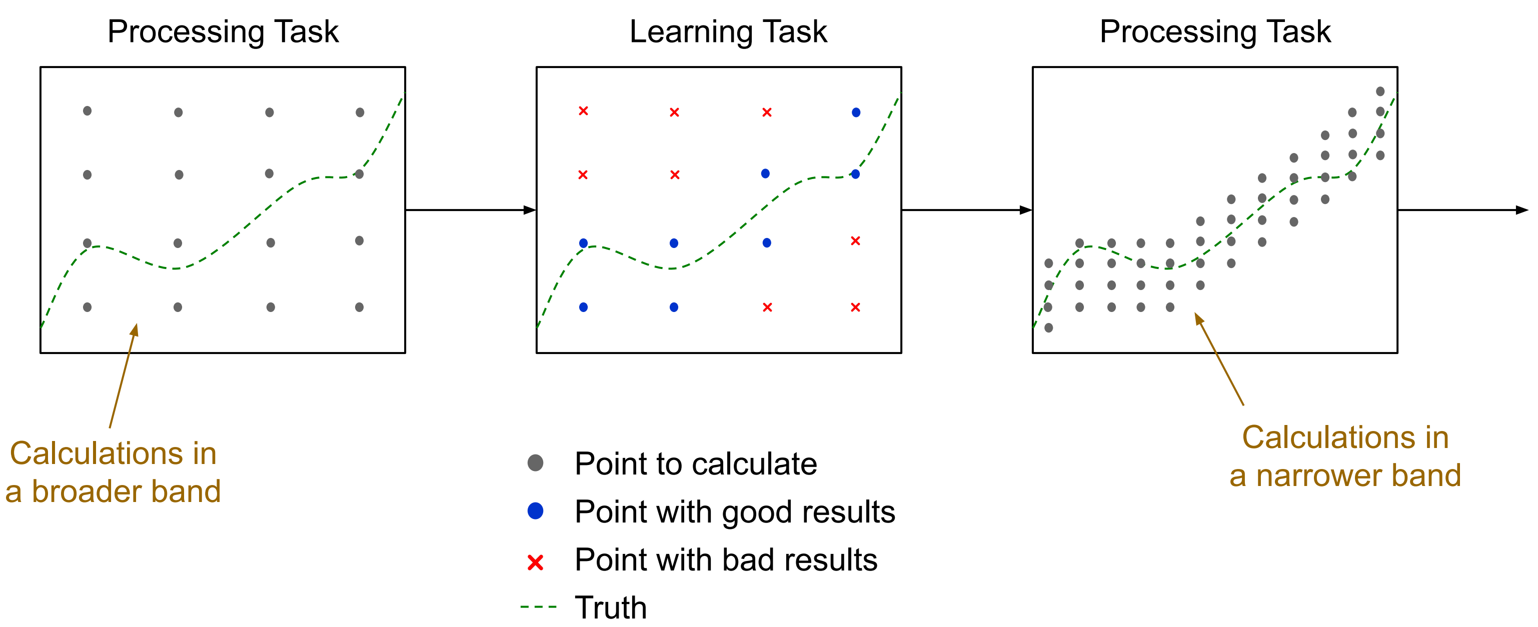
Distributed Training with Horovod on Amazon



- ➢ Multi-GPU payloads with Horovod becoming popular especially for DNN training
- ➢ To generalize multi-node job scheduling
  - New challenge as multi-node payloads are used only on some HPCs
- ➢ CPU-only Head on an on-demand instance + GPU Workers on spot instances
  - Workers can come and go thanks to resilience in Horovod

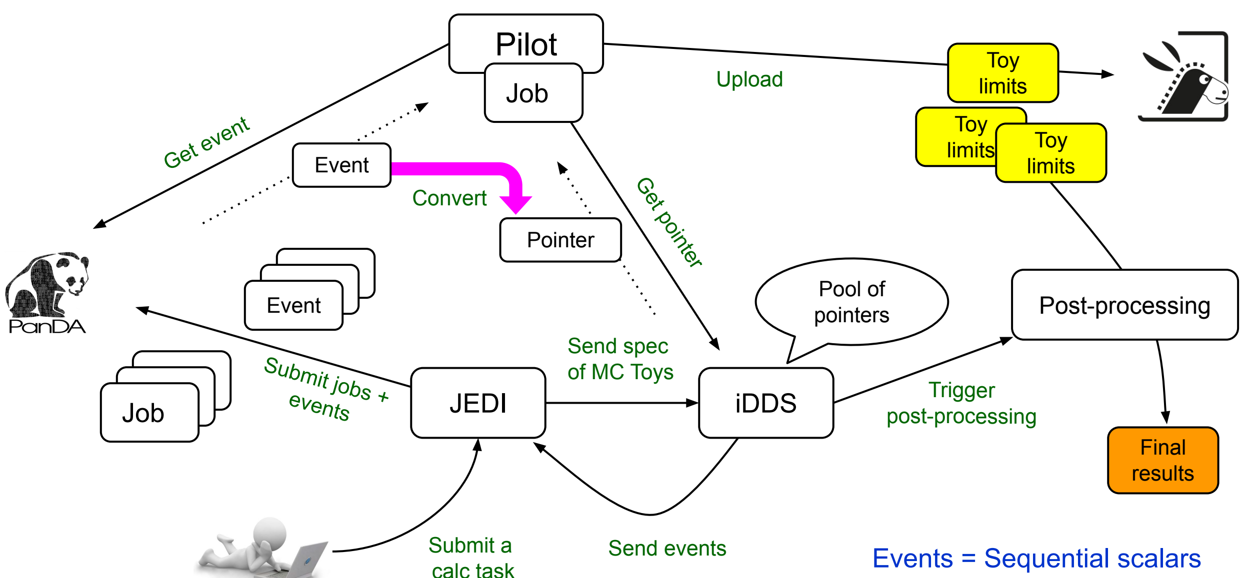
Active Learning

- To define the subsequent processing task based on the decision making in the learning task which analyzes the results of the previous processing task
  - Task chain + decision making between
  - A simple DAG usecase
- Two types of task templates to generate concrete tasks, and condition branches to control the workflow
- Being integrated in the system



RootStats-based Limit Calculation with MC Toys

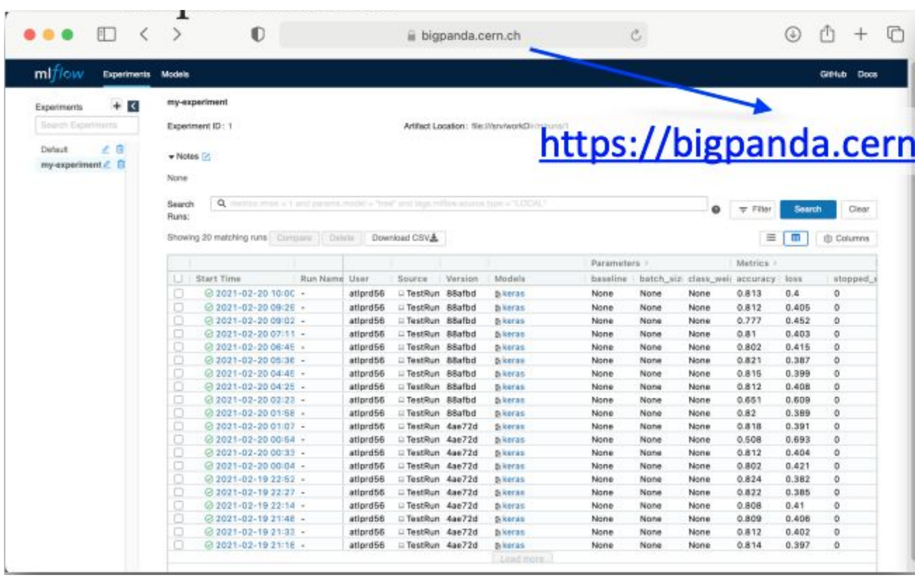
- 10000 MC toys would take approximately 55 hours to run, according to Xola's slides → Offloading random number generation to GPU
- Materials in Christian's repo: https://gitlab.cern.ch/chweber/StandardHypoTestInv
- Mapping to the system as a chain from toy limit calculations to post-processing without iteration
  - iDDS has a pool of pointers to MC toys
  - Each job takes a pointer to calculate the relevant toy limits and takes another pointer if the walltime is still available
  - iDDS triggers the post-processing that combines toy limits to the final results



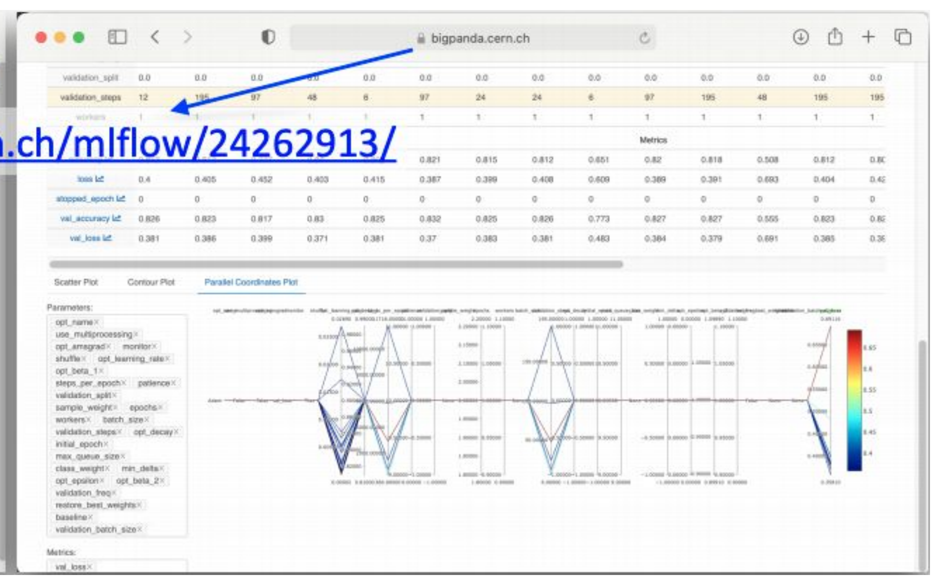
Events = Sequential scalars
Pointers = Any dictionaries

Visualization Support for HPO

- A visualisation tool MLflow is turned on in Evaluation Container
  - Useful for offline visualisation and analysis
- An α-version of the tool also integrated into PanDA Monitoring system
  - Fetch output from training jobs and centrally spin-up an MLFlow container to display results
  - Extendable to other visualisation tools (Neptune, WandB, Tensorboard, etc.)

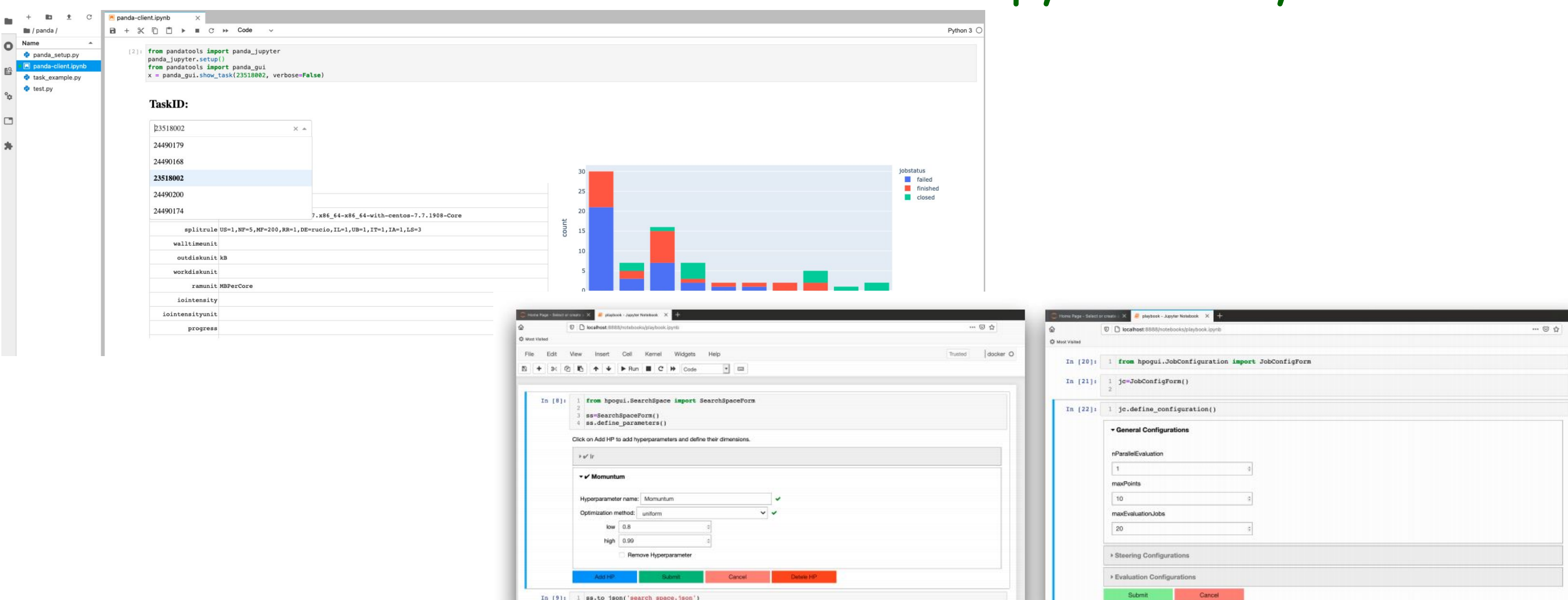


https://bigpanda.cern.ch/mlflow/24262913/



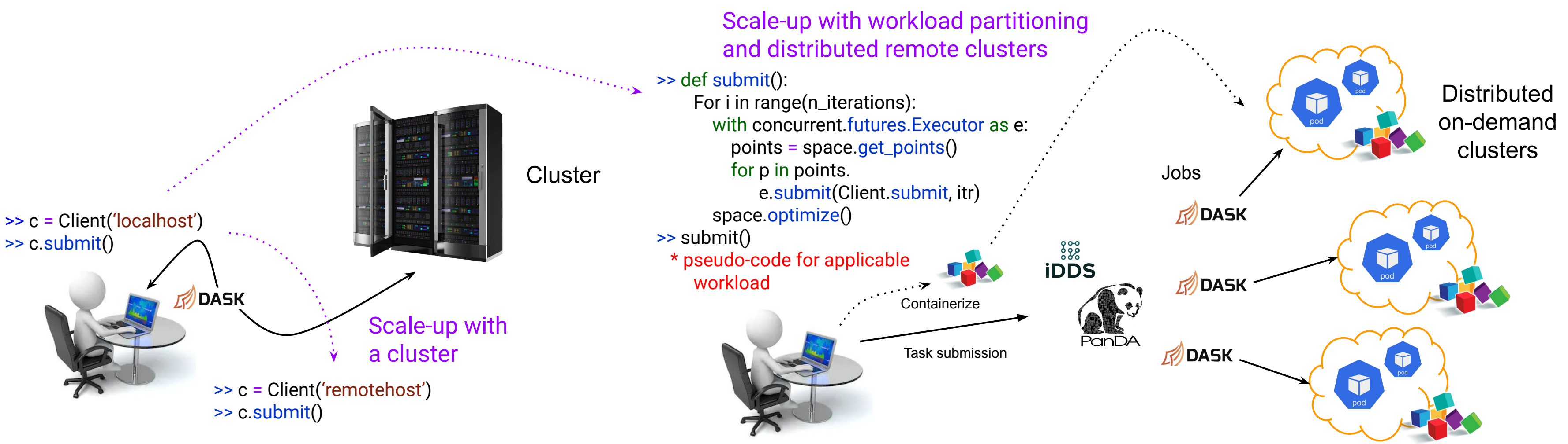
User Interface in Jupyter Lab

- ➢ Jupyter Notebook has become a popular user interface for research science, data science, data analytics, and ML
- ➢ Access to PanDA/iDDS from Jupyter
  - Seamless integration with user's analysis environment
  - Remote resources through PanDA/iDDS
- ➢ Easy to provide sophisticated look-and-feel especially for advanced workflows
  - E.g., a visual interface to define and control task networks for DAG workflows
- ➢ Thousands of extensions/tools in Jupyter ecosystem



Running Dask through PanDA/iDDS

- ➢ Dask is a Python library for parallel computing
  - Very easy to parallelize your analysis written with other Python projects such as NumPy, pandas, scikit-learn, etc
  - Built-in capability for seamless scale-up with clusters
- ➢ To offer further scale-up with workload partitioning and distributed remote clusters
  - Depending on characteristics and requirements of each workload
  - E.g. distributed training with Dask instead of Horovod
- ➢ Another type of multi-node payloads from the system point of view
  - Trying to reuse the mechanism that has been originally developed for Horovod payloads
- ➢ GCP as a part of Google R&D Y2, EKS, other k8s-based resources



Conclusions

- PanDA/iDDS-based HPO service is up and running
  - Experiment agnostic implementation
  - Support of both usual and segmented HPO workflows
  - Available for ATLAS users on ATLAS instances and for other experiment users on DOMA instances
- Many ongoing development activities to add ML-related functions to the service
  - HPO service → ML service
- Usecase-driven project
  - Inputs/feedbacks from (BNL) physics communities are highly appreciated
    - E.g., came up with the idea of RootStats-based limit calculation in a meeting with Christian
  - Happy to support more usecases

HPO Documentation:

https://panda-wms.readthedocs.io/en/latest/client/phpo.html