Hyperparameter optimisation for Machine Learning using iDDS

Fernando Barreiro, Doug Benjamin, Alkaid Cheng, Alessandra Forti, Lukas Heinrich, Alexei Klimentov, Fahui Lin, Tadashi Maeno, Paul Nilsson, Pavlo Svirin, Guan Wen, Torre Wenaus, Rui Zhang

University of Wisconsin-Madison

IRIS-HEP Blueprint Workshop: Future Analysis Systems and Facilities

Oct 27, 2020

Guan Wen (IRIS-HEP support) is supported by the National Science Foundation under Grant No. 1836650.

Contents

- Introduction
- intelligent Data Delivery Service (iDDS)
- Hyperparameter optimisation using iDDS on Grid
- Development for HPCs

Introduction: HPO service in ATLAS

- (Needless to say the importance of hyperparameter optimisation for ML training.)
- The goal is to provide an HPO service to ATLAS users for Machine Learning
  - Minimal user code adaption
  - Support for advanced search algorithms in addition to the traditional grid or random search algorithms
  - Visualisation of results
  - To integrate geographically distributed GPU resources to provide a single resource pool to end-users
- Single-function-call pattern for HPO
  - Computing resources are managed behind the scene
  - Not suitable since ATLAS has its own resource management
- Ask-and-tell pattern for HPO
  - Decoupled optimisation+sampling from training in space-time
  - Purely point searching, no resource management
  - We go in this way

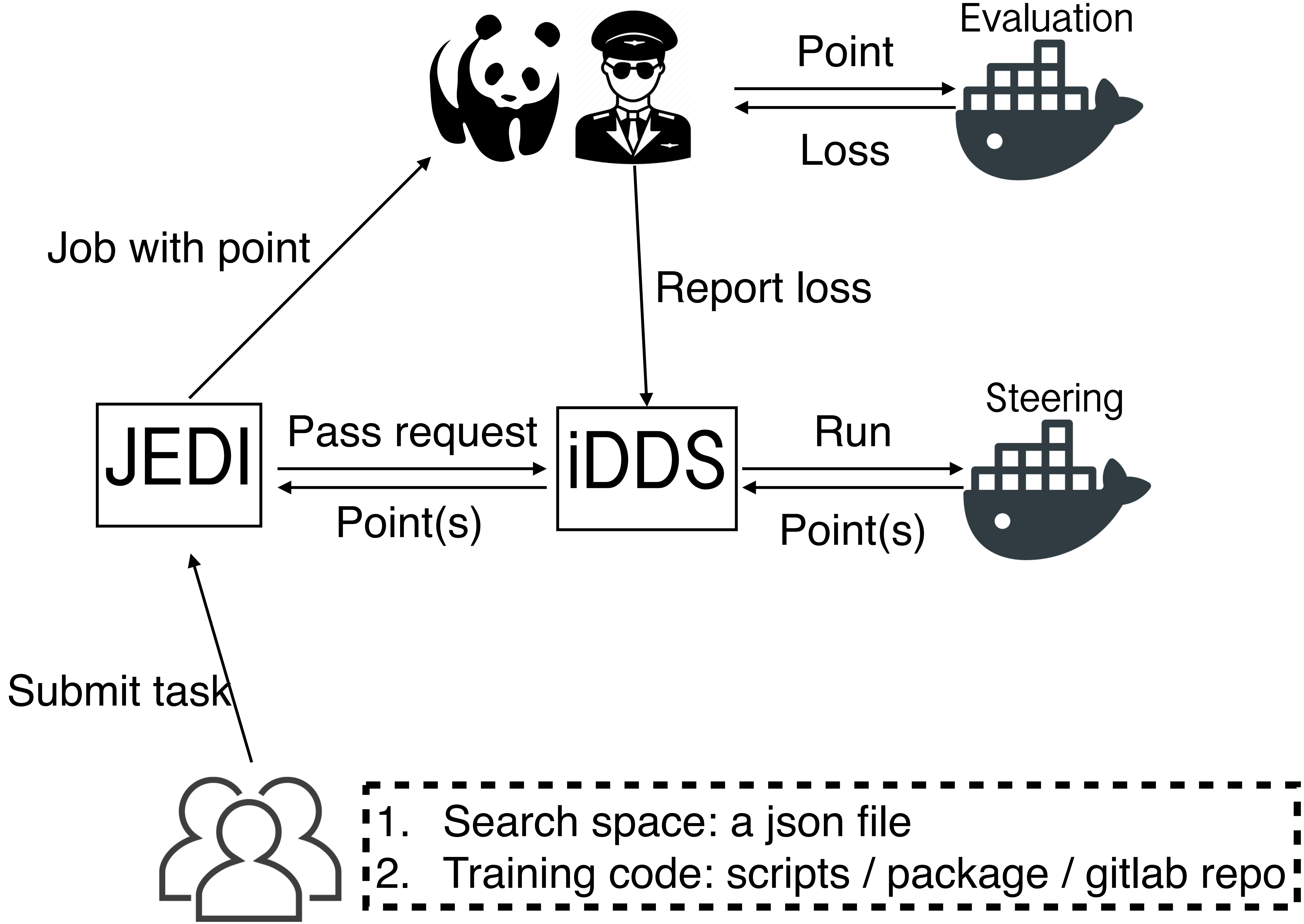
"The ask-and-tell pattern"

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

The intelligent Data Delivery Service (iDDS)

- iDDS is designed to intelligently transform and deliver needed data to workflows in a fine-grained way.
  - My takeaway: jobs of successive tasks start as soon as possible, no need waiting for precedent tasks to finish, optionally making decisions in between
  - Many applications share this paradigm (documentation for currently supported use cases), e.g.:
    - Data Carousel: job starts when its input is ready, no waiting for the full dataset to be transferred
    - A chain of tasks (DOMA): successive jobs start when enough inputs are produced by the precedent tasks
    - A chain of tasks (Active Learning): successive jobs are created and submitted by iDDS based on results of precedent tasks => extendable to a generic function-as-a-service type of workflow
- HPO is a series of tasks with decision-making in between - another use case

The HPO workflow



Containerisation of the workflow

✤ Two containers to fulfil the loop:

Steering

- `SteeringContainer` - optimisation at iDDS server
  - Generate next HP points with customised method
  - A wide range of HPO methods are supported

Evaluation

- `EvaluationContainer` - ML training at Grid (GPU) sites
  - Submodule `payload` contains model definition, training scripts (user specific)

HPCs as GPU resources

- Summit as an example
  - 4608 computer nodes
    - 2 Processors x 22 cores / node
    - 6 V100 GPUs / node
  - Wonderful workstation for ML/HPO
- Challenges
  - Short wall time
  - Standard Grid services and workflows unavailable or suboptimal

HPCs as GPU resources

- Solutions
  - Checkpointing supported in the HPO workflow
  - Evaluation containers with power9 or multi-architecture support
  - Harvester on edge to mediate communication between evaluation containers and iDDS/PanDA for network-less compute nodes
  - Leveraging the data transfer service at each HPC centre not officially adopted in Rucio
  - Evolutionally specialised workload:
    1) Multiple single-GPU payloads on a single node
    2) A multi-GPU payload on a single node
    3) A multi-node/GPU payload on a static multi-node cluster
    4) A multi-node/GPU payload on a dynamic multi-node cluster for elastic distributed training

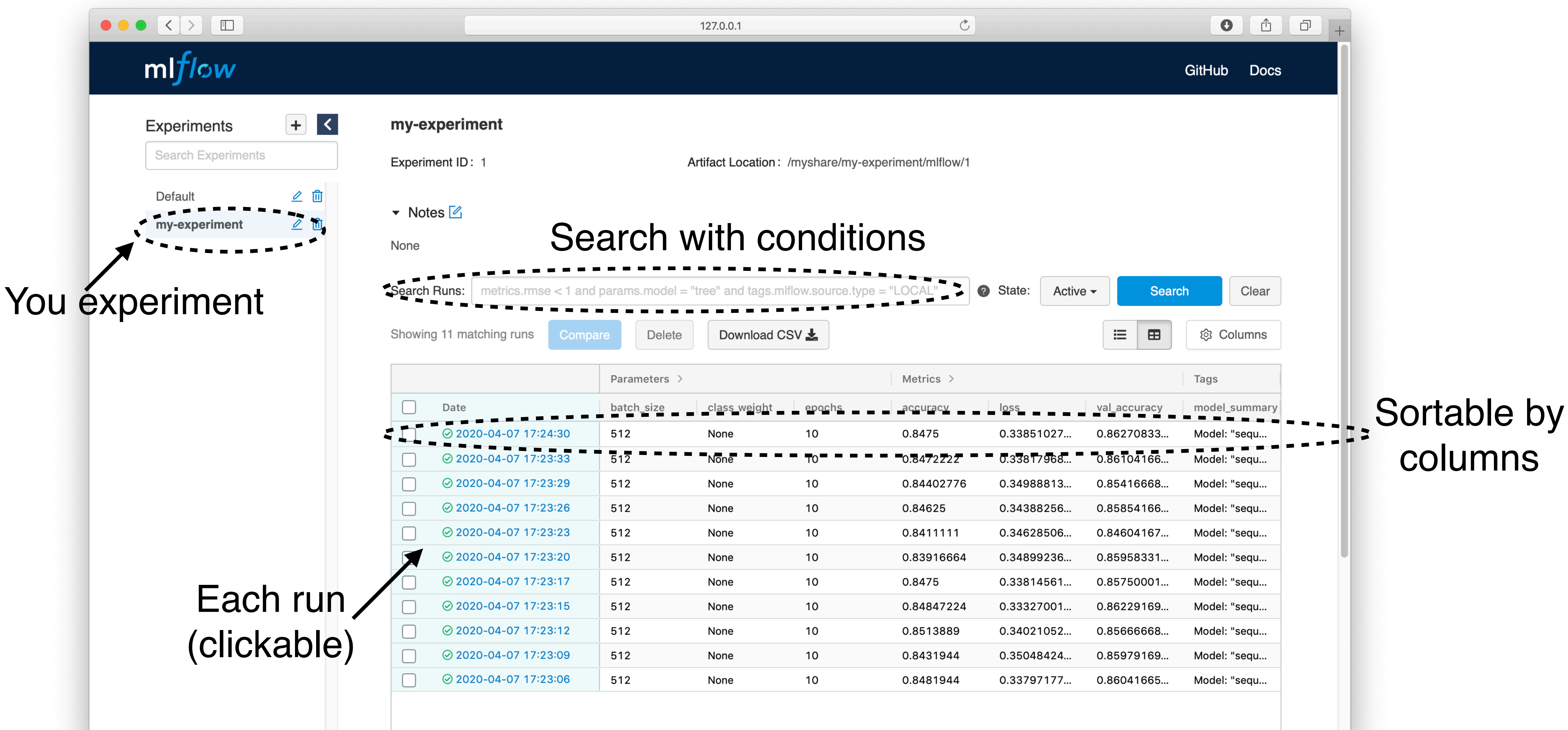
Summary

- Aim was to provide users resources for ML/HPO
  - Running with Grid site in production. Being extended to run with Google and Amazon cloud resources.
  - Running on HPC/Summit is in progress
- A survey is sent out from Physics Coordination on how much GPU resources are/will be needed by ATLAS users

Backup

Visualisation

- By default `MLflow` is turned on in `EvaluationContainer`
  - Offline visualisation on any laptop with MLflow installed is possible
  - More than visualisation - it is a ML lifecycle system
- Working with the PanDA Mon team to get a visualisation directly from Panda



Documentations

- Walk-through the Calo Image-based DNN example
  - SteeringContainer: https://gitlab.cern.ch/zhangruihpc/SteeringContainer
  - EvaluationContainer: https://gitlab.cern.ch/zhangruihpc/EvaluationContainer
- How to submit HPO task
  - https://twiki.cern.ch/twiki/bin/view/PanDA/PandaHPO
- iDDS Readme about the interfaces of ask-and-tell pattern
  - https://idds.readthedocs.io/en/latest/usecases/hyperparemeter_optimization.html