Machine Learning for ATLAS Distributed Data Management Network Metrics

Mario Lassnig (CERN)
Wesley Toler (GMU), Ralf Vamosi (CERN)

On behalf of the ATLAS Collaboration

Introduction

Distributed Data Management (DDM) involves a heterogeneous infrastructure with a highly dynamic state

- ↪ Data management is involved at all layers: software, computing, storage, network
- ↪ Difficult to get reliable and consistent instrumentation in a distributed system

Quasi-static, reactive way of system operation

- ↪ For important actions a human is involved — "signing-off" of decisions and tasks
- ↪ Algorithms and parameters tuned based on human experience

System works, but high potential for improvements

↪ **Data rebalancing**	e.g., disk space doesn't match CPU, tape migration, …
↪ **Hot replication**	e.g., create additional copies of frequently used data, …
↪ **Placement selection**	e.g., data distribution based on resource pledges, …
↪ **Source selection**	e.g., use which replica if multiple ones are available, …
↪ **Robustness**	e.g., automatically reschedule tasks and transfers, …

DDM Network Metrics

Centrally collect and make available DDM metrics to help with those problems

Static link metrics

- ↪ **Source** and **destination** site
- ↪ **Closeness** as defined by ATLAS Distributed Computing Operations, updated monthly

Dynamic link metrics

↪ **Packetloss** as a percentage	[perfSONAR]
↪ **Latency** as median one-way link delay	[perfSONAR]
↪ **Percentile File Throughput** in mbps	[FTS, Dashboard, FAX]
↪ **Link Throughput** in mbps	[perfSONAR]
↪ **Queued files** per activity	[Rucio]
↪ **Done files** per activity in the last *n* hours	[Rucio]

First objective: Heavy Ion placement

minimise	**job waiting time**	t[activated - defined]
subject	**limited number of potential sites**	with himem queues
	existing data across	all sites
	available free space at	potential destination sites
	DDM network metrics	latency, packetloss, throughput, closeness
	all involved queue lengths	prodsys, panda, rucio
learn	**for all heavy ion data subject to given constraints → classify destination sites**	

Place or rebalance heavy ion data as close as possible to potential scheduling targets
Constrained learning function with all input and output metrics available

Time to complete transfer estimator

Close in the geographical sense is misleading, instead train an estimator

↪ **Learn input** DDM network metrics, including categorized variates

↪ **Model input** (*bytes, source, destination, activity*)

↪ **Model output** *file transfer duration*

Data Consolidation, T0 Export, Production Input, etc...

full workflow, including queues, not only time on the network

Method uses decision trees

↪ Effective and efficient tool for classification and regression of large datasets

↪ Finds nonlinear relationships between variates

Cross-validation against overfitting

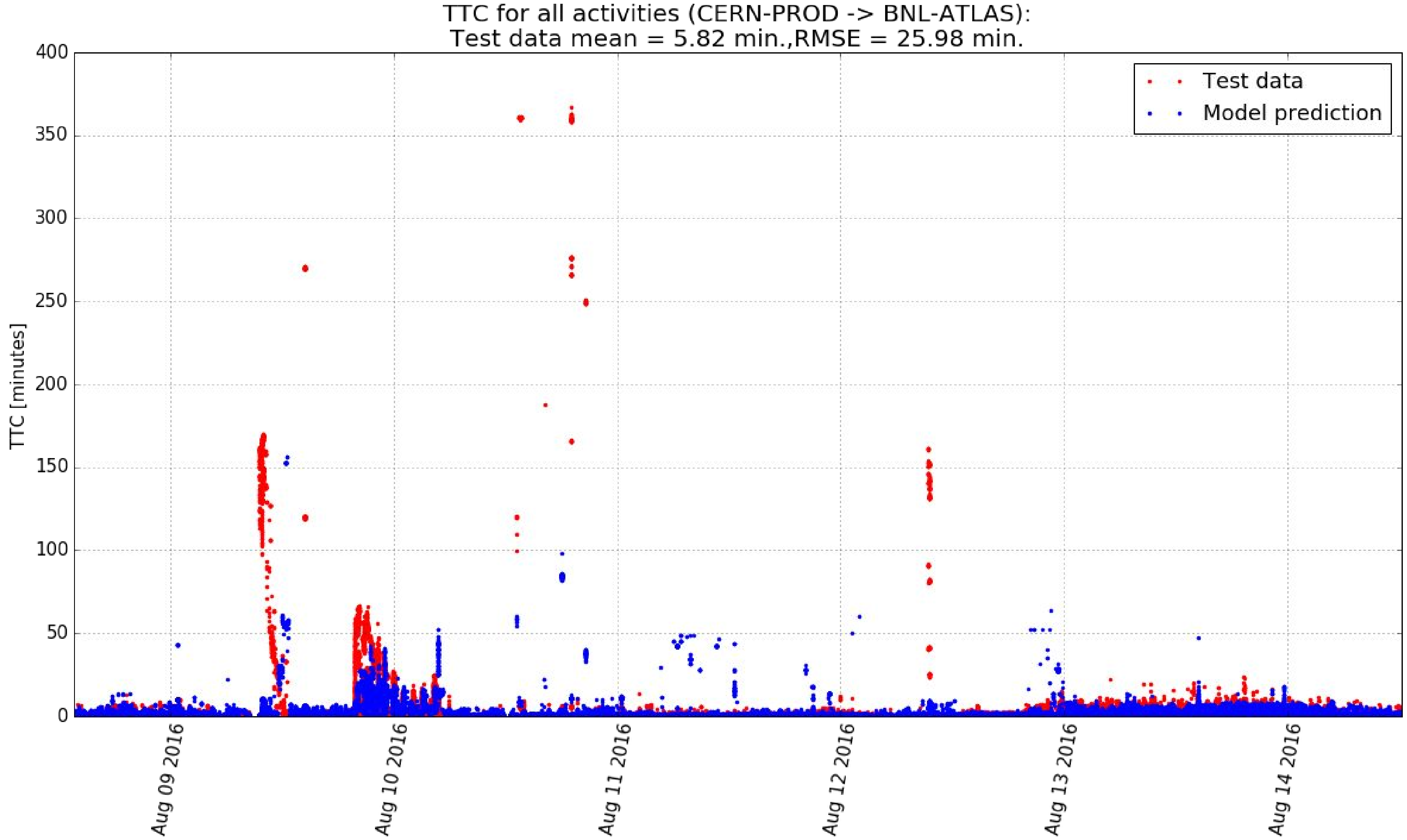
↪ Many random samples generated, each with 80% training, 20% test split

↪ Each sample fitted with separate tree, in our first evaluation 1 month of data used

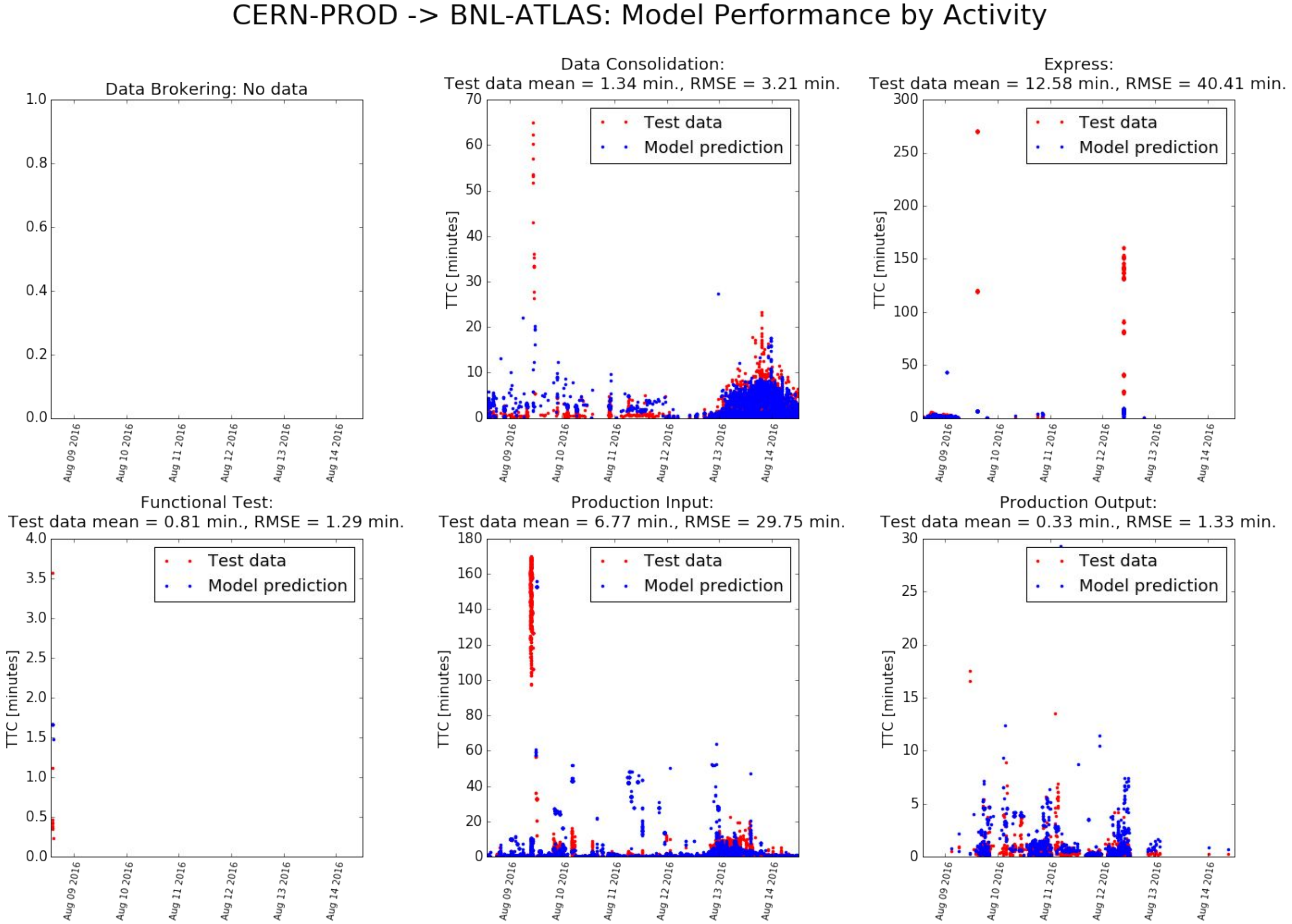
Random forest regressor of many trees

↪ Final prediction which is robust to outliers and noise (Breiman, 2001)

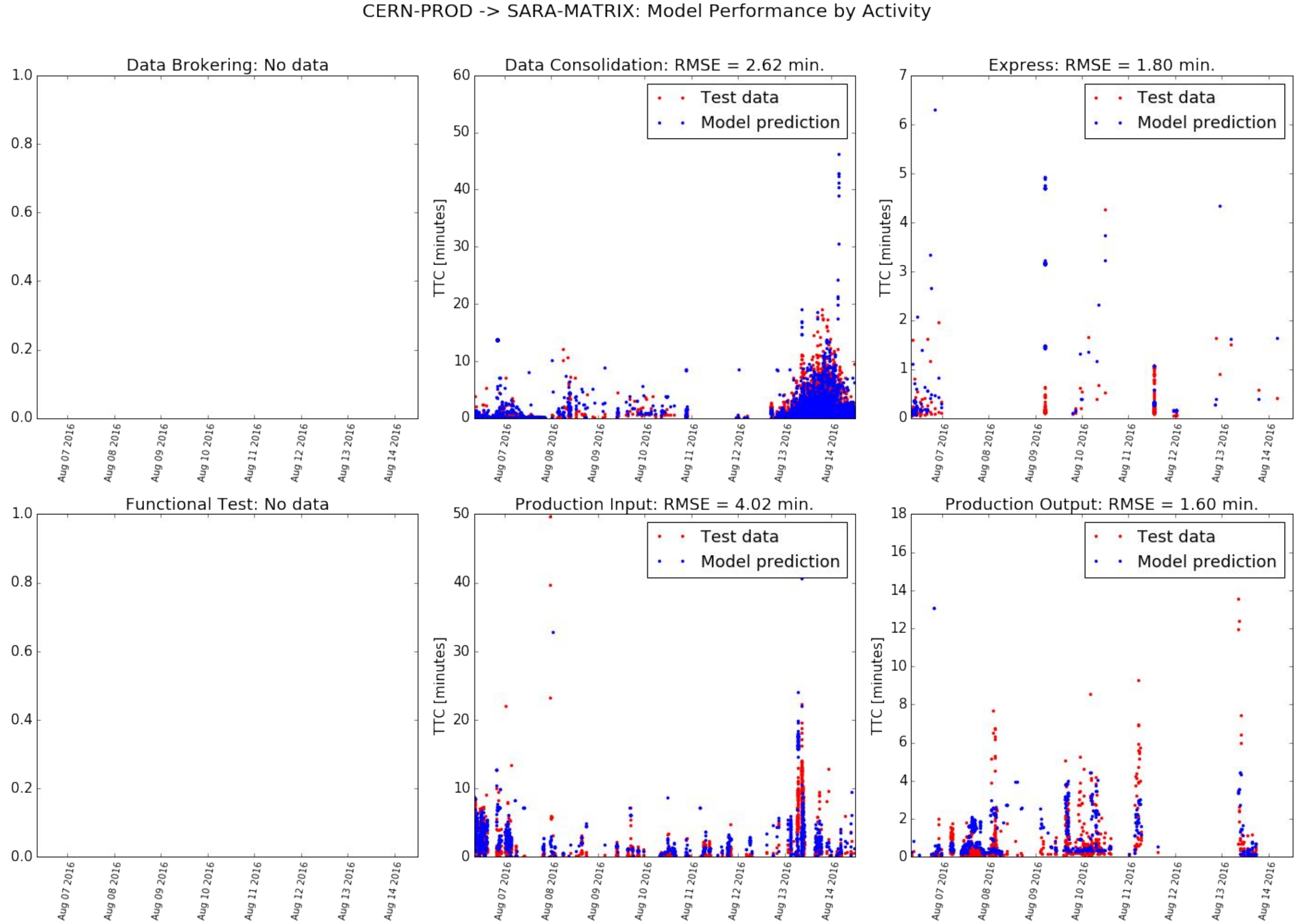
Time to complete transfer estimator



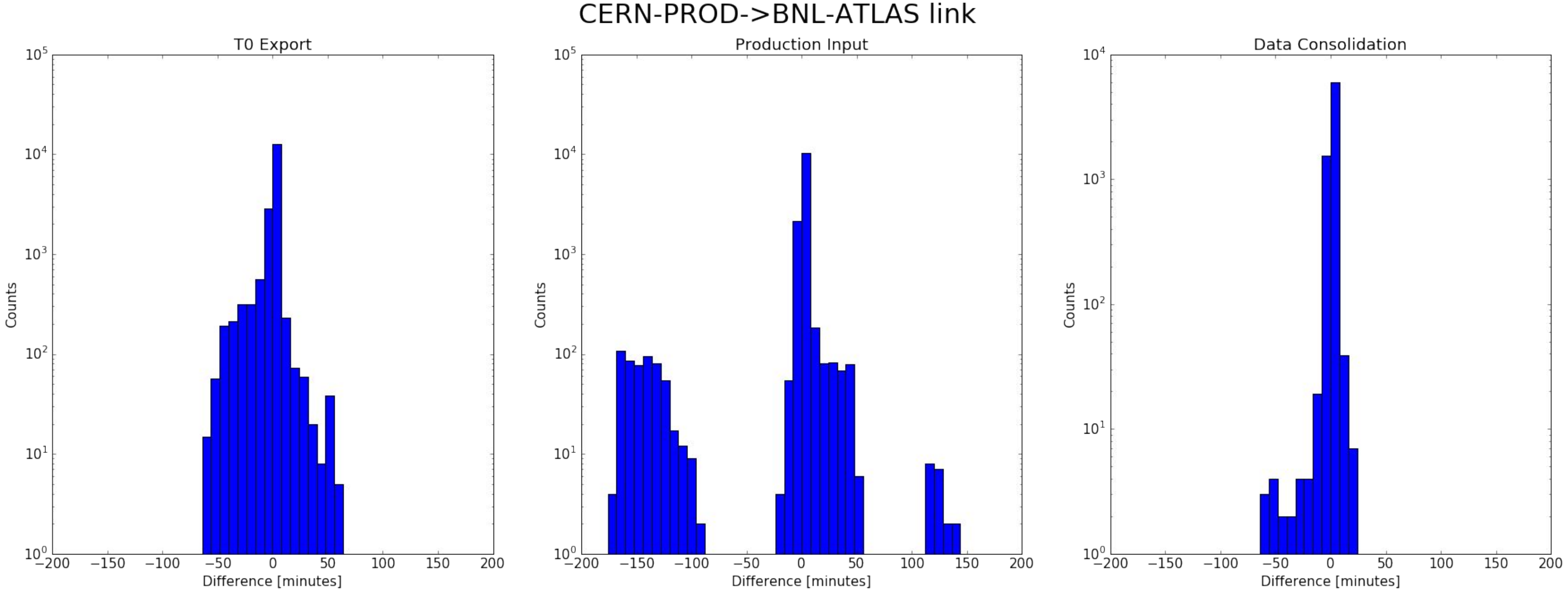
Time to complete transfer estimator



Time to complete transfer estimator



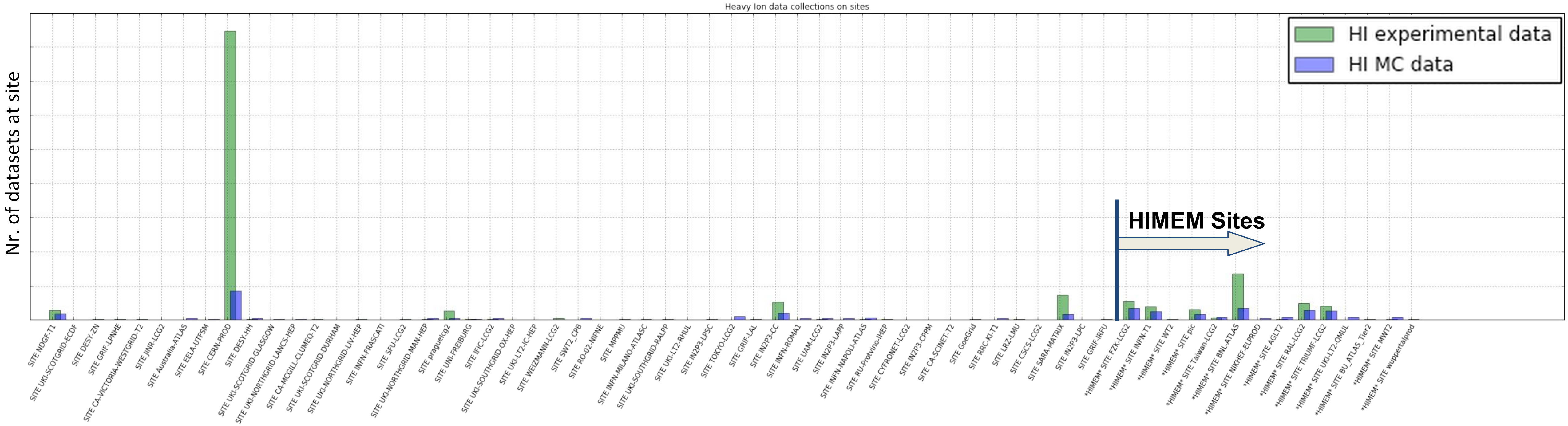
Time to complete transfer estimator



So far we are happy with the first results of the estimator

- ↪ Improvements can now occur in parallel to the bigger data placement activity
- ↪ Understand occasional multimodality of model output w.r.t. different activities
- ↪ Try different regression models (boosted decision trees, networks, SVM)

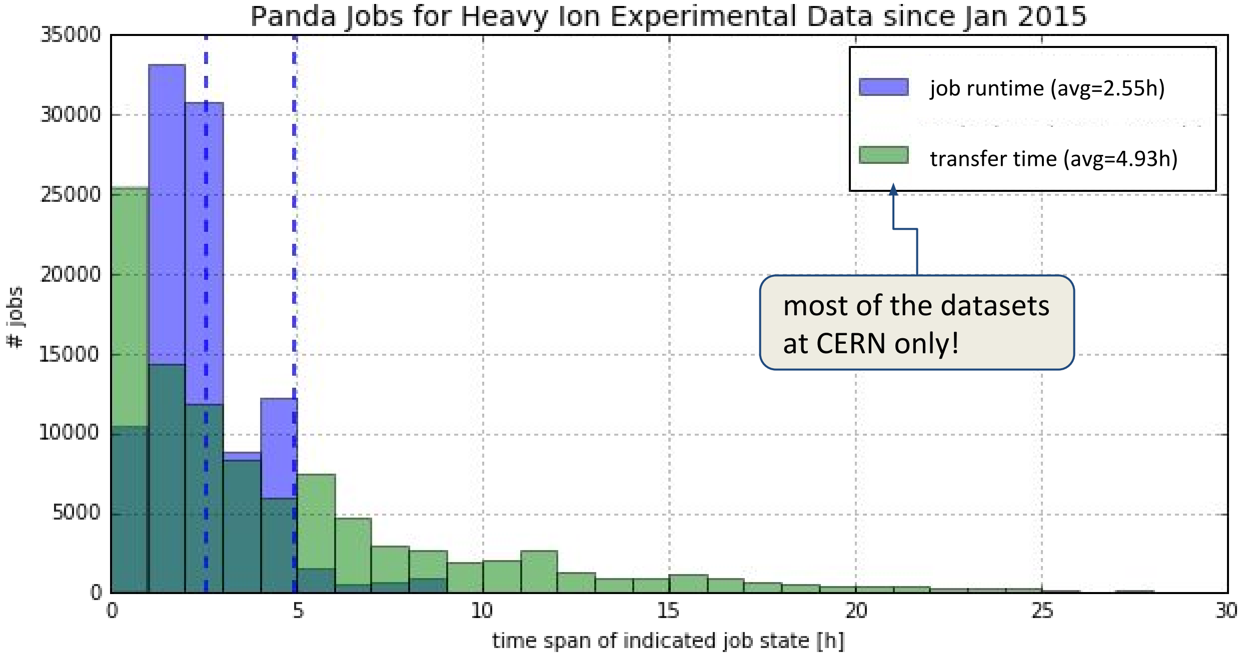
Back to Heavy Ion data placement

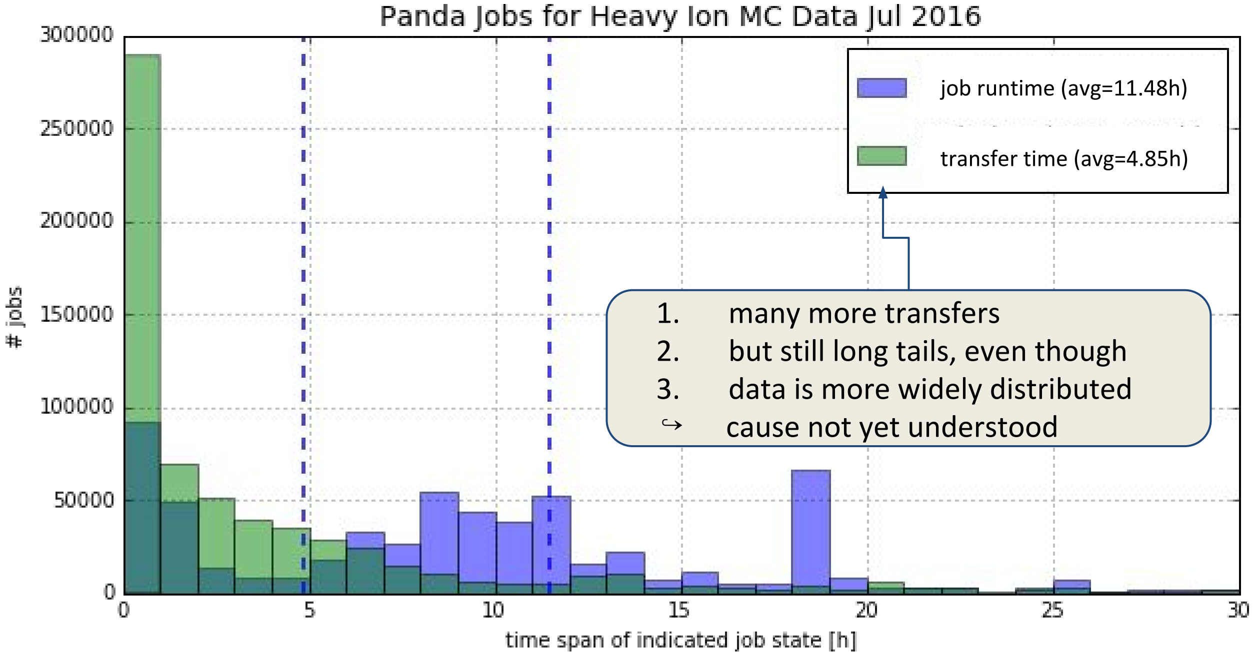


Many of the datasets only at CERN — results in transfer queuing delays
Jobs spend a lot of time waiting for input data — wait time distribution with long tails

↪ With a distance estimator we can quantify data placement improvements
↪ But we cannot overfill sites — queued files and space important for feedback loop

Back to Heavy Ion data placement



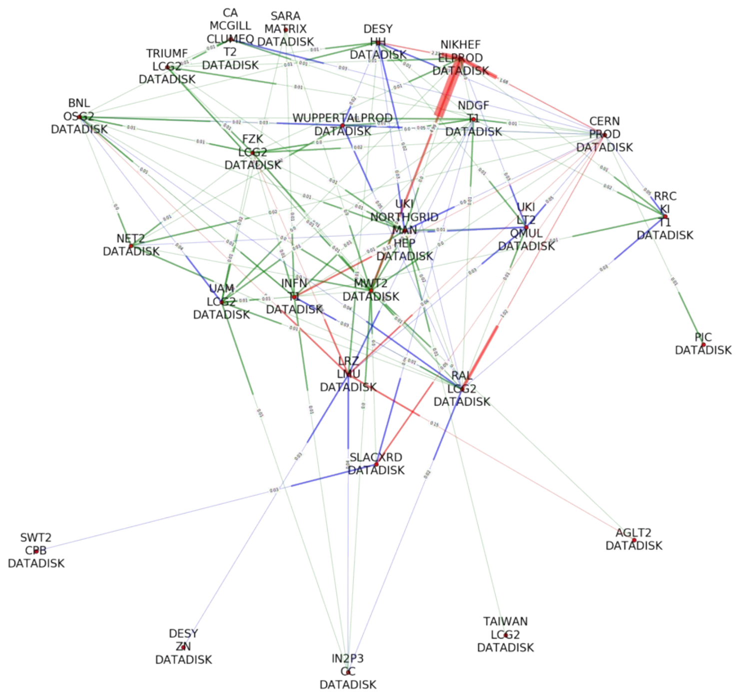


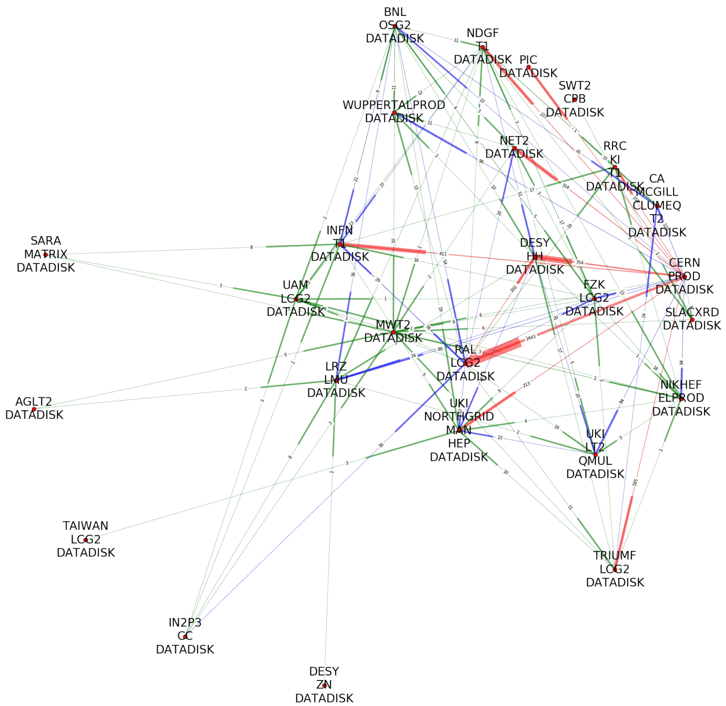
Many of the datasets only at CERN — results in transfer queuing delays

Jobs spend a lot of time waiting for input data — wait time distribution with long tails

- ↪ With a distance estimator we can quantify data placement improvements
- ↪ But we cannot overfill sites — queued files and space important for feedback loop

Back to Heavy Ion data placement





Many of the datasets only at CERN — results in transfer queuing delays

Jobs spend a lot of time waiting for input data — wait time distribution with long tails

- ↪ With a distance estimator we can quantify data placement improvements
- ↪ But we cannot overfill sites — queued files and space important for feedback loop

Summary

DDM Network metrics centrally stored and made available

Estimator for **time-to-complete** of transfers using machine learning methods

- ↪ Good agreement but need to better understand model output
- ↪ Point improvements can be made in parallel, e.g., other learning methods, …

Heavy Ion data placement selected as first constrained focus study

- ↪ Demonstrate feasibility of machine learning methods for automated improvements
- ↪ Have a full chain and workflow in place
- ↪ Eventually, open up automatic rebalancing for all types of data

In the future...

- ↪ For full studies, we will require the move from scikit-learn to MLLib/Spark
- ↪ Incremental steps in agreement with our human operators