Modelling High-Energy Physics Data Transfers

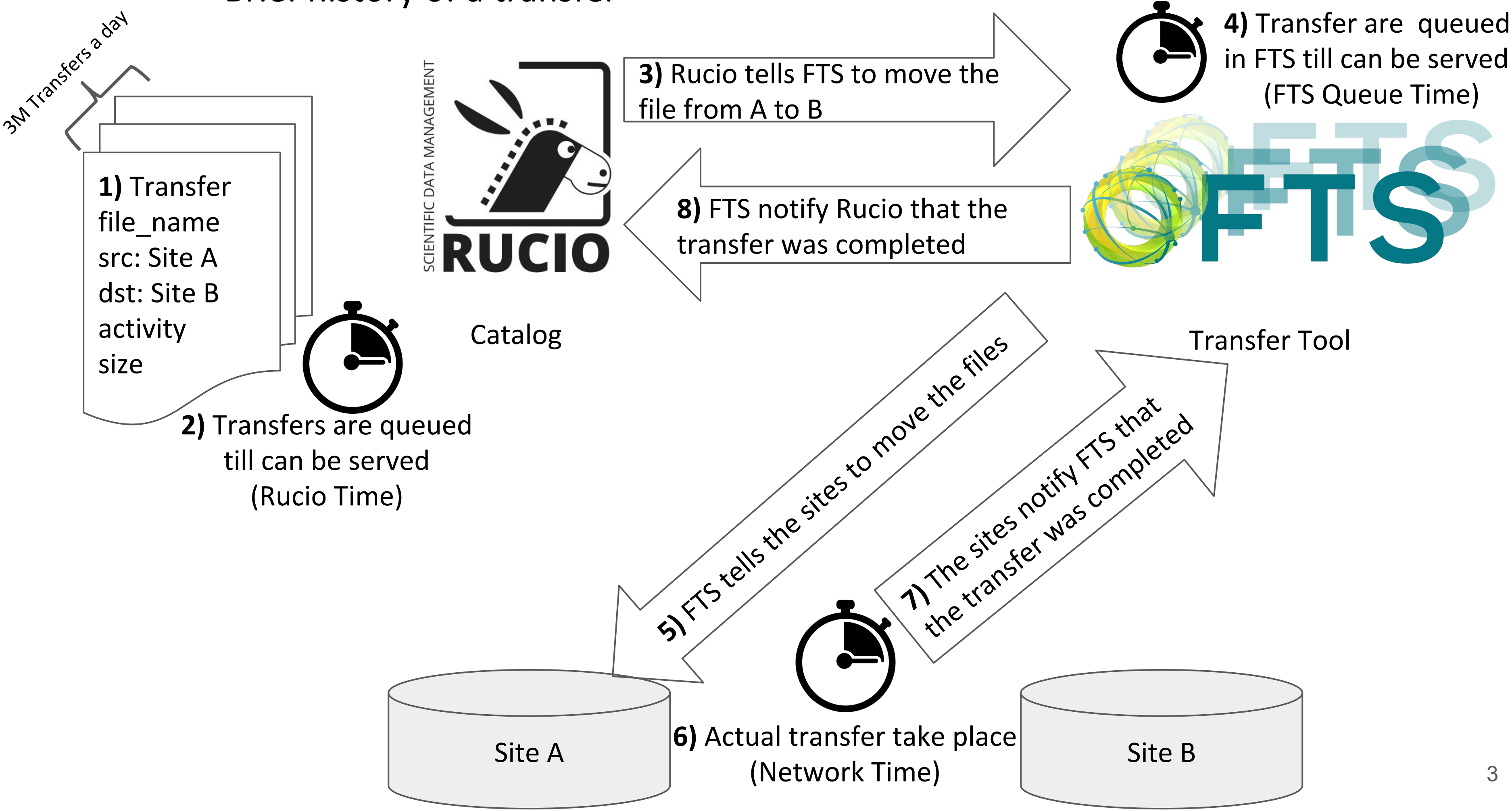
Joaquin Bogado [1]
Mario Lassnig [2]
Fernando Monticelli [1]
Ilija Vukotic [3]
Javier Diaz [1]
On behalf of the ATLAS collaboration

1. Universidad Nacional de la Plata, Buenos Aires, Argentina
2. CERN, Geneva, Switzerland
3. University of Chicago, Illinois, USA

Introduction

- The study is based on data provided for a tool called Rucio.
- Rucio is the Distributed Data Management system in charge of managing all ATLAS data on the grid. Recently open to the community.
- Typical Rucio tasks
  - Transfer data to/from sites around the world (relaying in another tool called FTS).
  - Delete data from sites.
  - Enforce the experiment computing model.
- Objective of this work (part of a PhD ongoing)
  - to understand the relation between tools in the scientific data management environment.
  - to identify problematic scenarios that could lead to delays in transfer times.
  - to make predictions about how much time a transfer will take, at submission time.
  - to allow decision makers to improve the performance of the architecture using this knowledge.

Brief history of a transfer



3M Transfers a day

1) Transfer
file_name
src: Site A
dst: Site B
activity
size

2) Transfers are queued till can be served (Rucio Time)

Catalog

3) Rucio tells FTS to move the file from A to B

4) Transfer are queued in FTS till can be served (FTS Queue Time)

Transfer Tool

5) FTS tells the sites to move the files

6) Actual transfer take place (Network Time)

7) The sites notify FTS that the transfer was completed

8) FTS notify Rucio that the transfer was completed

Site A

Site B

Challenges

- Dealing with real, unfiltered, unlabeled data, directly dumped from Rucio's DataBase.
- Data is difficult to filter. Can't discard anything without being sure is not related with the variables we need to study.
- The amount of data is huge, about 3 million transfers per day. The dataset used to study one busy link (CERN to Chicago) has 2.5 million transfers for a 5 days timespan.
- Overall system is rich in complex interactions. I.e: Tape systems are weird.
- Rucio is composed of several demons, each with several instances.
- Different instances of FTS servers, each of which using different configurations.

The FTS Queue Time problem

- Meanwhile **Rucio Time** and **Network Time** are relatively well understood, **FTS Queue Time** has proven to have big dispersion without clear reasons.
- **Black box approach used** to understand FTS Queue Time using information provided by Rucio's Database about the transfers.
- Can the number of queued transfers in FTS be approximated using Rucio's Database information only?
- The main hypothesis of this work was that the number of queued transfers in FTS could be inferred from the number of submissions from Rucio to FTS and from the time the transfers took in the network.
- The rate of files exiting the queue must be equal to the rate of files exiting the network. If the submission rate is bigger than the network exit rate, then FTS will queue the exceeding transfers.

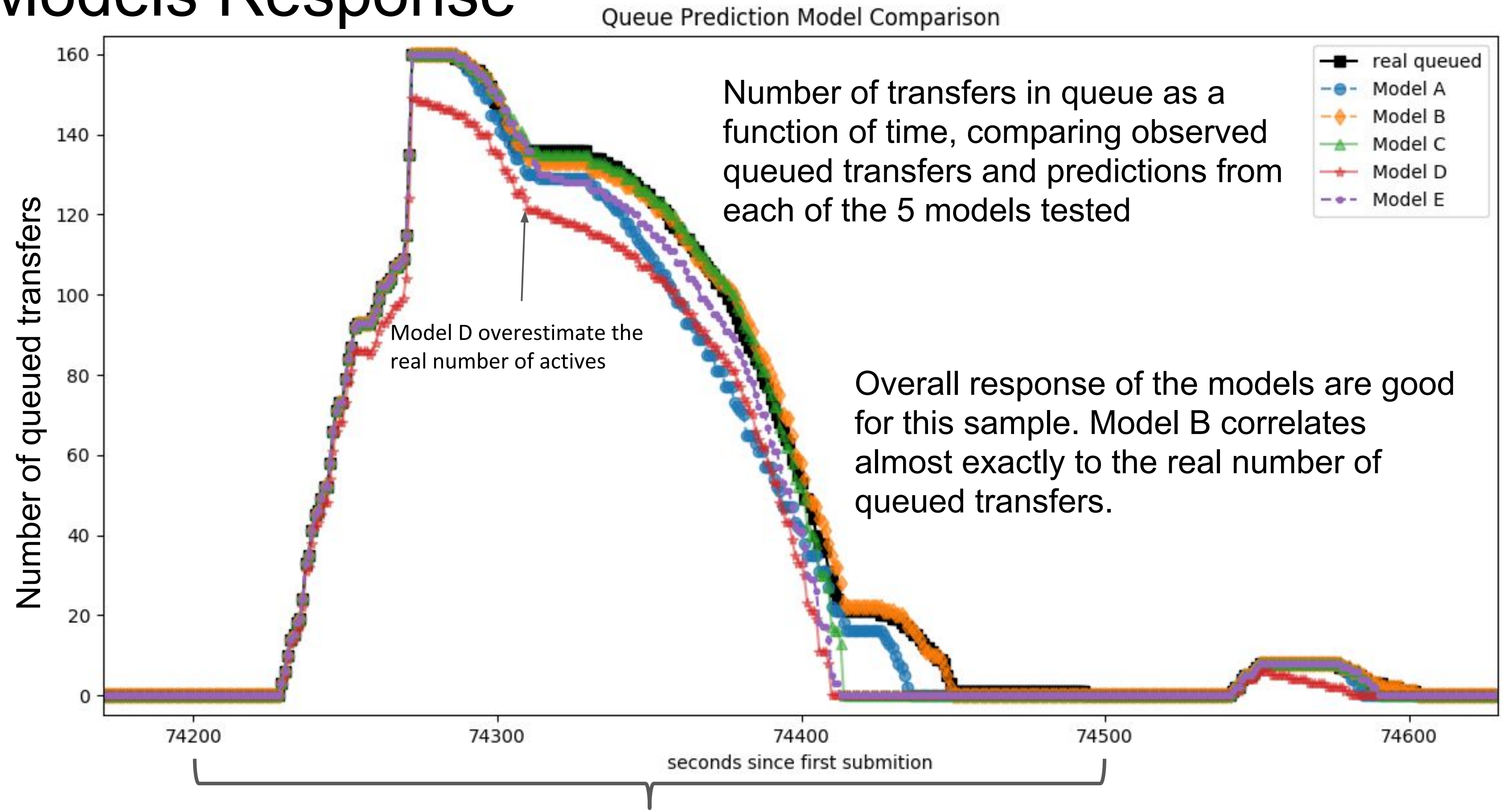
Models assumptions and considerations

- Transfers are simulated using different parameters. Difference in these parameters define a model.
- The simulated FTS response time is zero. We know it's not the case.
- Communications between FTS and the Sites is not simulated.
- Behavior per activity is not simulated. All transfers have the same priority.
- Some filters were applied to the data.
  - Traffic for one link was selected to do the study (Chicago to Michigan)
  - 7 days of transfers were took into account
  - Another set of 7 days of transfers were used to validate the models
- Models labelled from A to E were built incrementally, restricting the amount of information available for each model in every step.

Simulation process

- **Five models** were studied, the difference between them relays on how the number of active transfers and the available bandwidth is calculated.
- **Every second** in the lifetime of the transfer is simulated.
- Creation and submission times are took from Rucio's Database information.
- Transfers sent to a simulated FTS Queue until they can be took out.
- The number of transfers that can be served (the **number of concurrent active transfers**) is calculated based on real data and **depends on the model**.
- When a transfer exit FTS Queue, it's Network Time is simulated dividing the available bandwidth among the current amount of active transfers.
- The **available bandwidth calculation depends on the model**.

Models Response

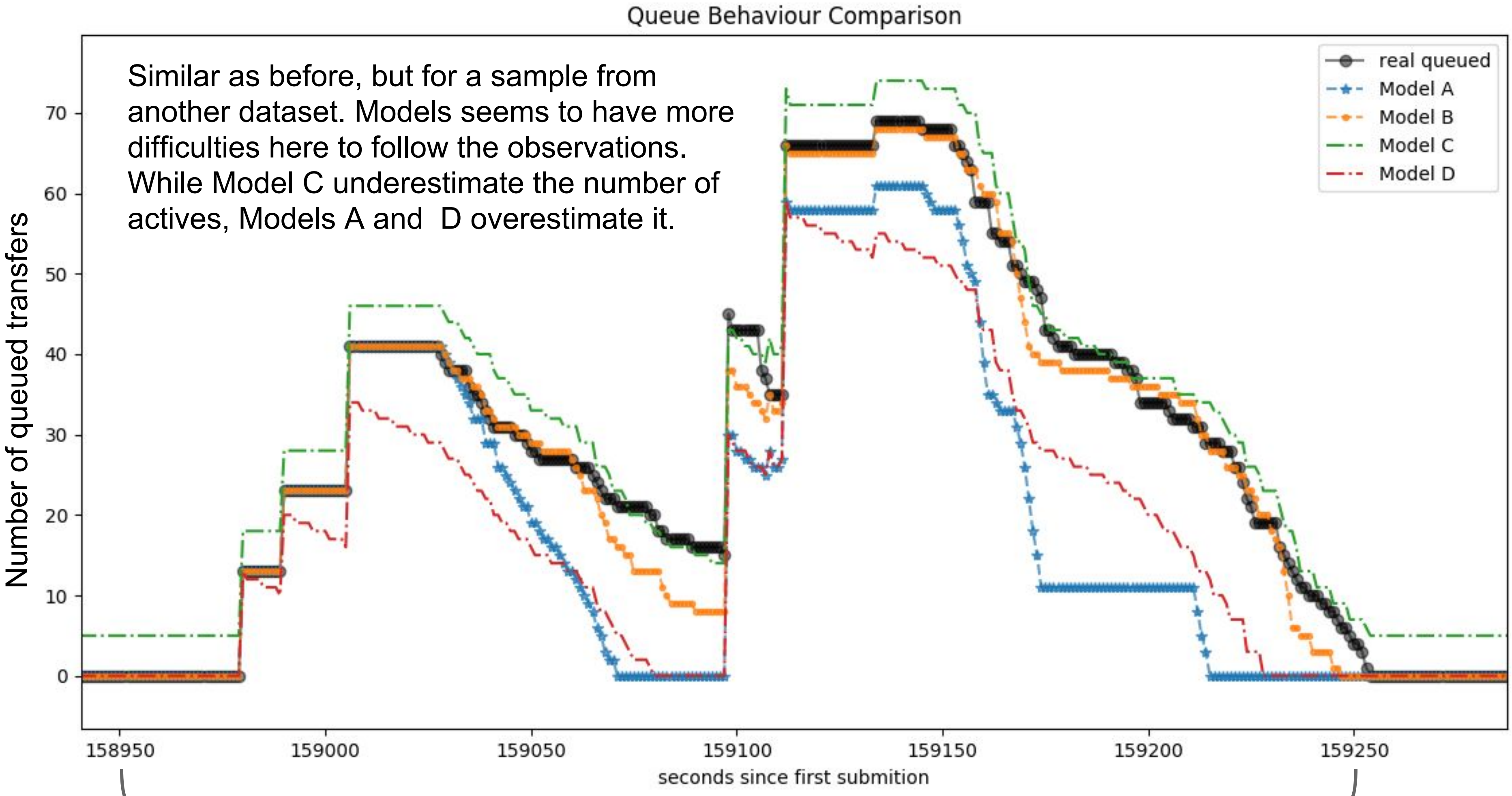


Number of transfers in queue as a function of time, comparing observed queued transfers and predictions from each of the 5 models tested

Model D overestimate the real number of actives

Overall response of the models are good for this sample. Model B correlates almost exactly to the real number of queued transfers.

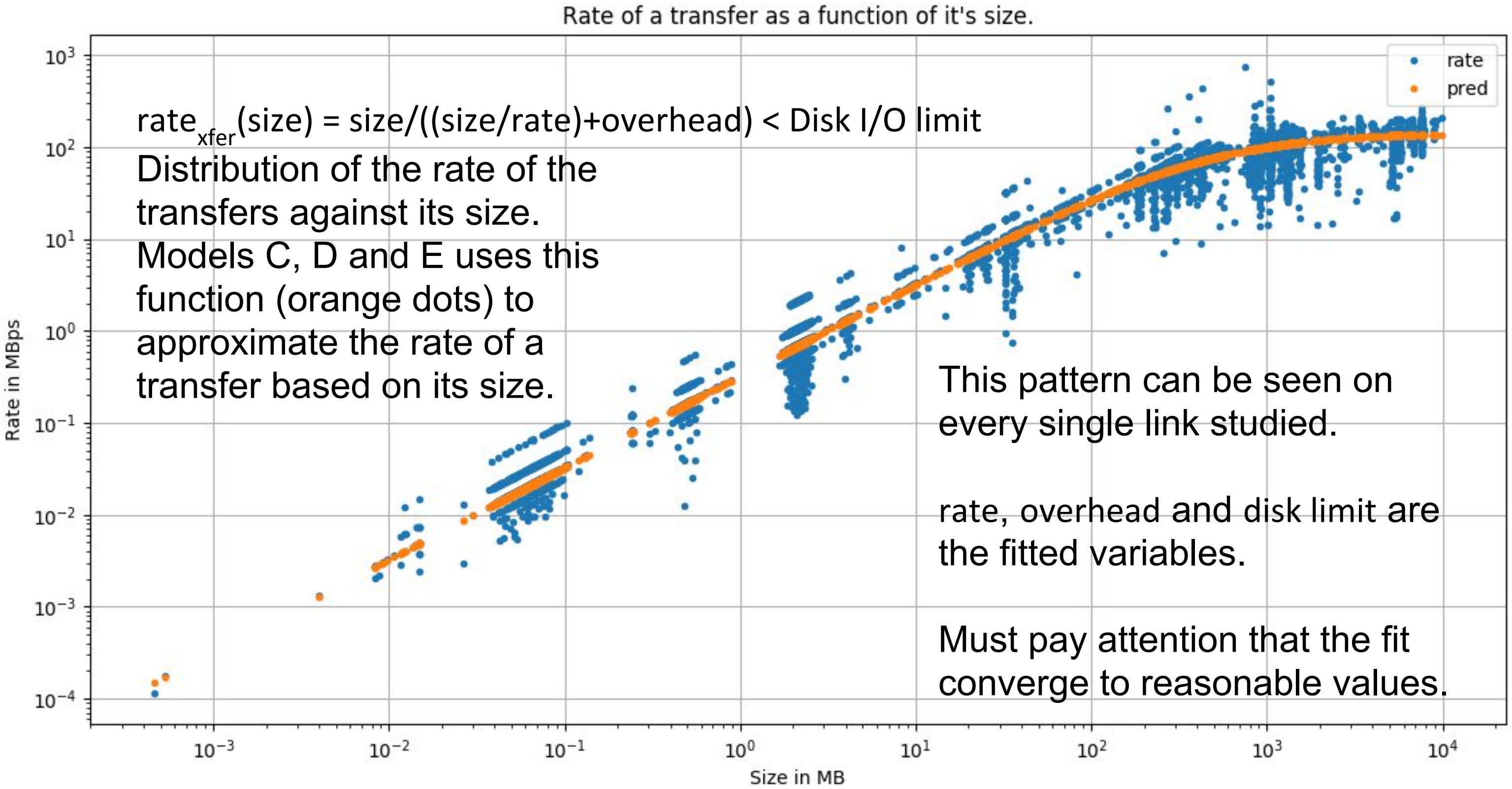
5 minutes



Similar as before, but for a sample from another dataset. Models seems to have more difficulties here to follow the observations. While Model C underestimate the number of actives, Models A and D overestimate it.

5 minutes

Rate approximation for Network Time based on size



$\text{rate}_{xfer}(\text{size}) = \text{size}/((\text{size}/\text{rate})+\text{overhead}) < \text{Disk I/O limit}$

Distribution of the rate of the transfers against its size. Models C, D and E uses this function (orange dots) to approximate the rate of a transfer based on its size.

This pattern can be seen on every single link studied.

rate, overhead and disk limit are the fitted variables.

Must pay attention that the fit converge to reasonable values.

Prospects and Limitations

- Models heavily reliant on the number of active transfers and bandwidth approximations.
- Models B, D and E are very promising and generalize well to other links and scenarios.
- Models B and D are simple and generalize well, even over other links studied later but not included in this work.
- Models systematically fail to predict the Queue Time for individual transfers.
- Models can be improved including other variables.
  - Static number of concurrent active transfers limit per Link
  - Sites maximum input/output active transfers per Site
  - Activity shares are used to control how transfers exits FTS queue

Optimizer related configurations per FTS server

Thanks for your attention.

Questions?

Joaquin Bogado - jbogadog@cern.ch

Backup

Model A

As was determined by in-depth analysis, the bandwidth available for transfers is not constant. But by assuming that it is constant during some time period however, it is possible to estimate the transfer rate. Model A uses the size of the transfers and the started at and transferred at timestamps to predetermine the rate based on active islands. The total volume of bytes transferred during this time is then divided by the number of active transfers.

An empirical search was conducted to find the scaling factor in the previous equation, and the best performance of the models over the first dataset was obtained with f = 0.66. We couldn’t find a consistent explanation for this value in the observed data. This factor seems to be not constant as Model A behaves very well only in selected time spans. The reason why f is not 1, and not constant is unclear, and will be kept as a bias to be determined from historical data.

Model B

Doing post-mortem analysis it is possible to calculate the average rate of each transfer instead of the average of a group of transfers. The modifications to Model A involve adding a new predetermined rate to the simulated transfer structure. This rate is used at simulation time to make the transfer finish successfully.

This is the model that achieves the best results, but also the one that has the most information about the transfers available. Most of the data used to feed this model is not available at submission time though. Therefore, this model is not suitable to implement in online systems to make predictions about the transfers in real time, but only allows to validate the underlying model and is useful to make predictions about changes in the configuration of the underlying systems.

Model C

Model C replaces the rate calculation of Model B with a function, as described in Equation (1), which allows to predict the rate as a function of the size of the transfer. As this value is known at submission time, models that use this approach are suitable to be used in real time predictions of the number of queued transfers.

Model C however doesn’t behave well when the bandwidth is underestimated. As some transfers take more time to finish in the simulation than in the observed data, the maximum number of actives used is the observed one, some new transfers could remain in the queue, where the maximum actives (*max active*) are zero, and these transfers will not exit until the next set of transfers trigger the max actives above zero. This problem had been seen in Model A and in with some insignificant effects, also seen in Model B.

Model D

To avoid queue starvation due to lack of actives the simplest solution is to limit the minimum *max active* transfers to 1, meaning if there are transfers in the queue at any given moment the simulator can take a transfer off the queue, and progress it. Model D uses this approach to approximate the maximum number of concurrent active transfers. This approach yields good results, comparable with Model B.

Model E

Model E uses a more sophisticated method to avoid starvation due to lack of active transfers. This method involves smoothing and shifting the observed number of concurrent actives (*`max active`*). In some cases, the method outperforms the results obtained to Models B and D. Yet, overall, Models B and D are more general, simpler, and with comparable results, thus preferred over Model E.