# *Fair-share scheduling algorithm for a tertiary storage system*

**Pavel Jakl**[1] Jérôme Lauret[2]

[1]Nuclear Physics Institute, Academy of Science, Czech Republic

[2]Brookhaven National Laboratory, United States of America

CHEP 2009

Prague, Czech Republic

# Outline

*What do we mean by fair-share scheduling ?*

- Storage challenges at STAR experiment
- Scalla/Xrootd at STAR
- Analysis of a usage scenario

*Key parameters of MSS/HPSS performance*

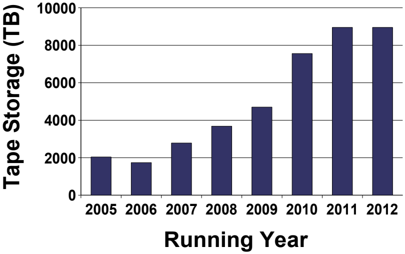
- Number of files per tape mount parameter
- File size paramater

*MSS scheduling to achieve performance and QoS*

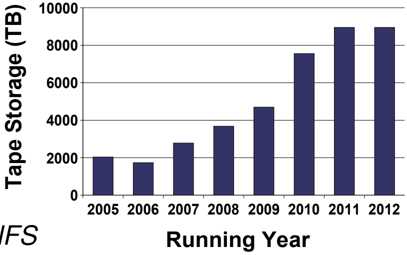
- Scheduling goals and proposed algorithms
- Evaluation of algorithms
- Evaluation results

*Summary*

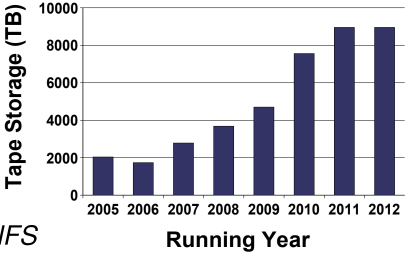
- over 1PB data per year at STAR



- over 1PB data per year at STAR
- *Permanent* location:
  - tape system (MSS): offers several **PBs**
- *Temporary* locations:
  - centralized disk space: **150 TB** *via NFS*
  - distributed disk space: **350 TB** *spread over 1000 nodes*

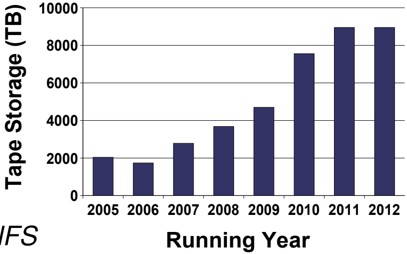


- over 1PB data per year at STAR
- *Permanent* location:
  - tape system (MSS): offers several **PBs**
- *Temporary* locations:
  - centralized disk space: **150 TB** *via NFS*
  - distributed disk space: **350 TB** *spread over 1000 nodes*

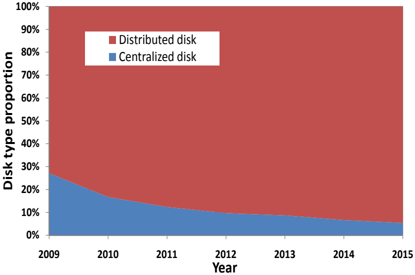



- **distributed** vs centralized disk:
  + very low cost (factor of ∼10)
  + less human resources to maintain
  – worse manageability (one has to build aggregation)
  – no native OS/system provides scalable/workable solution

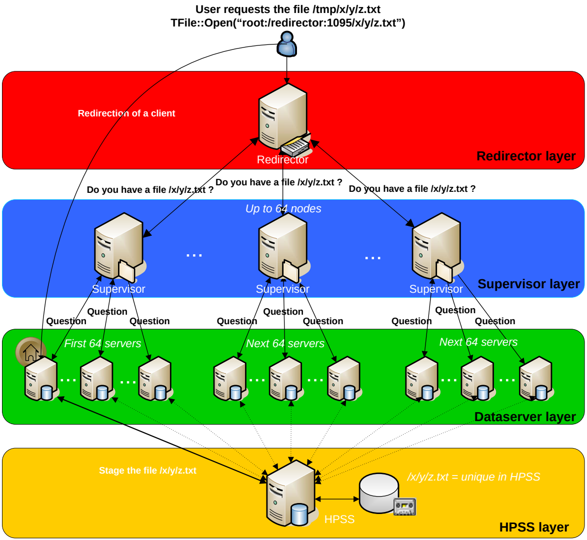
- over 1PB data per year at STAR
- *Permanent* location:
  - tape system (MSS): offers several **PBs**
- *Temporary* locations:
  - centralized disk space: **150 TB** *via NFS*
  - distributed disk space: **350 TB** *spread over 1000 nodes*




- **distributed** vs centralized disk:
  - + very low cost (factor of ∼10)
  - + less human resources to maintain
  - – worse manageability (one has to build aggregation)
  - – no native OS/system provides scalable/workable solution



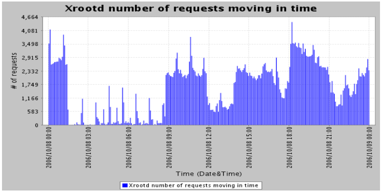
# Quick Scalla architecture overview at STAR

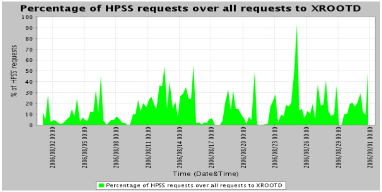



**Each CE hosts SE = sharing of resource**

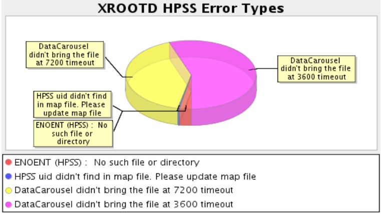
# *Scalla in production and real analysis scenario*

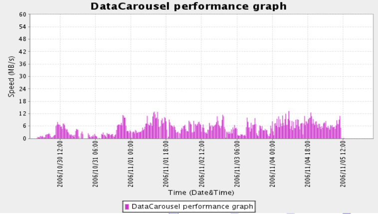
- possible to see up to 35 requests/sec to open files, users use Scalla to access HPSS data-sets



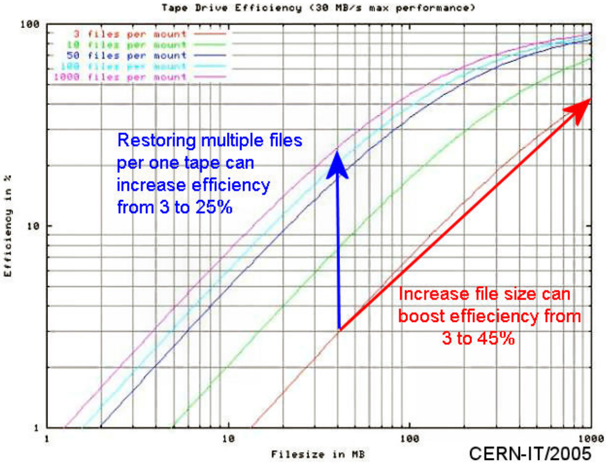




- most of errors are caused by timeouts ⇒ slow performance per a HPSS tape drive (14 at STAR) ⇒ high latencies



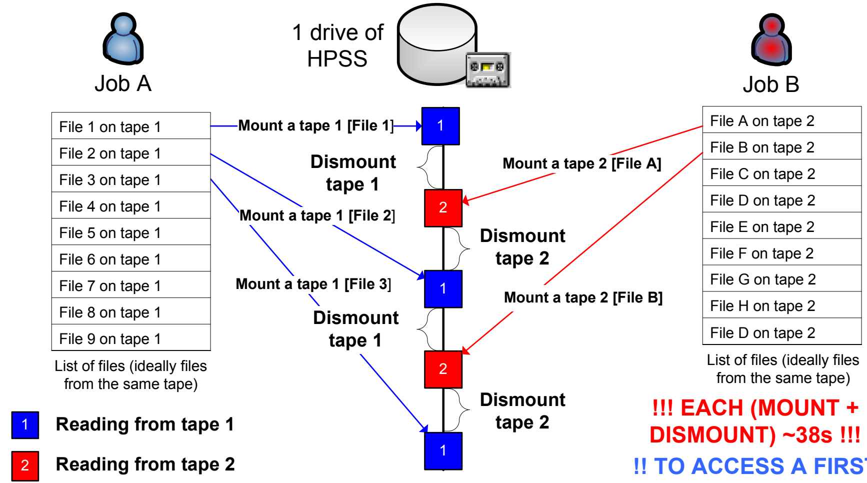



# *Reasons for slow HPSS performance*



# *Impact of sequential processing on a HPSS drive*

- the processing of data in HENP applications has sequential behaviour

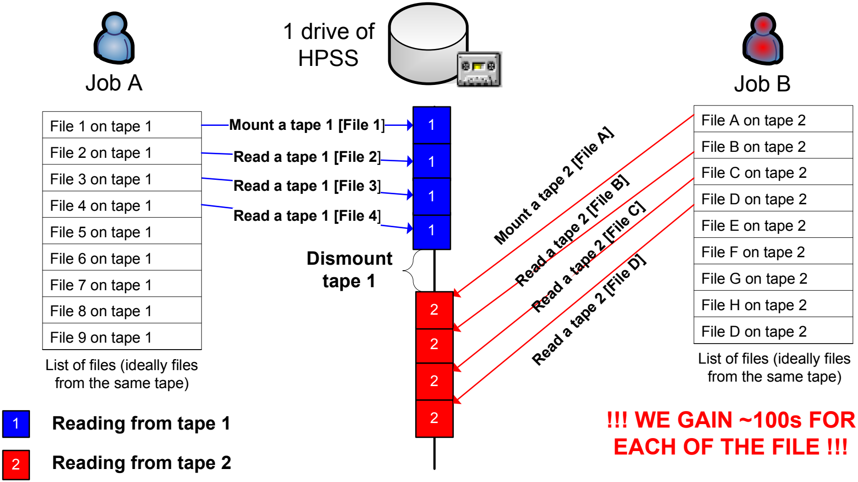



**!!! EACH (MOUNT + DISMOUNT) ~38s !!!**

**!! TO ACCESS A FIRST FILE ON THE TAPE ~72s !!**

# *Influence of the pre-staging on a HPSS drive*

- Pre-staging ⇒ each job publishes its whole intend



# *Proof of prestaging on the production system*

**100% of improvement**

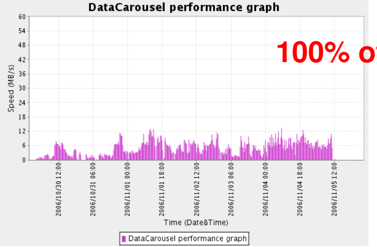



*Figure:* Before pre-staging

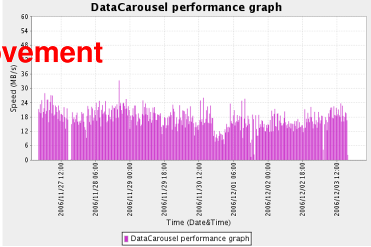
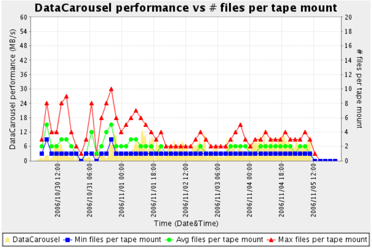
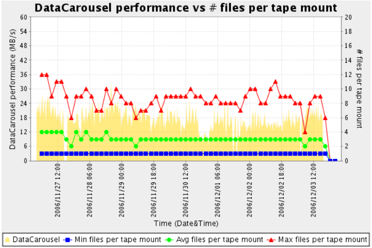


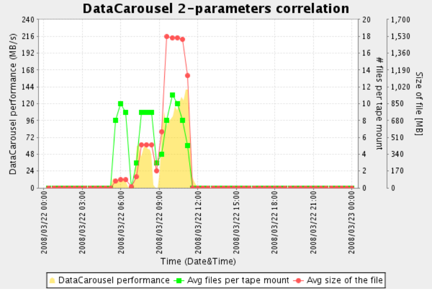

*Figure:* After pre-staging



# *File size impact on HPSS performance*

- size of file has the biggest impact on HPSS performance
- realized 3 independent tests having 15 files/per tape mount and varying in average file size:
  - 80 MB MuDst files ⇒ 2% efficiency (files used for analysis)
  - 500 MB event files ⇒ 12% efficiency
  - 1500 MB MC files ⇒ 26% efficiency




$$\text{Efficiency} = \frac{\text{throughput}}{\text{max\_speed}} \; [\%]$$

# *Outline*

What do we mean by fair-share scheduling ?
- Storage challenges at STAR experiment
- Scalla/Xrootd at STAR
- Analysis of a usage scenario

Key parameters of MSS/HPSS performance
- Number of files per tape mount parameter
- File size paramater

*MSS scheduling to achieve performance and QoS*
- Scheduling goals and proposed algorithms
- Evaluation of algorithms
- Evaluation results

Summary

# *Scheduling problem of MSS and goals*

- requests can be made by many users and asking for several different datasets spread over many distinct tapes
- are naturally dis-organized (ahead of the time) affecting an overall performance and a delay of delivery in respect to the users (QoS)
- a focus is to prevent resource starvation while introducing speed and fair-share
- an algorithm should incorporate mentioned key performance parameters
  - scheduling requests from the same tapes (i.e. sorting according to the tape location)
  - scheduling requests with bigger file size
- an ultimate goal is to *"re-organize"* requests and deliver
  - sustained data throughput
  - maximal quality of service (QoS)

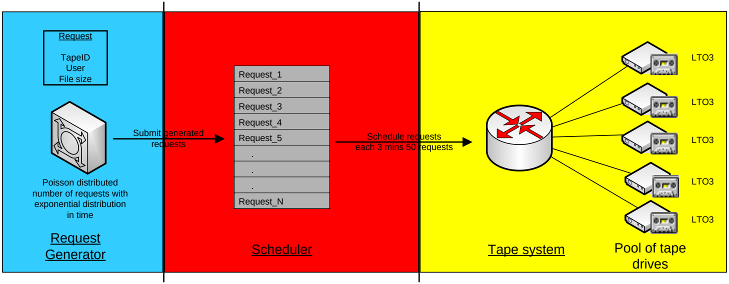
# *Proposed scheduling algorithms*

- FIFO (First In First Out)
  - serving in the order of the arrival (first coming user can feed the system for a long time)
- WFQ (Weighted Fair Queuing) -
  - each user has own queue that is weighted by an assigned priority
  - user with high priority can feed the system for a long time
- WFSG (Weighted Fair-Share Grouping) -
  - the priority is being dynamically adjusted according to the previous history of the user
  - 3 parameters are linearly combined:
    1. Number of files per tape 0.6
    2. Usage of the system 0.3
    3. Size of the file 0.1

    where each of them has assigned weight

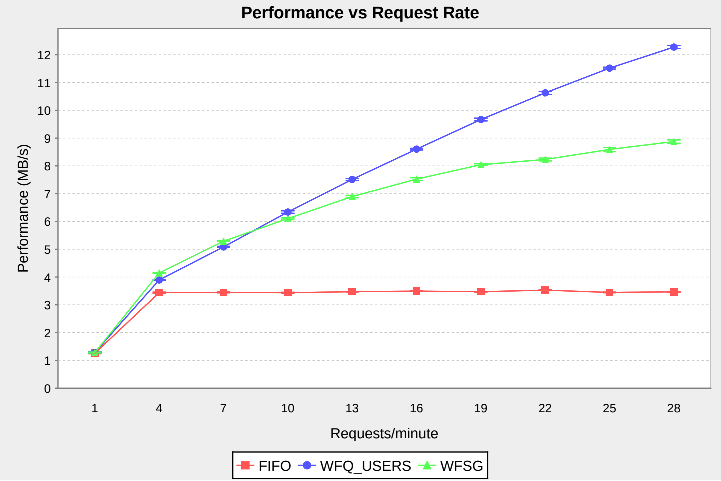
# Evaluation of algorithms

- we have build MC continuous time-discrete event based simulator of HPSS
  - supports robotics operation (i.e. switching of a tape)
  - simulates mounting, dismounting, seeking, streaming operations of tape drives




- 3 main evaluation parameters:
  - Performance - an average data throughput measured in MB/s
  - Delay of request - an average delay of request in the system
  - QoS - percentage of successful requests satisfied in a timeout

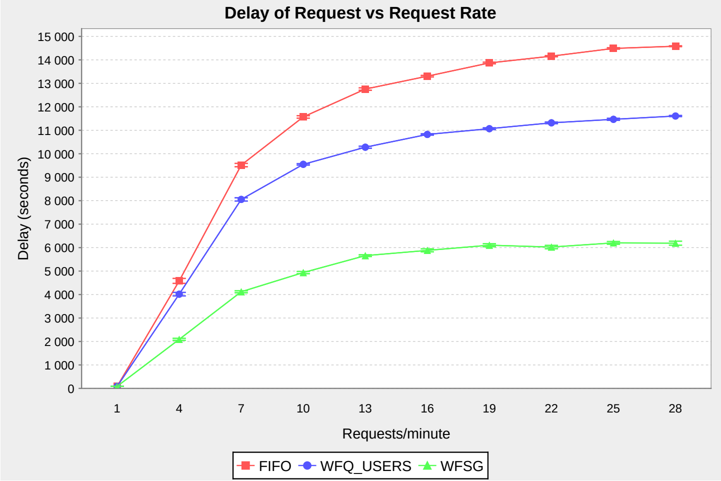
# *Performance vs request rate*




*the higher the number, the better the performance*

**best ⇒ WFQ**

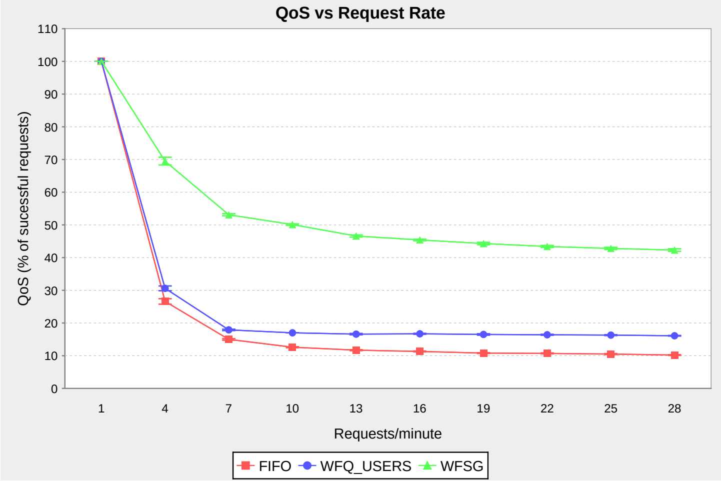
# *Delay of request vs request rate*




*the smaller the delay to deliver files, the better*

**best ⇒ WFSG**

# *Quality of service vs request rate*




*the higher success within our defined time, the better*

**WFSG ⇒ best algorithm**

# *Outline*

What do we mean by fair-share scheduling ?
- Storage challenges at STAR experiment
- Scalla/Xrootd at STAR
- Analysis of a usage scenario

Key parameters of MSS/HPSS performance
- Number of files per tape mount parameter
- File size paramater

MSS scheduling to achieve performance and QoS
- Scheduling goals and proposed algorithms
- Evaluation of algorithms
- Evaluation results

*Summary*

# *Summary*

- we have shown and demonstrated rational behind key performance parameters of the HPSS
  - future runs in STAR have optimal file size
  - pre-staging technique has to be used for the efficient tape optimizations
- scheduling algorithm should not only incorporate performance parameters but also has to be "enough" fair to the users
- simulation of the tape system distinguished efficient fair-share scheduling algorithm
  - we recommend Weighted Fair-share Grouping (WFSG) algorithm to achieve good throughput, maximal QoS and lowest delay
- Future work:
  - an implementation of the WFSG algorithm into the production system
  - an measurement of the algorithm's efficiency in the production system

# International conferences/workshops


- P. Jakl, J. Lauret, A. Hanushevsky, A. Shoshani, A. Sim
  *From rootd to xrootd: From physical to logical file*
  *Proc. of Computing in High Energy and Nuclear Physics (CHEP'06)*
- P. Jakl, J. Lauret, A. Hanushevsky, A. Shoshani, A. Sim, J. Gu
  *Grid data access on widely distributed worker nodes*
  *Proc. of Computing in High Energy and Nuclear Physics (CHEP'07)*
- P. Jakl, J. Lauret
  *Efficient access to distributed data: A "many" storage element paradigm*
  *Diploma thesis, Czech Technical University, Prague (2008)*