Metadata Performance Improvements

Presentation

for

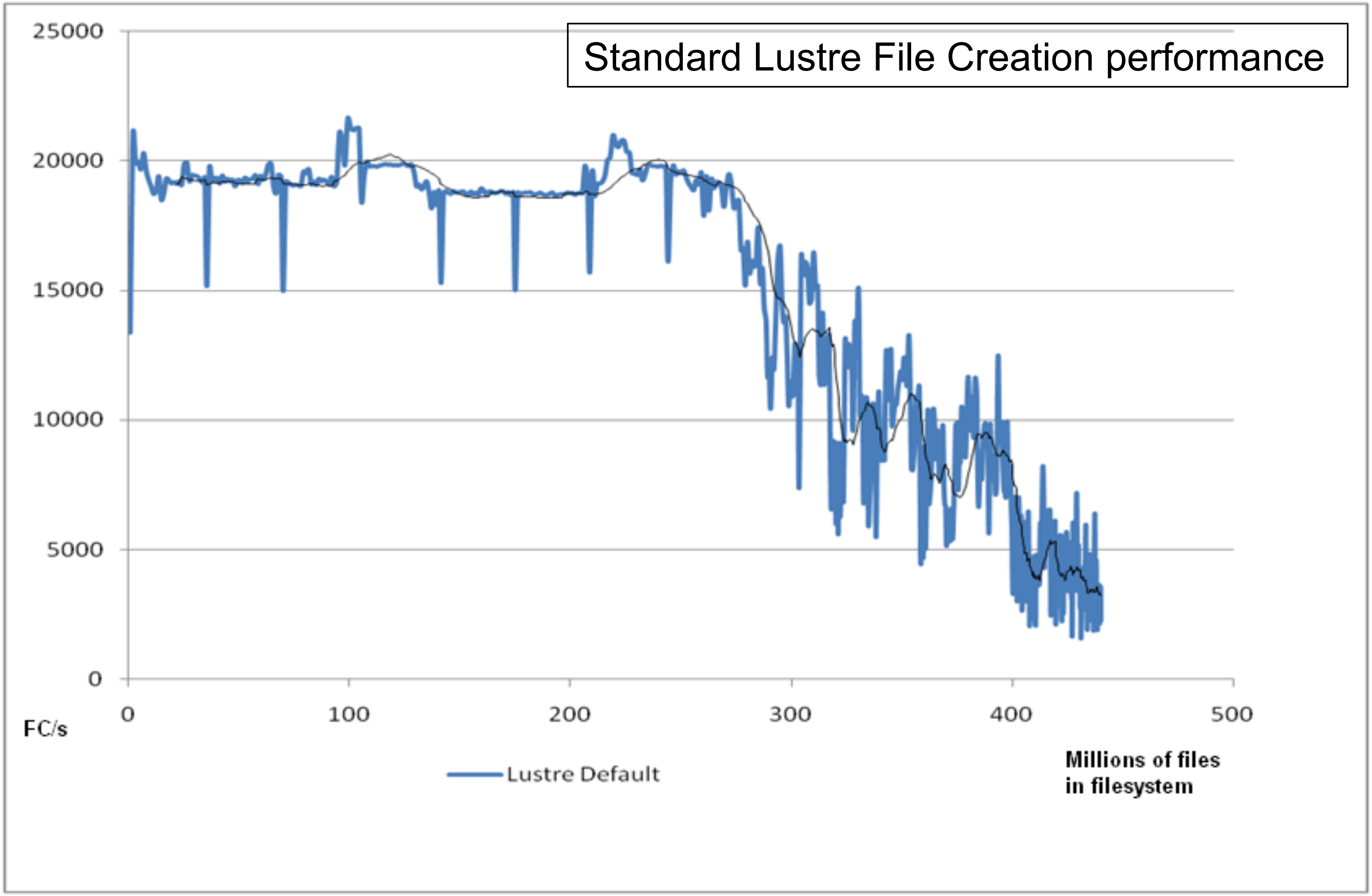
LUG 2011

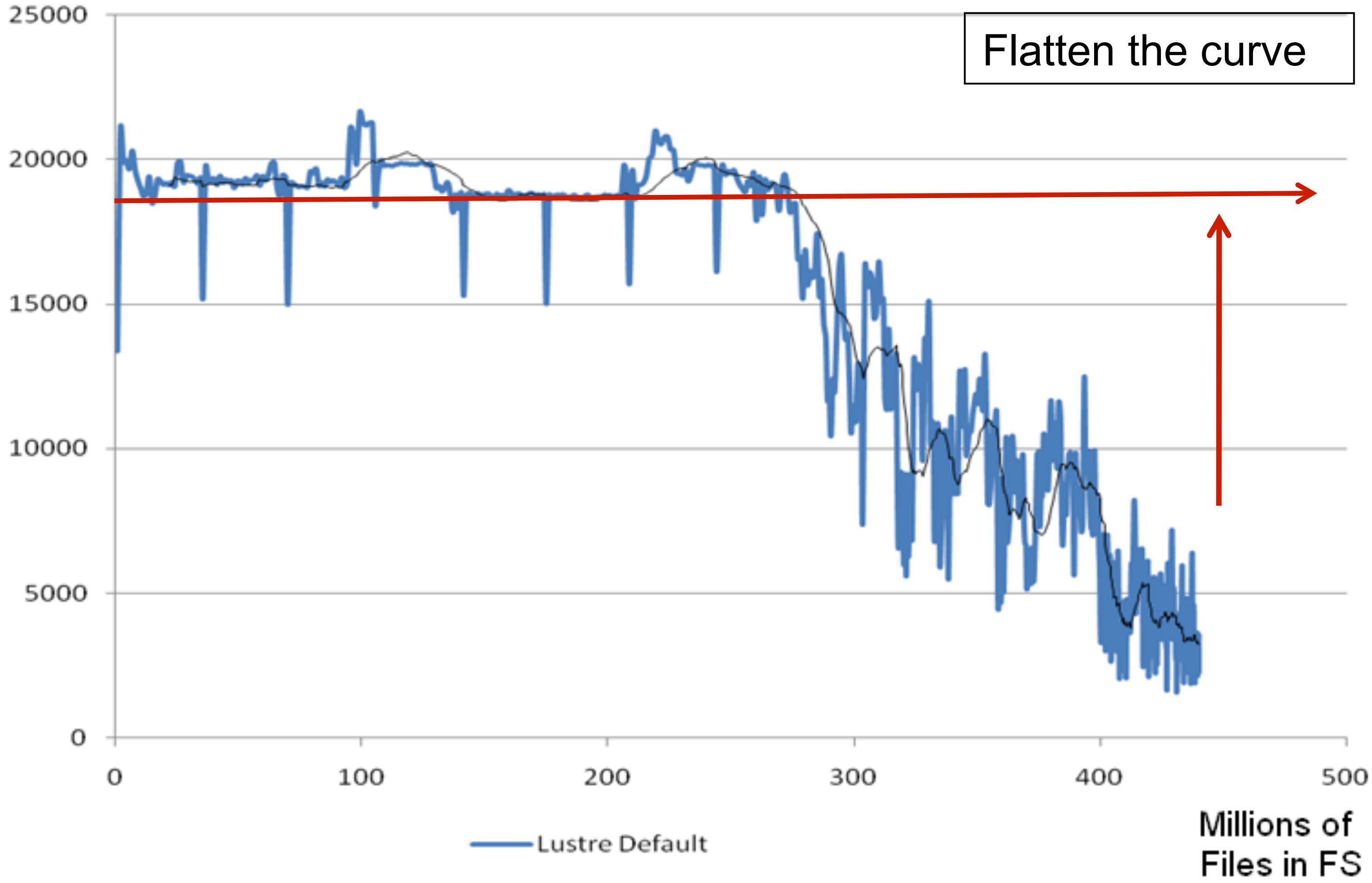
Ben Evans
Principal Software Engineer
Terascala, Inc

The Problem:

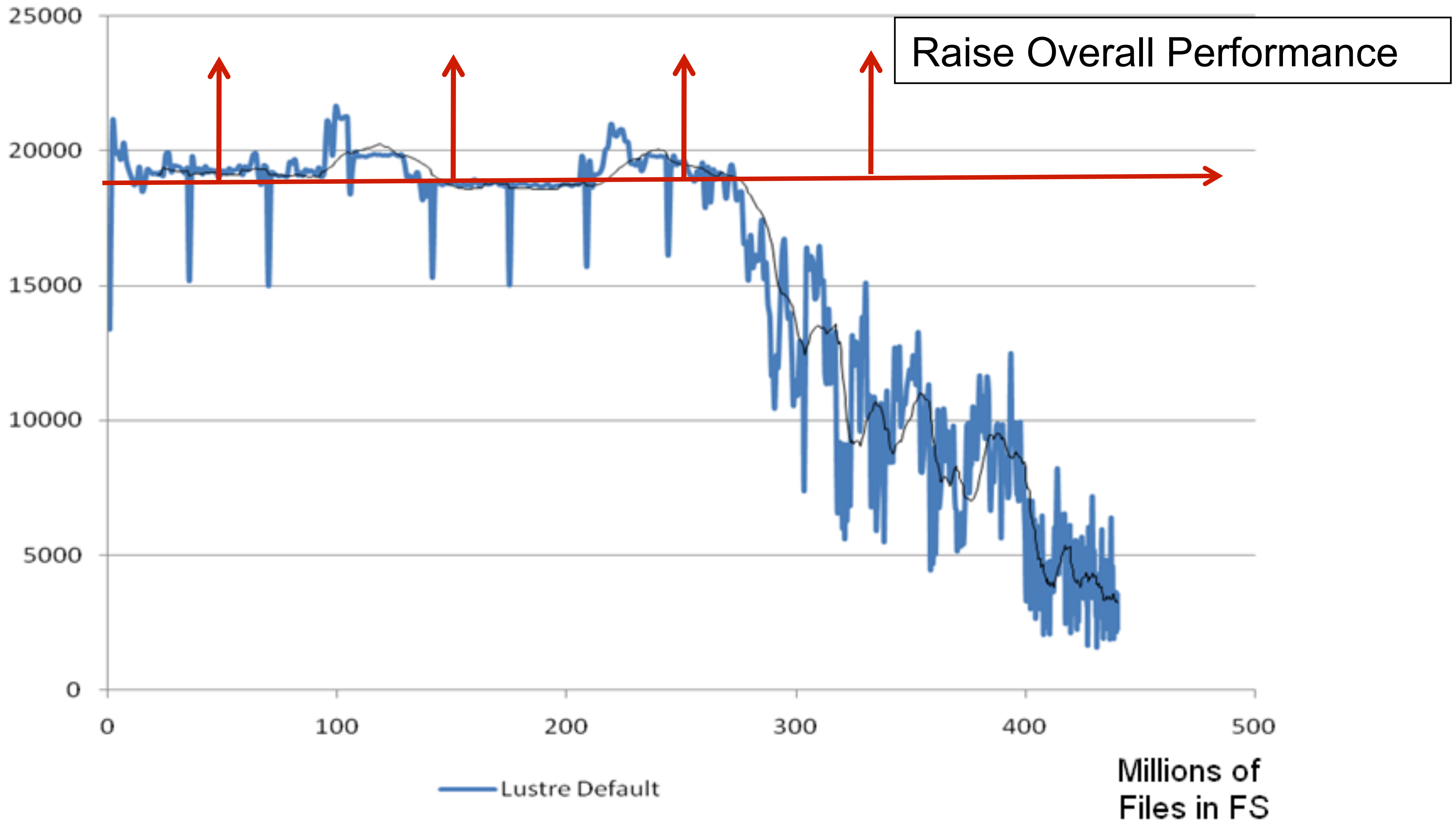
- "Time spent creating files is time taken away from compute cycles"

- Two distinct problems need to be addressed:
  1. OST performance impacts file create performance
  2. Allowing the MDS to go faster


Standard Lustre File Creation performance
25000
20000
15000
10000
5000
0
FC/s
0
100
200
300
400
500
Millions of files in filesystem
Lustre Default



Flatten the curve



Raise Overall Performance

Test Environment:

- ***Jaguar, a Cray XT4 system***

Oak Ridge National Laboratory

- 7832 computes nodes, each with a quad-core AMD Opteron @ 2.1 GHz, 8 GB of memory.
- SeaStar 2.1 NICs in a 3D torus configuration -- ~1.8-2.0 GByte/s injection bandwidth, ~3.3 GByte/s in each direction on the links, full duplex (~6.6 GB/s total).
- 48 LNET routers, dual-core AMD Opterons @ 2.6 GHz and 8 GB of memory, DDR IB links to IB fabric

Lustre Changes Terascala Made:

- Low watermark detection
- Preallocation schemes
- Directory locking during creates

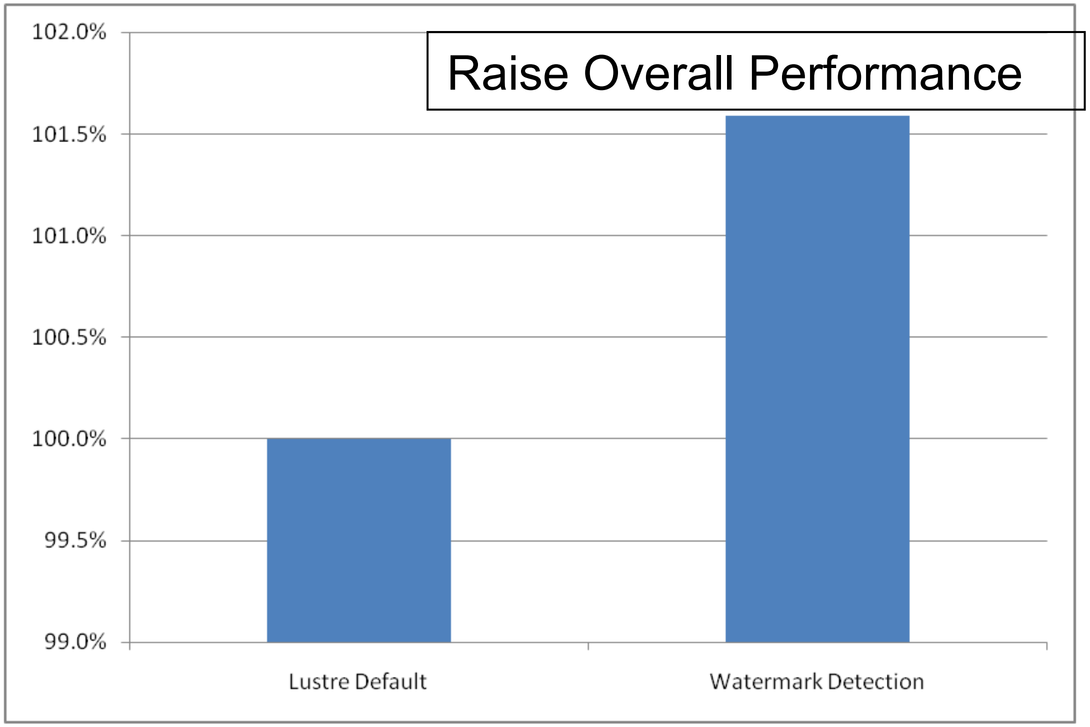
- Developed under 1.8.x
  - Portable to 2.x

Low Watermark Detection:

- MDS keeps a list of preallocated objects on each OST
- When the low watermark is passed, the MDS instructs an OST to create more items
  - Instructions are to create files in small chunks (typically 32)

Low Watermark

- Current scheme triggers low watermark at alloc_size/2
- alloc_size is 32
- Change to max_objects/2
- max_objects is 20,000
- This helps even out bursts of file creations
- Rather than a static number, it is based on load, with a max of 10k


Raise Overall Performance

Conclusions

- Improvement is **about 1.5%,** independent of filesystem layout (files per directory, etc.)
- Simple, one-line change
- No effect to current systems

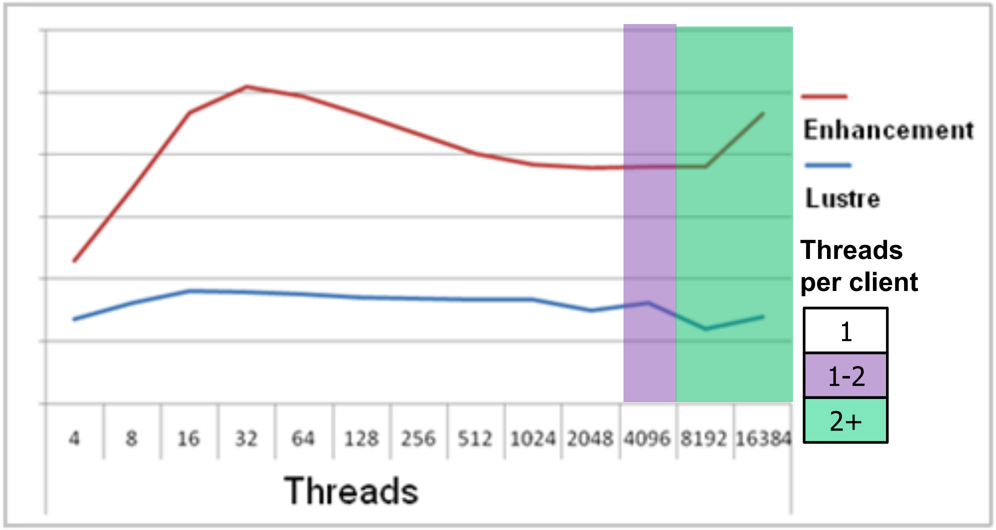
Directory Locking:

- Directory locks during file creation are required to ensure a consistent filesystem
- Locks are held for too long during periods which they are not needed
  - Minimizing critical sections allows for more parallelization

Directory Locking

- During File create, each thread locks the directory in which it’s creating a file
- After file is created, the dir lock is returned to the client, who releases it
- This is slow, especially when multiple threads are creating files in the same directory

Raise Overall Performance



Conclusions

- Release lock after directory-critical operations are complete
- Performance increase depends on filesystem usage
- Multiple clients creating files in the same directory sees greatly improved performance, **up to 340%**
- One client in its own directory, increase is negligible

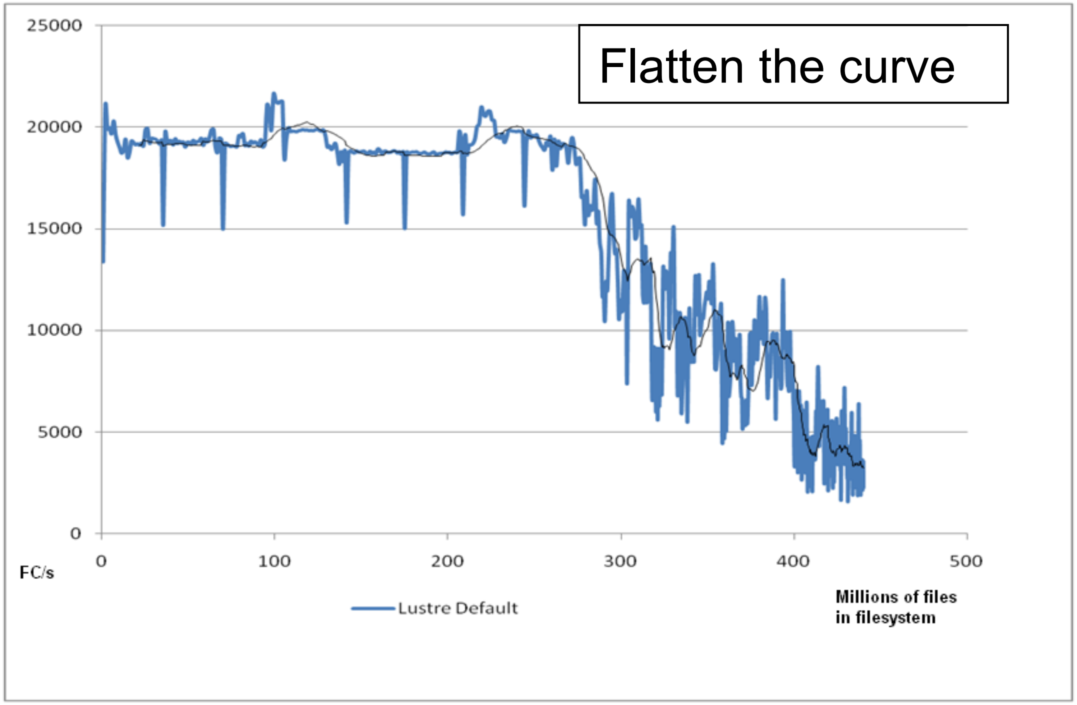
Flattening the Curve:

- Metadata performance is affected by total system performance
  - The system only moves as fast as its slowest part
- At a certain point OSTs become the bottleneck

OST Allocation Change

- Lustre by default uses 32 directories per OST to store objects

- This is fine for awhile but…



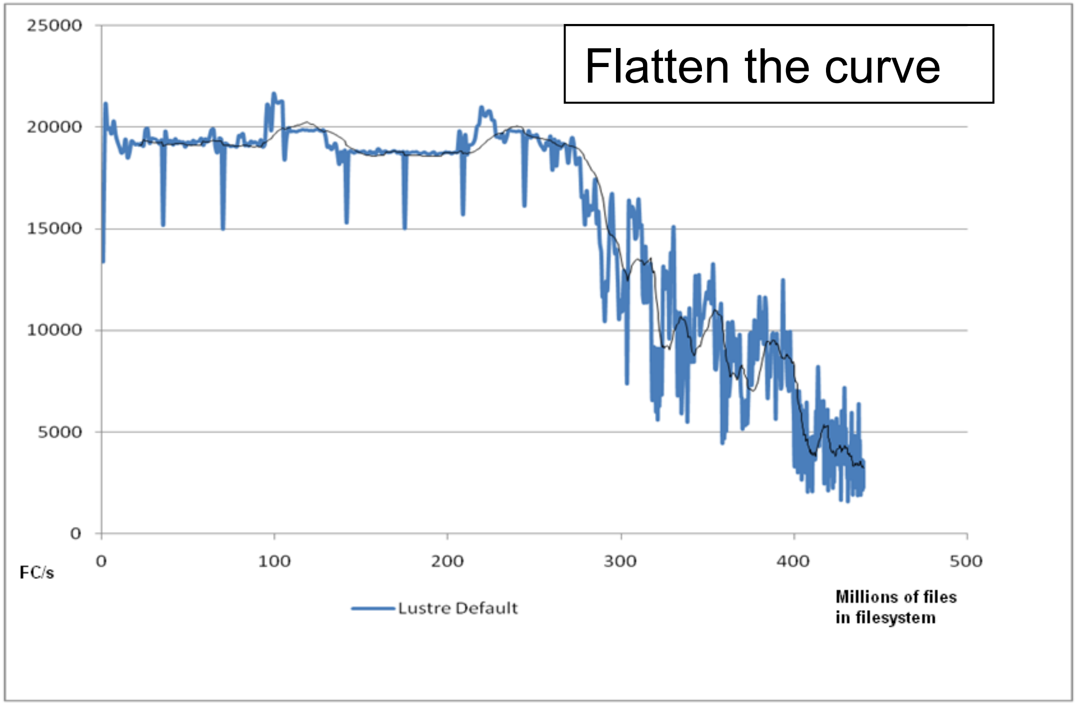
OST Allocation Change

- Lustre by default uses 32 directories per OST to store objects

- This is fine for awhile but…

the more files in the filesystem the slower file creates are

- Due to directory cache thrashing, bad locality, longer searches, ext overhead, etc.

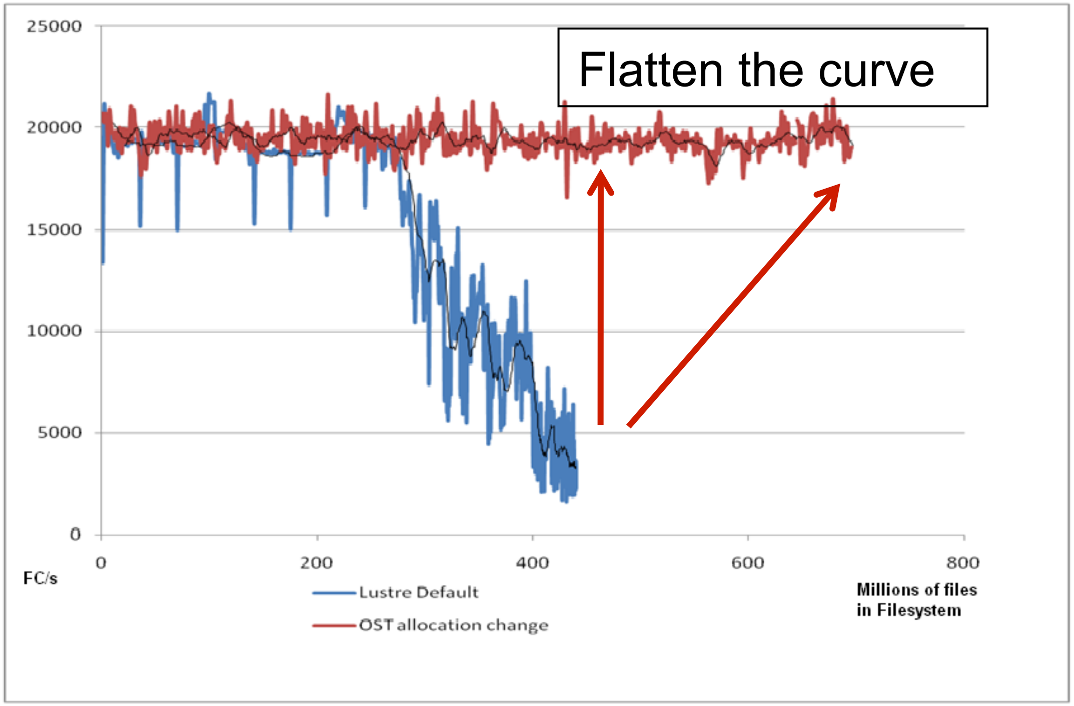

Flatten the curve

OST Allocation Change

- Lustre by default uses 32 directories per OST to store objects
- This is fine for awhile but…

the more files in the filesystem the slower file creates are

- Changing OST allocation algorithms is the solution
- Change From: node_num%32
- To: (node_num/65536)%4096


Flatten the curve

Conclusions

- All OST directories become quite small
- New allocations have good locality, performance is constant

Conclusions:

- Significant performance increase from 2 distinct areas
- Simple, straightforward patches will be available from Terascala Website (www.terascala.com)
  - Use signup sheet at Terascala table to get notified