SDC
STORAGE DEVELOPER CONFERENCE
SNIA SANTA CLARA, 2017

File Systems Fated for Senescence? Nonsense, Says Science!

Ainesh Bakshi
Martin Farach-Colton
Rutgers University
Yizheng Jiao
Yang Zhan
Donald E. Porter
UNC Chapel Hill

Alex Conway
Rutgers University

Michael A. Bender
William Jannen
Rob Johnson
Stony Brook University
Jun Yuan
SUNY Farmingdale
Bradley C. Kuszmaul
MIT and Oracle Corporation

SDC

STORAGE DEVELOPER CONFERENCE

SNIA SANTA CLARA, 2017

File Systems Fated for Senescence? Nonsense, Says Science; The Essence of Semperjuvenescense is Coalescence!

SDC
STORAGE DEVELOPER CONFERENCE
SNIA SANTA CLARA, 2017

Old age

File Systems Fated for Senescence? Nonsense, Says Science; The Essence of Semperjuvenescense is Coalescence!

Merging together

Being forever young

What is file system aging?

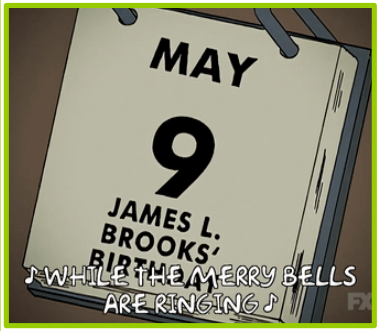
- Aging is fragmentation over time

What is file system aging?

- Aging is fragmentation over time

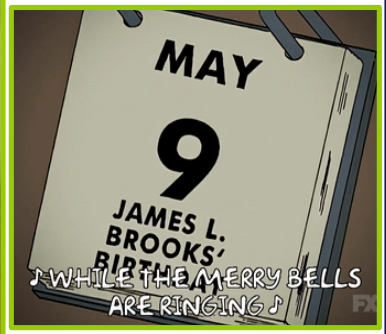
What is file system aging?

- Aging is fragmentation over time



What is file system aging?

- Aging is fragmentation over time

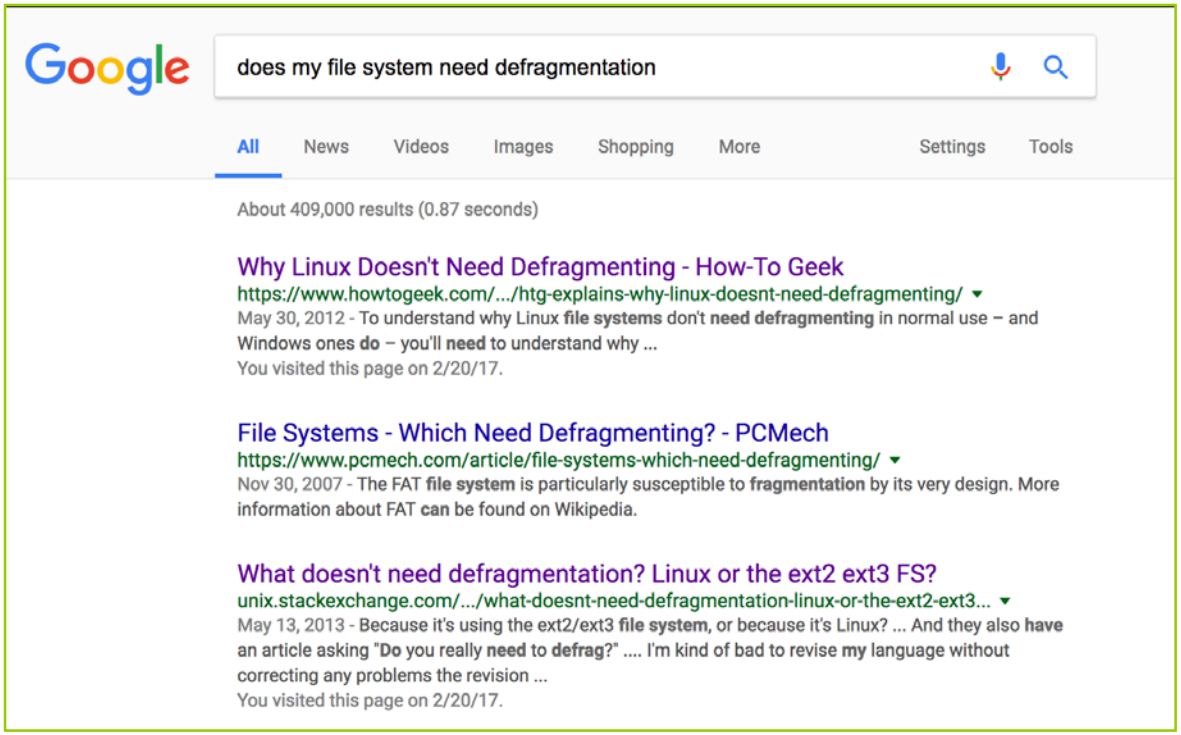


In this talk

- Do file systems age?
  - How can we reproduce aging?
  - How can we measure aging?
- How can we stop aging?

Is aging a problem?

Is aging a problem?



Is aging a problem?

Chris Hoffman at howtogeek.com says:

"Linux's ext2, ext3, and ext4 file systems… [are] designed to avoid fragmentation in normal use."

"If you do have problems with fragmentation on Linux, you probably need a larger hard disk."

Is aging a problem?

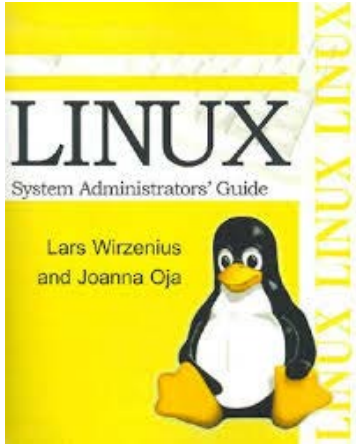
I'm Feeling Lucky

Chris Hoffman at howtogeek.com says:

"Linux's ext2, ext3, and ext4 file systems… [are] designed to avoid fragmentation in normal use."

"If you do have problems with fragmentation on Linux, you probably need a larger hard disk."

"Modern Linux filesystems keep fragmentation at a minimum…Therefore it is not necessary to worry about fragmentation in a Linux system."



Is aging a problem?

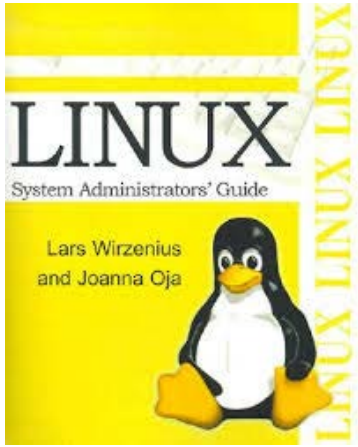
I'm Feeling Lucky

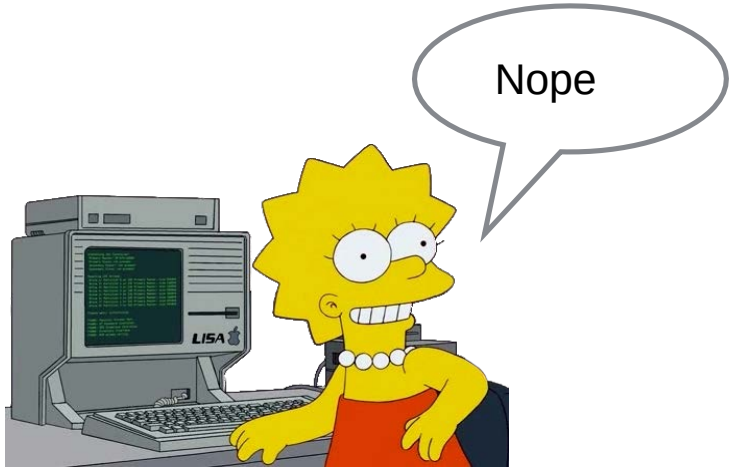
Chris Hoffman at howtogeek.com says:

"Linux's ext2, ext3, and ext4 file systems… [are] designed to avoid fragmentation in normal use."

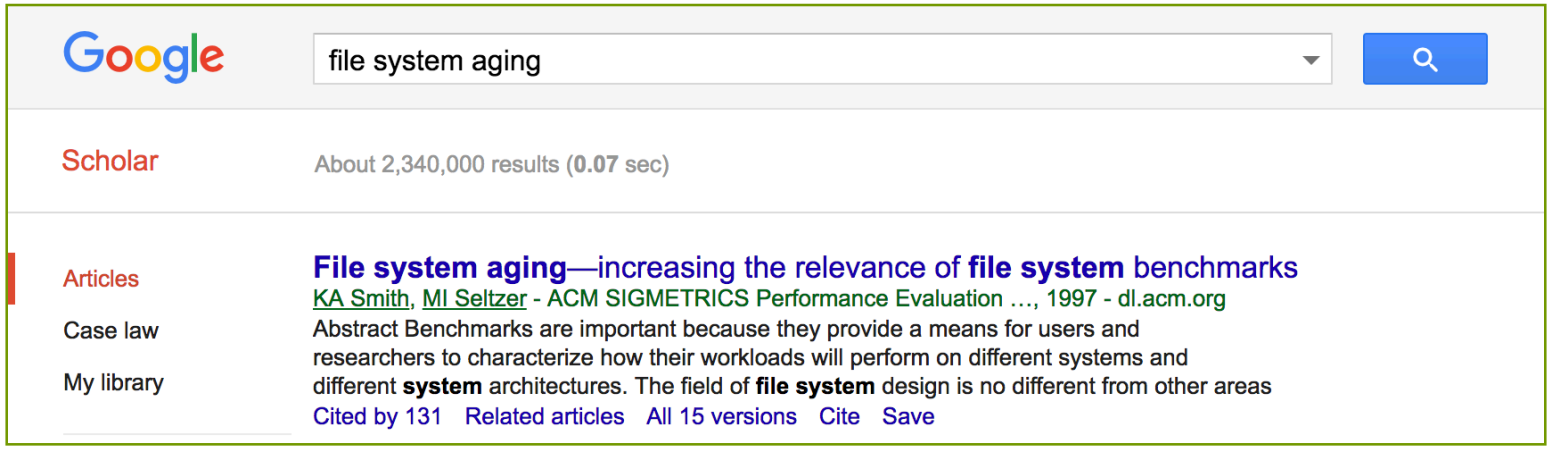
"If you do have problems with fragmentation on Linux, you probably need a larger hard disk."

"Modern Linux filesystems keep fragmentation at a minimum…Therefore it is not necessary to worry about fragmentation in a Linux system."





Is aging a problem?

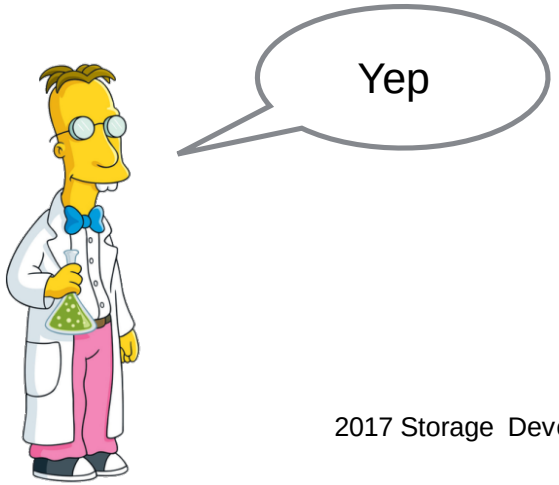


Is aging a problem?

- Aging happens in real file systems
  - Smith and Seltzer (‘97)
- Benchmarks should incorporate aging
  - Zhu, Chen and Chiueh (‘05)
  - Agrawal, A. Arpaci-Dusseau and R. Arpaci-Dusseau (‘09)

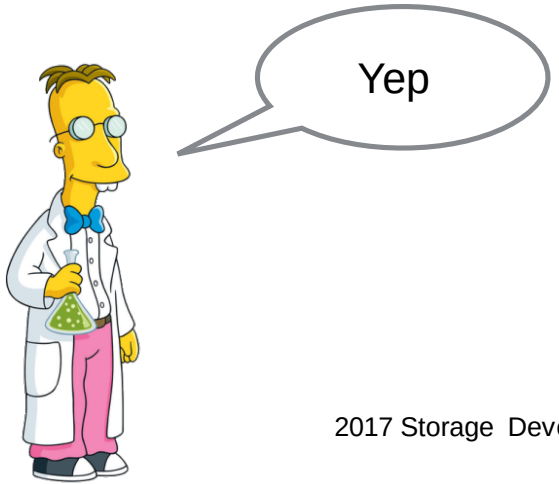
Is aging a problem?

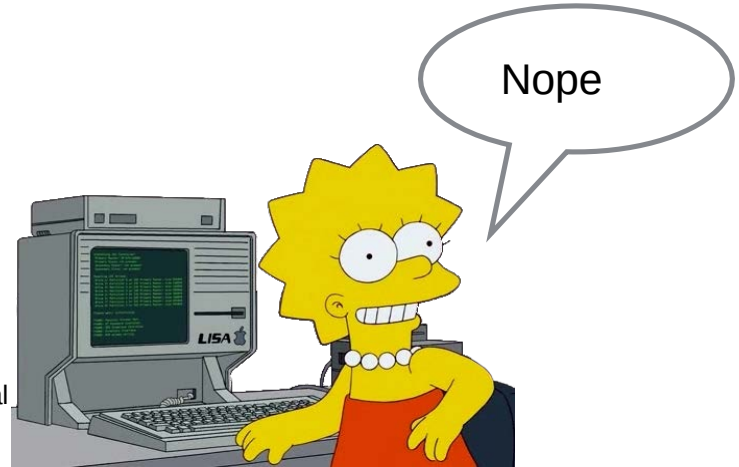
- Aging happens in real file systems
  - Smith and Seltzer (‘97)
- Benchmarks should incorporate aging
  - Zhu, Chen and Chiueh (‘05)
  - Agrawal, A. Arpaci-Dusseau and R. Arpaci-Dusseau (‘09)



Is aging a problem?

Is aging a problem?





Let’s do some science!

Inducing Aging

- We use three different workloads:
  - Developer workload
  - Server workload
  - Synthetic workloads

Inducing Aging

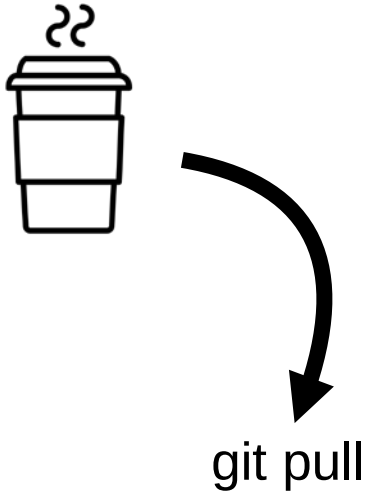
- We use three different workloads:
  - Developer workload
  - Server workload
  - Synthetic workloads

} See the paper

Simulating a developer

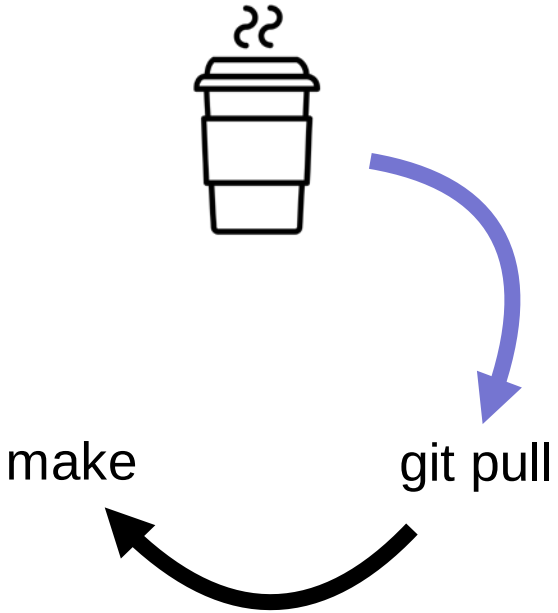
get coffee

Simulating a developer



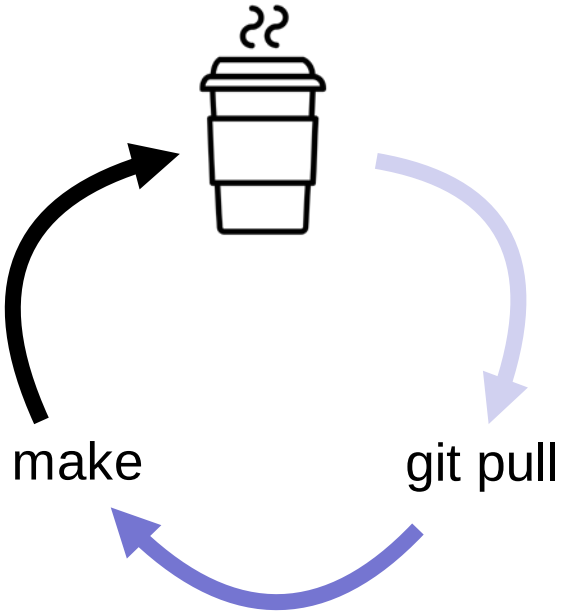
get coffee
git pull

Simulating a developer



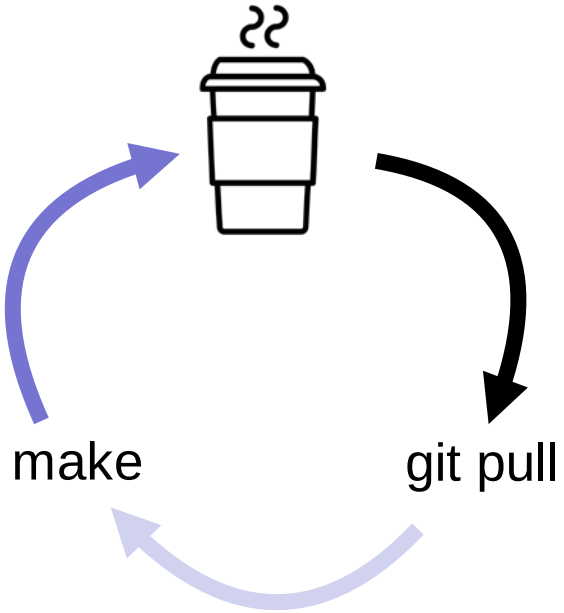
get coffee
git pull
make

Simulating a developer



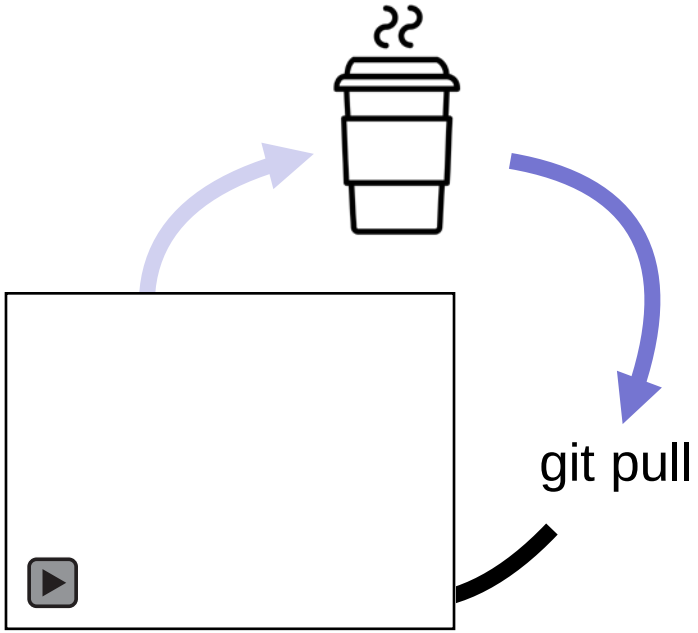
get coffee
git pull
make
get coffee

Simulating a developer



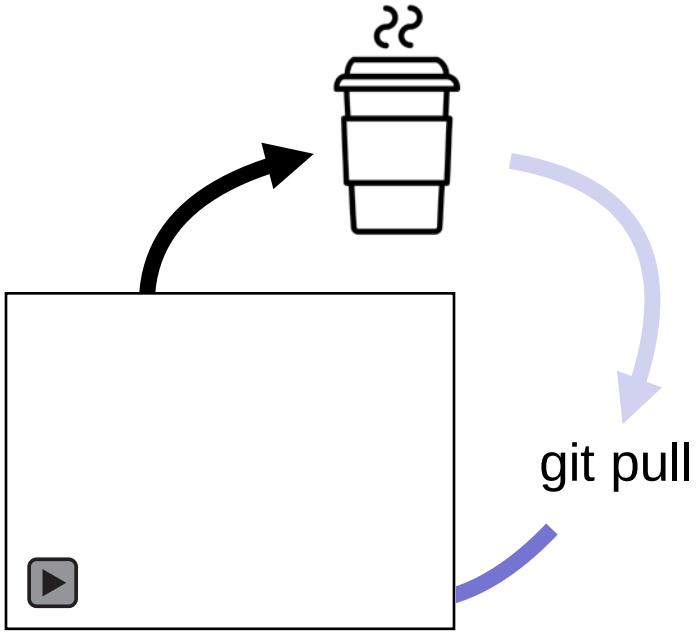
get coffee
git pull
make
get coffee
git pull

Simulating a developer



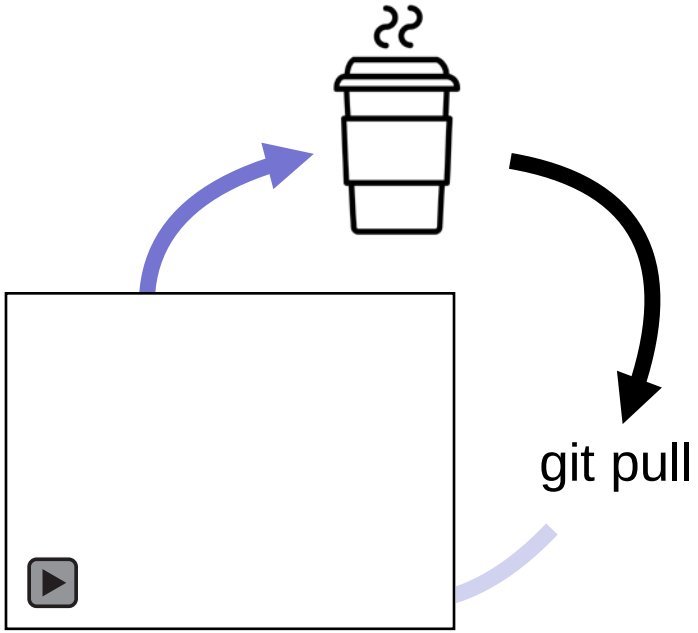
get coffee
git pull
make
get coffee
git pull
add awesome features

Simulating a developer



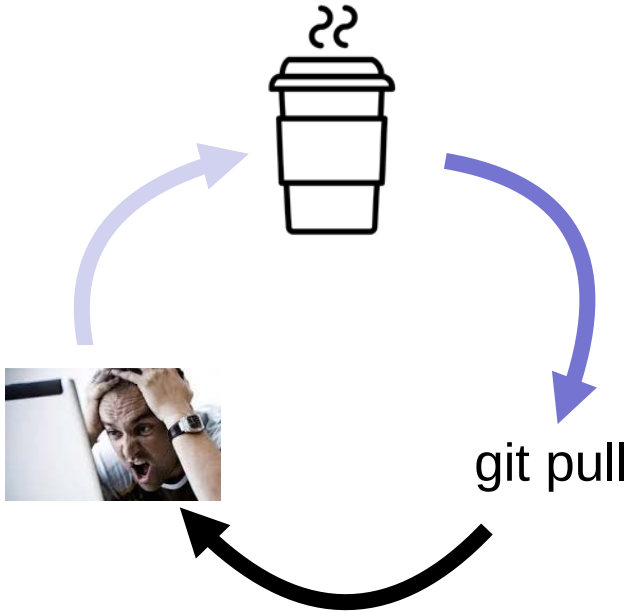
get coffee
git pull
make
get coffee
git pull
add awesome features
get coffee

Simulating a developer



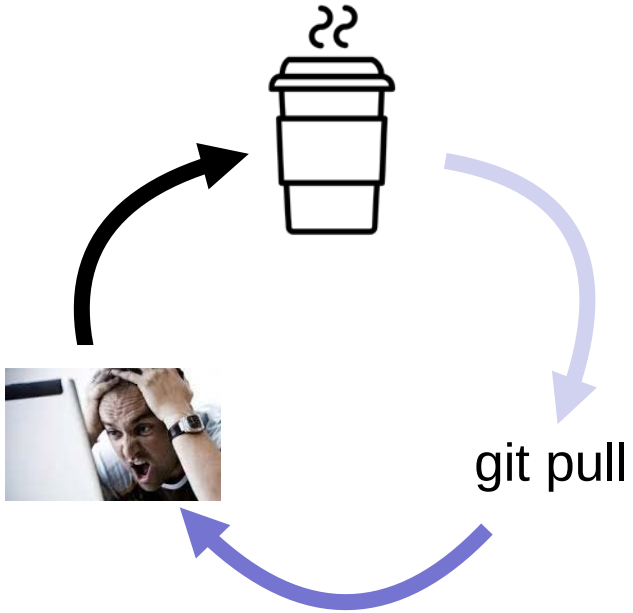
get coffee
git pull
make
get coffee
git pull
add awesome features
get coffee
git pull

Simulating a developer



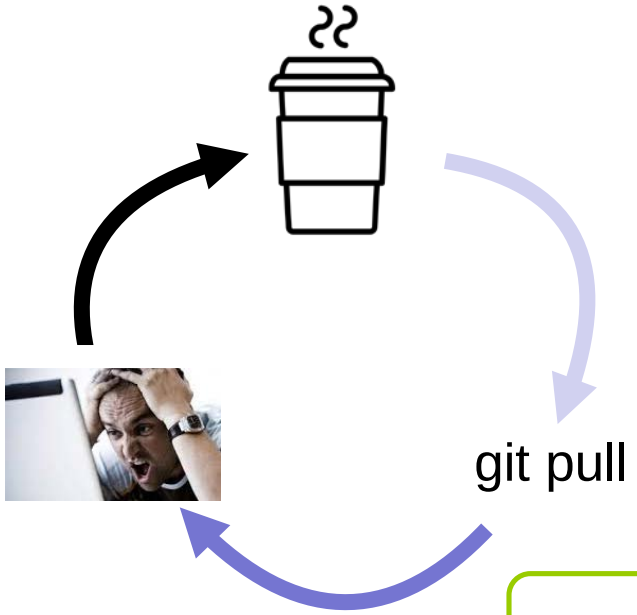
get coffee
git pull
make
get coffee
git pull
add awesome features
get coffee
git pull
fix bugs

Simulating a developer



get coffee
git pull
make
get coffee
git pull
add awesome features
get coffee
git pull
fix bugs
…

Simulating a developer



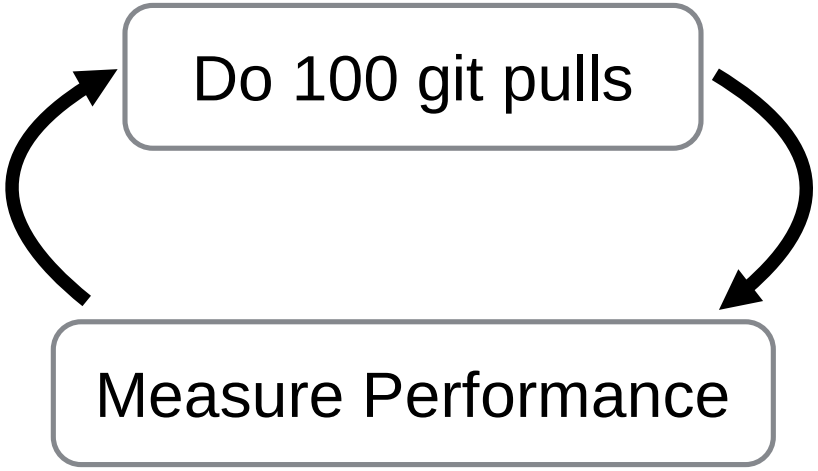
get coffee
git pull
make
get coffee
git pull
add awesome features
get coffee
git pull
fix bugs
…

Simulate a developer by replaying git histories

Simulating a developer

Simulating a developer

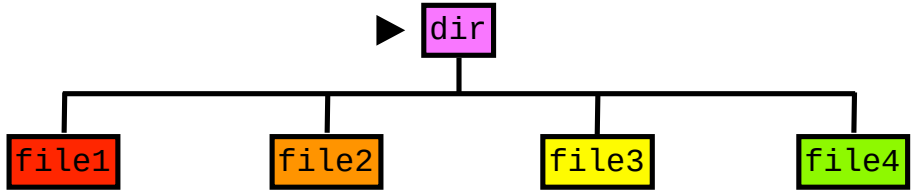
Use the Linux kernel repo from github.com



Measuring aging

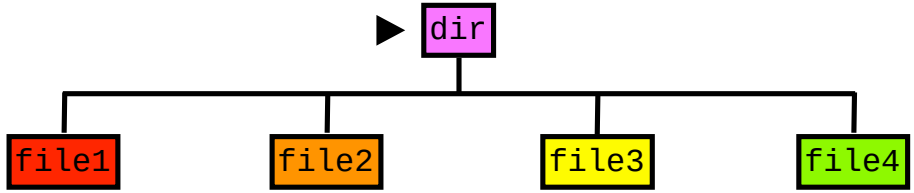
Measuring aging

time grep -r random_string /path/to/filesystem



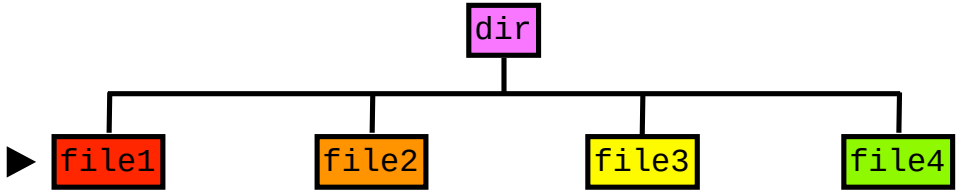
Measuring aging

time grep -r random_string /path/to/filesystem



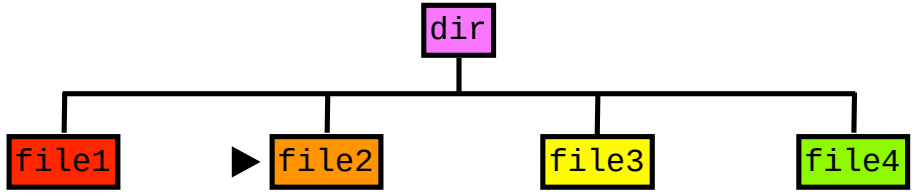
Measuring aging

time grep -r random_string /path/to/filesystem



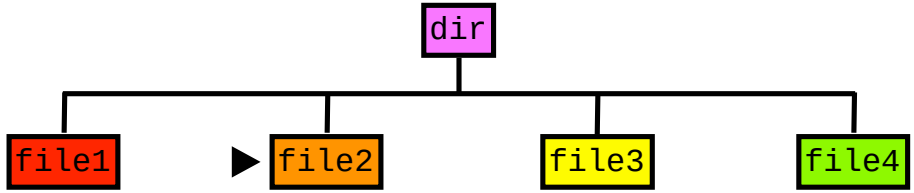
Measuring aging

time grep -r random_string /path/to/filesystem



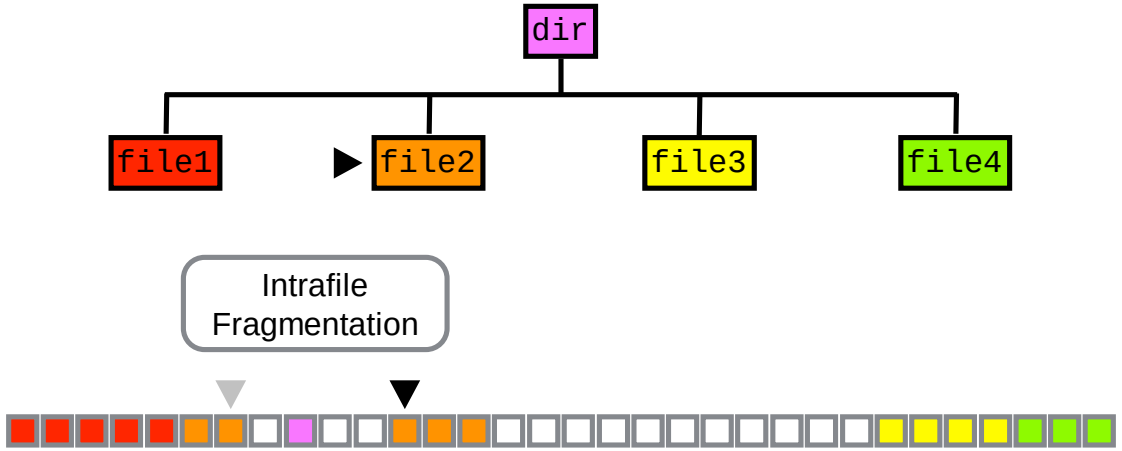
Measuring aging

time grep -r random_string /path/to/filesystem



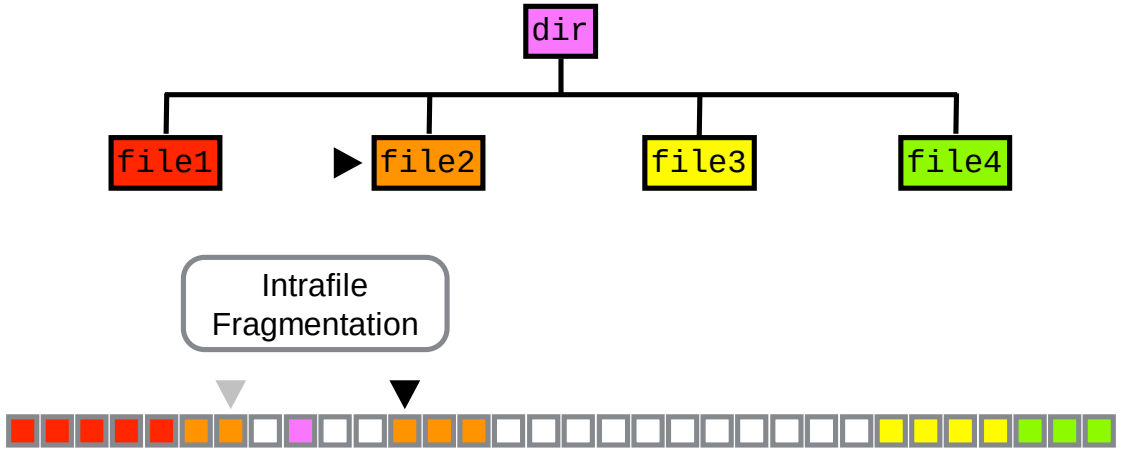
Measuring aging

time grep -r random_string /path/to/filesystem



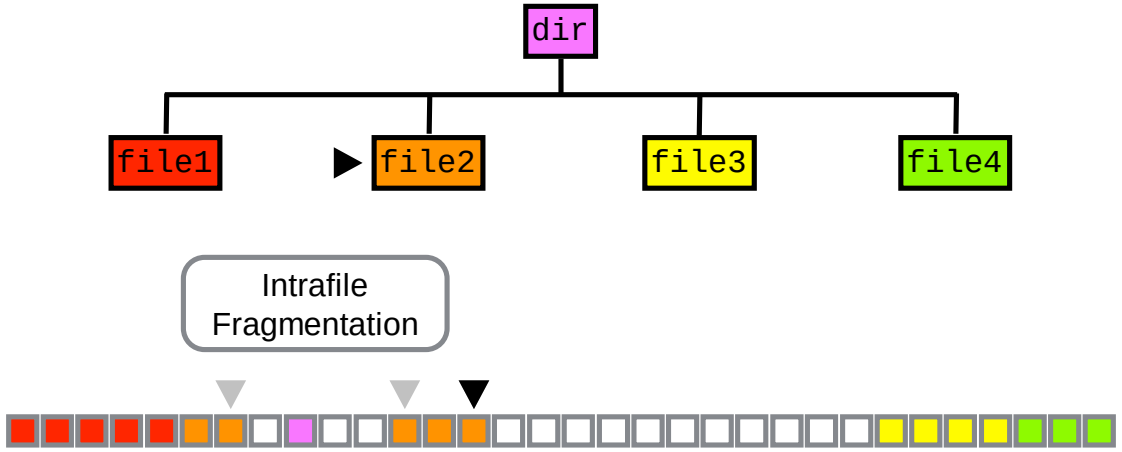
Measuring aging

time grep -r random_string /path/to/filesystem



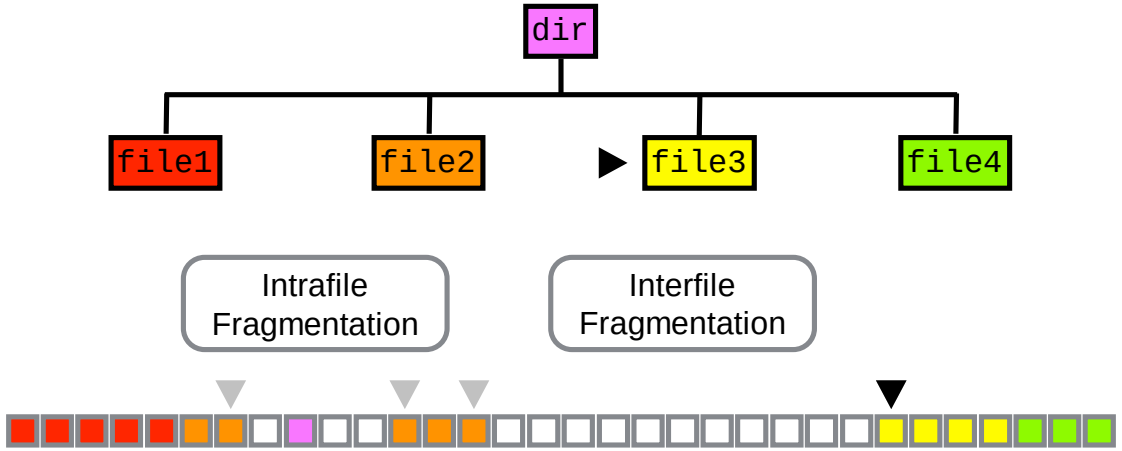
Measuring aging

time grep -r random_string /path/to/filesystem



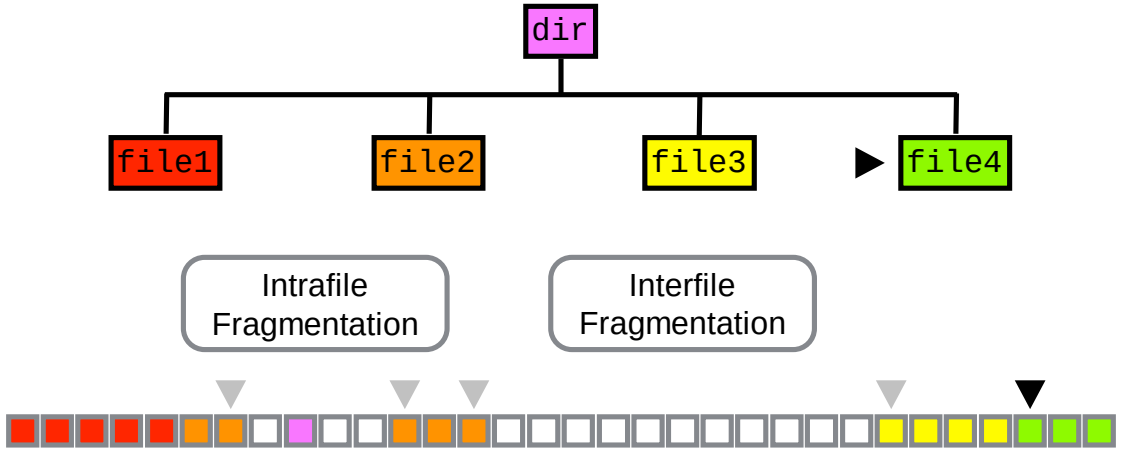
Measuring aging

time grep -r random_string /path/to/filesystem



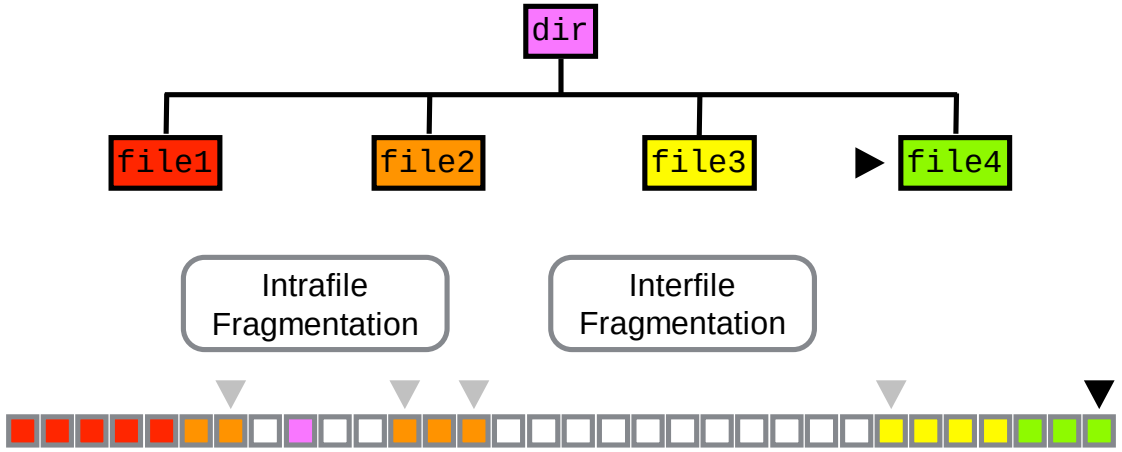
Measuring aging

time grep -r random_string /path/to/filesystem



Measuring aging

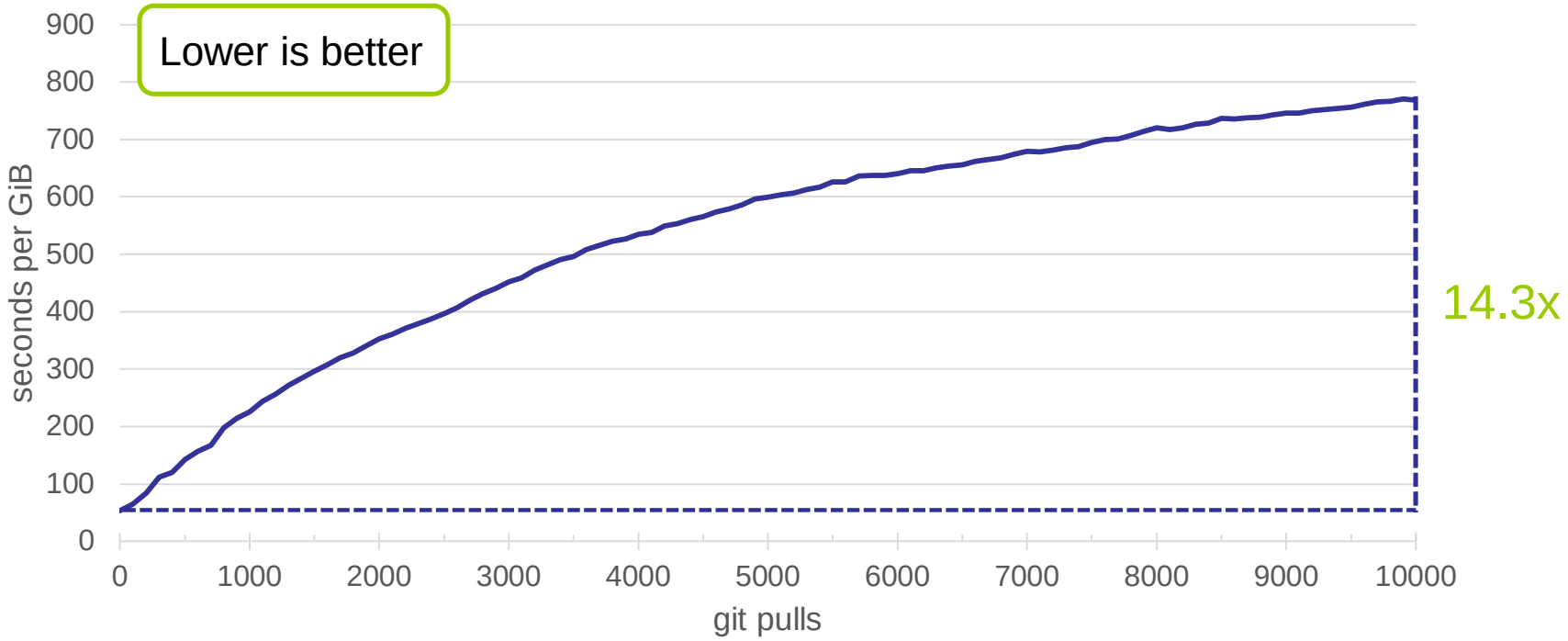
time grep -r random_string /path/to/filesystem



Then normalize per gigabyte read

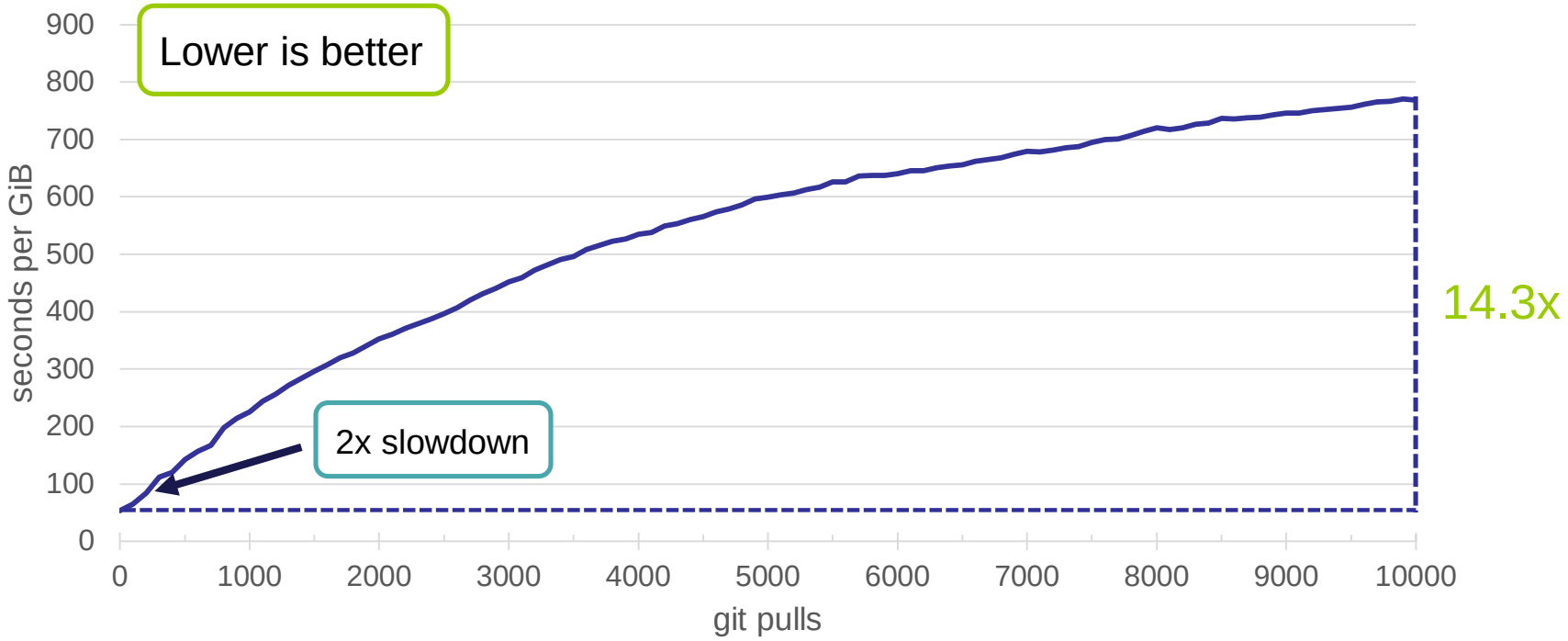
Do modern file systems age?

Git workload – ext4 HDD



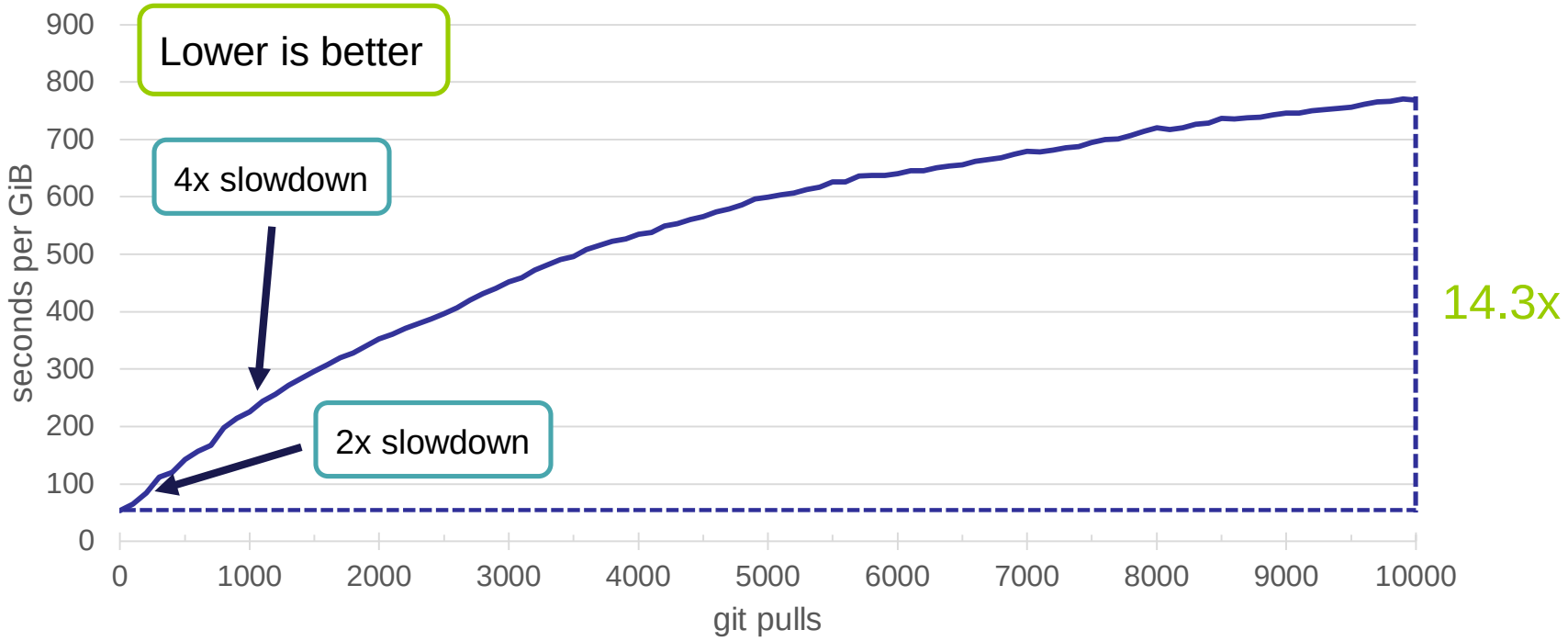
Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Git workload – ext4 HDD



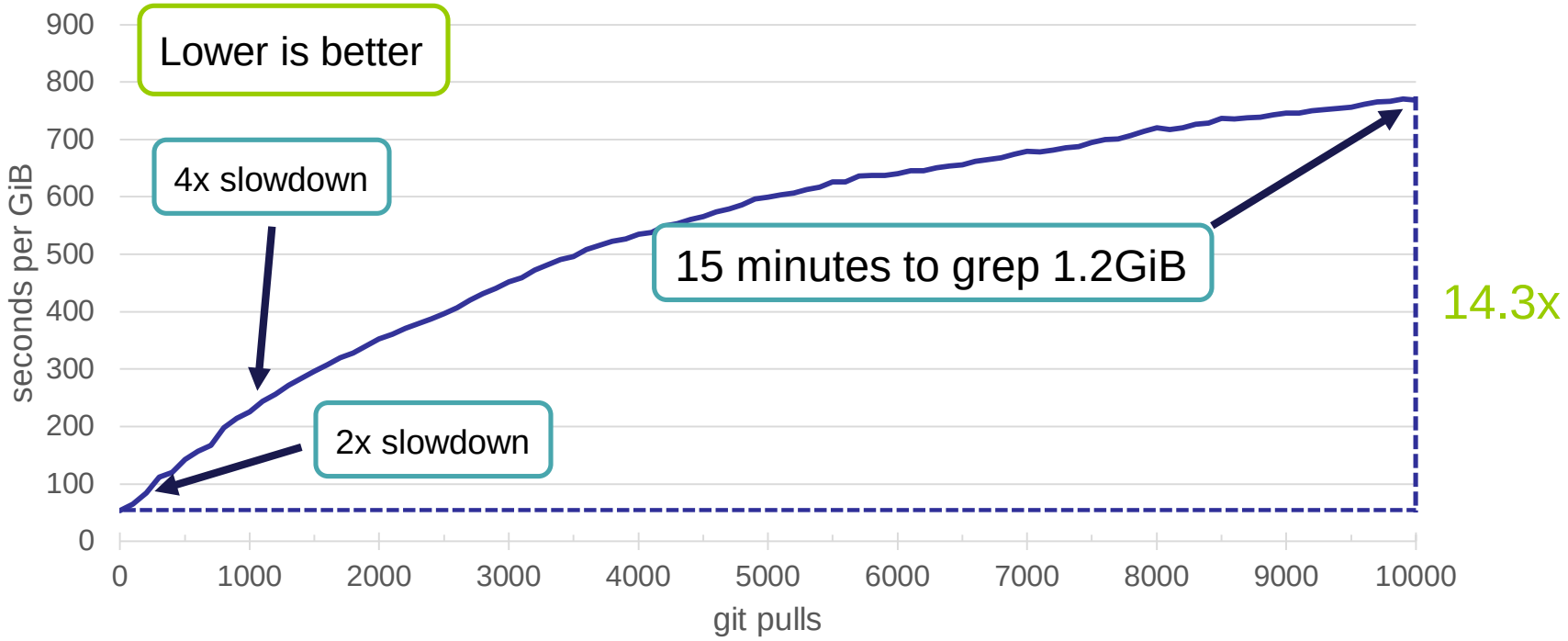
Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Git workload – ext4 HDD



Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Git workload – ext4 HDD



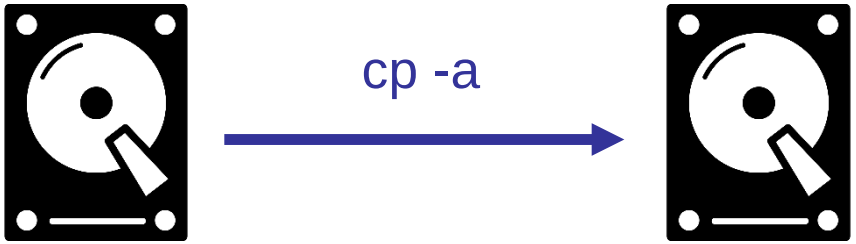
Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

How can we be sure this is aging? Let's rule out some alternatives…

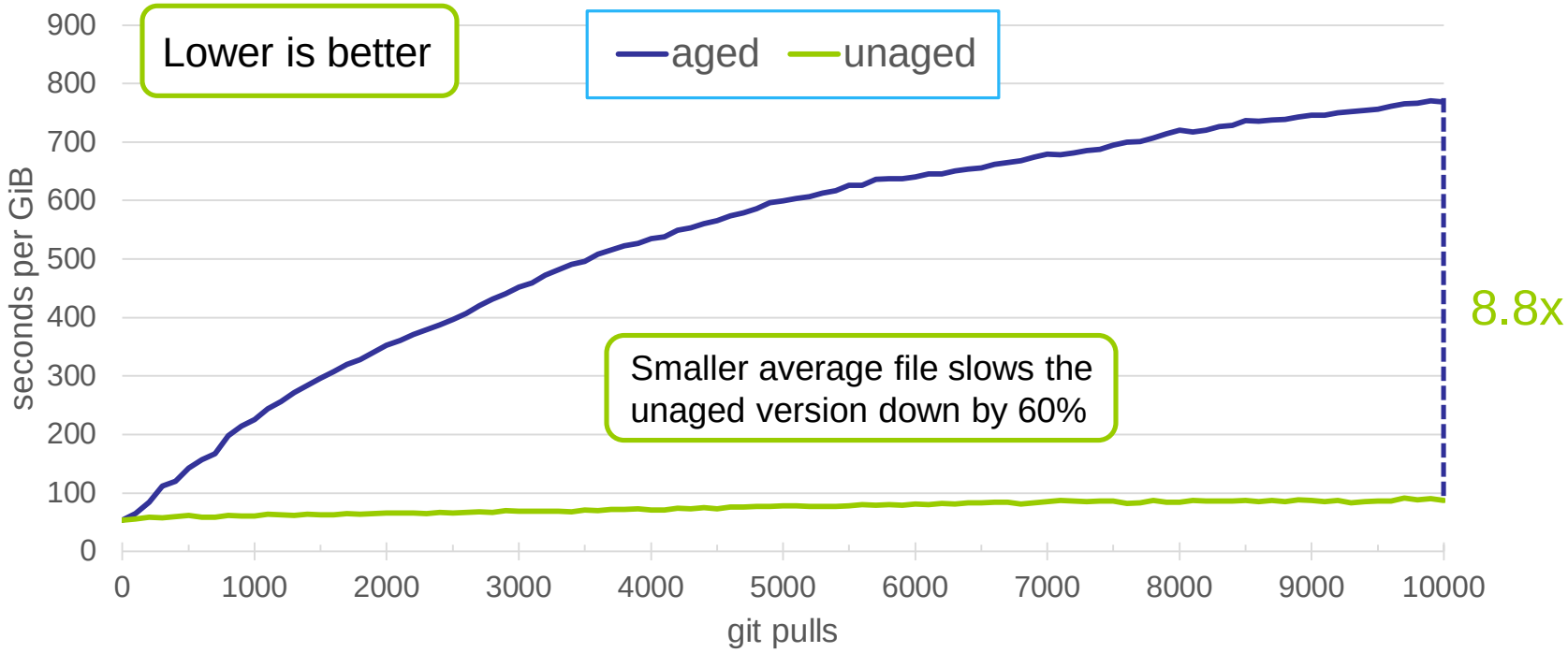
Alternative:
The file system structure changed.

Rejuvenate the file system

- Copy the logical state to newly formatted file system
  - After every 100 git pulls
  - Measure grep cost



Git workload – ext4 HDD

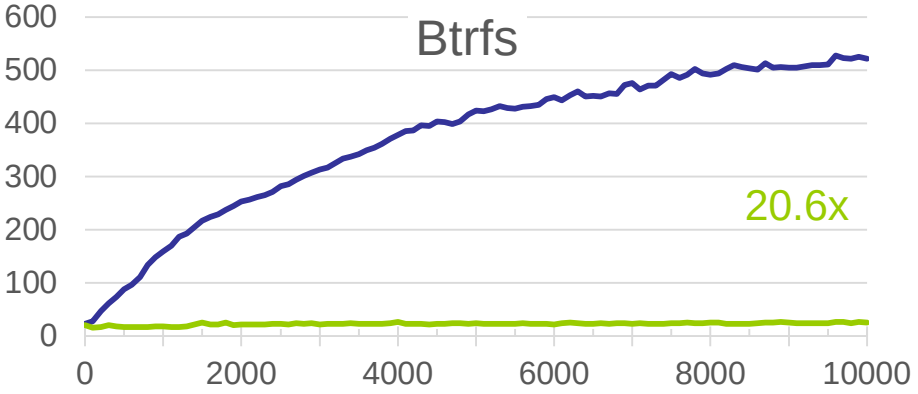


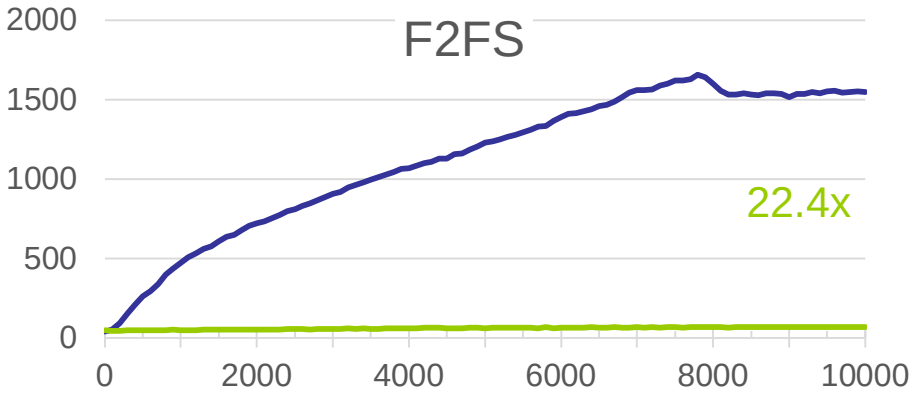
Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

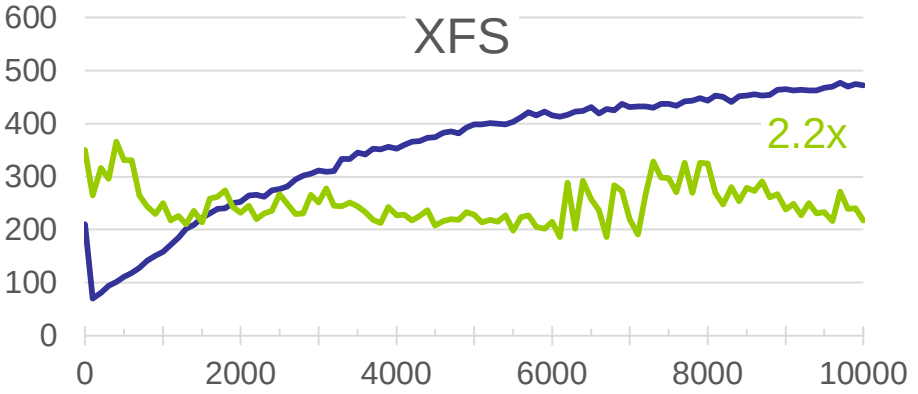
Alternative:
This only happens to ext4.

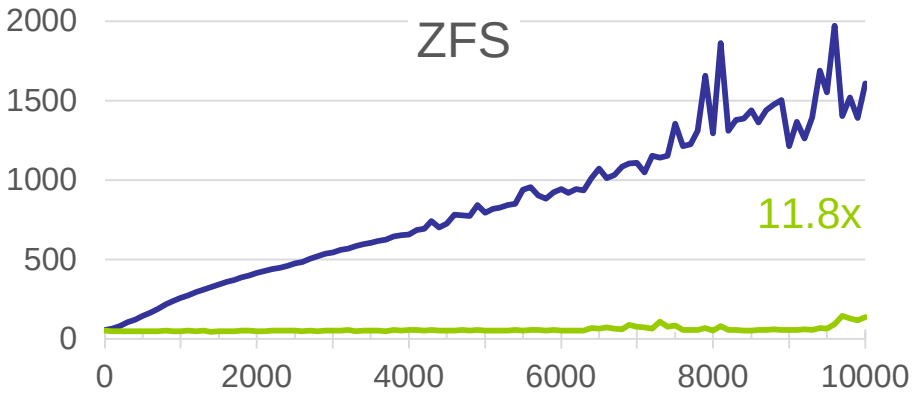
Git workload – HDD

Lower is better



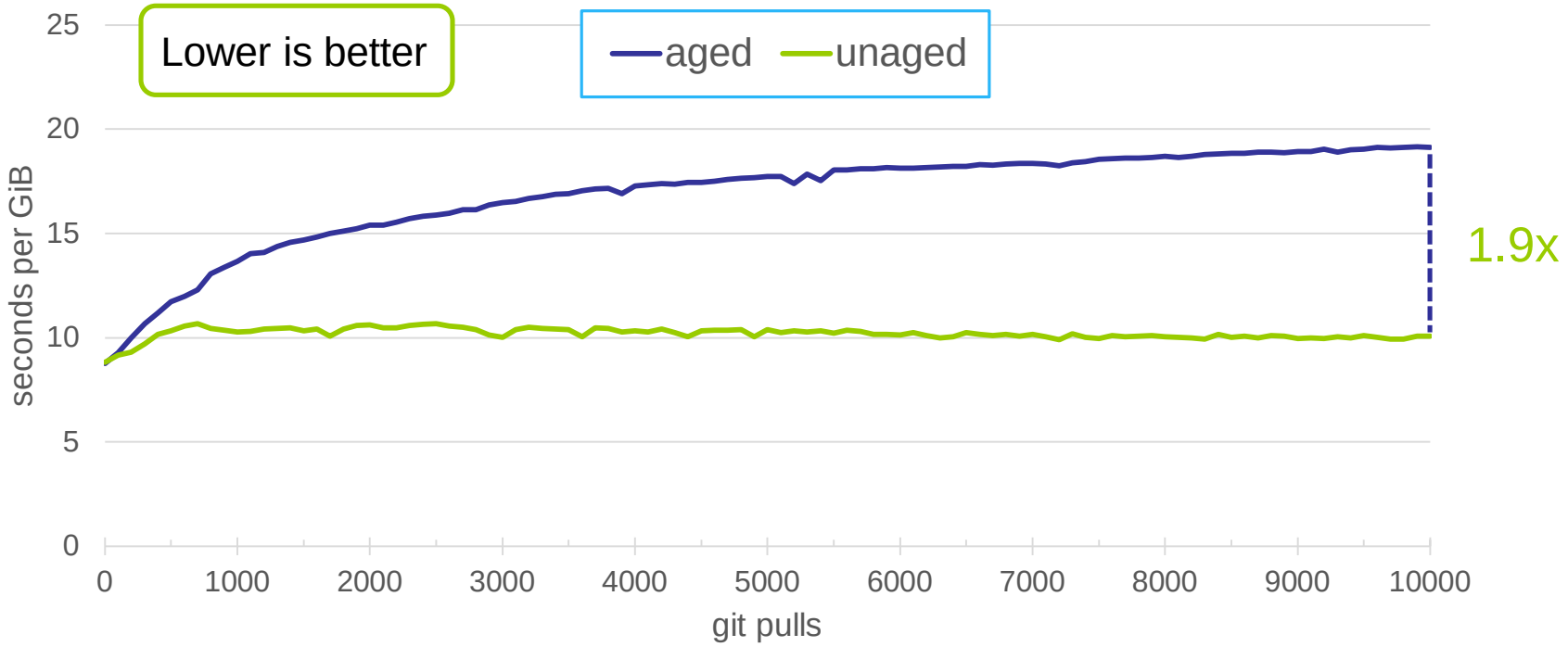






Alternative:
Aging only happens on hard drive.

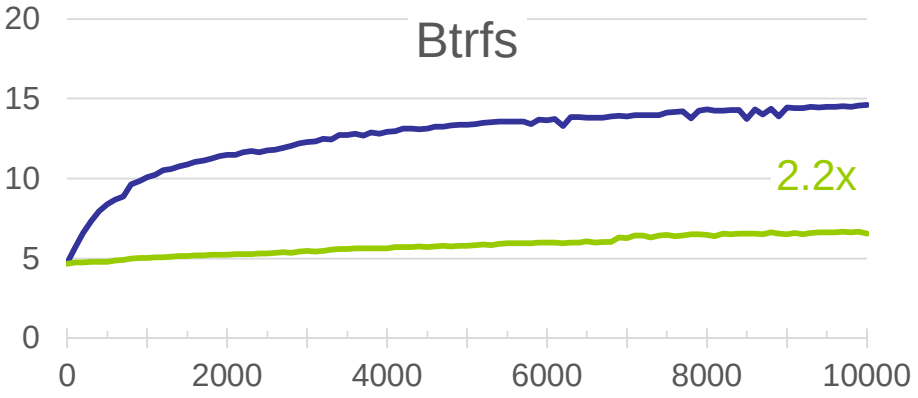
Git workload – XFS SSD

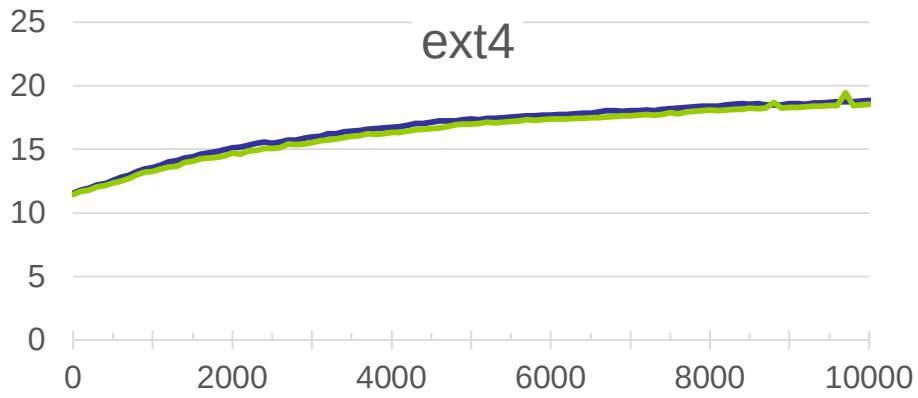


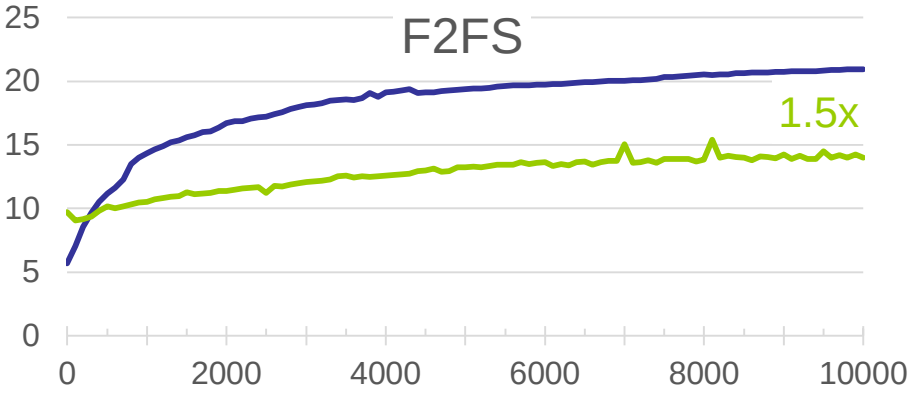
Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB SSD partition – Sandisk Extreme Pro

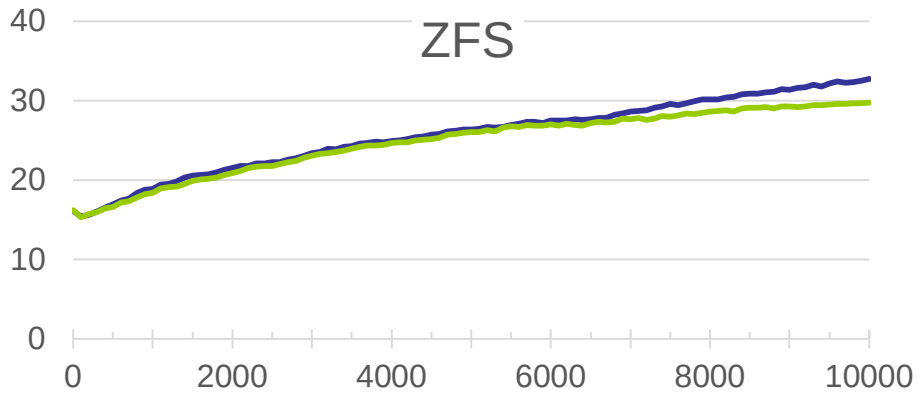
Git workload – SSD

Lower is better



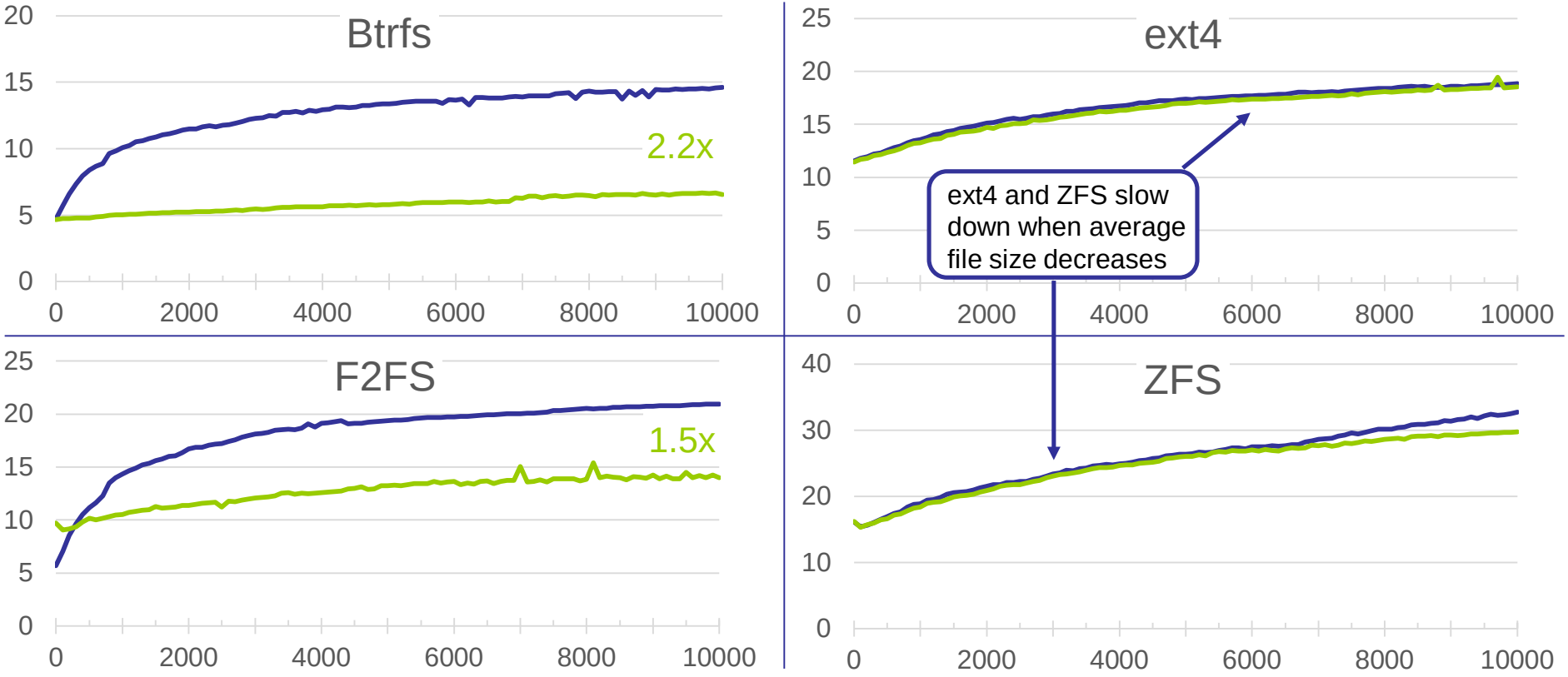






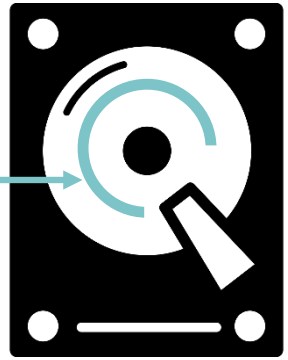
Git workload – SSD

Lower is better



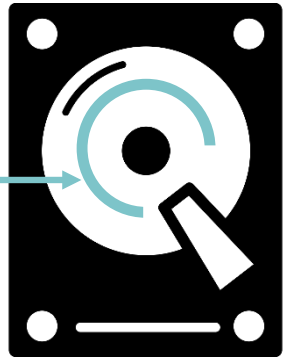
Why do SSDs age?

On hard drive, sequential LBAs are written to nearby physical locations

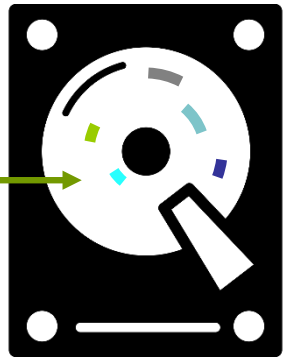


Why do SSDs age?

On hard drive, sequential LBAs are written to nearby physical locations



Non-sequential LBAs may be written to physically separate locations



Why do SSDs age?

On SSDs, physical locality is not an issue.

But the hardware can serve small requests faster than the OS can make them!

“Improving File System Performance of Mobile Storage Systems Using a Decoupled Defragmenter” Hahn et al ATC 2017

Why do SSDs age?

On SSD, requests for logically sequential data are efficient

read(512, 16384)

Non-sequential data generates more requests

read(512, 1024)
read(2024, 512)
read(8704, 512)
read(6656, 1024)
read(11264, 512)

Why do SSDs age?

On SSD, requests for logically sequential data are efficient

read(512, 16384)

Non-sequential data generates more requests

read(512, 1024)
read(2024, 512)
read(8704, 512)
read(6656, 1024)
read(11264, 512)

NVMe is likely to make this worse, not better

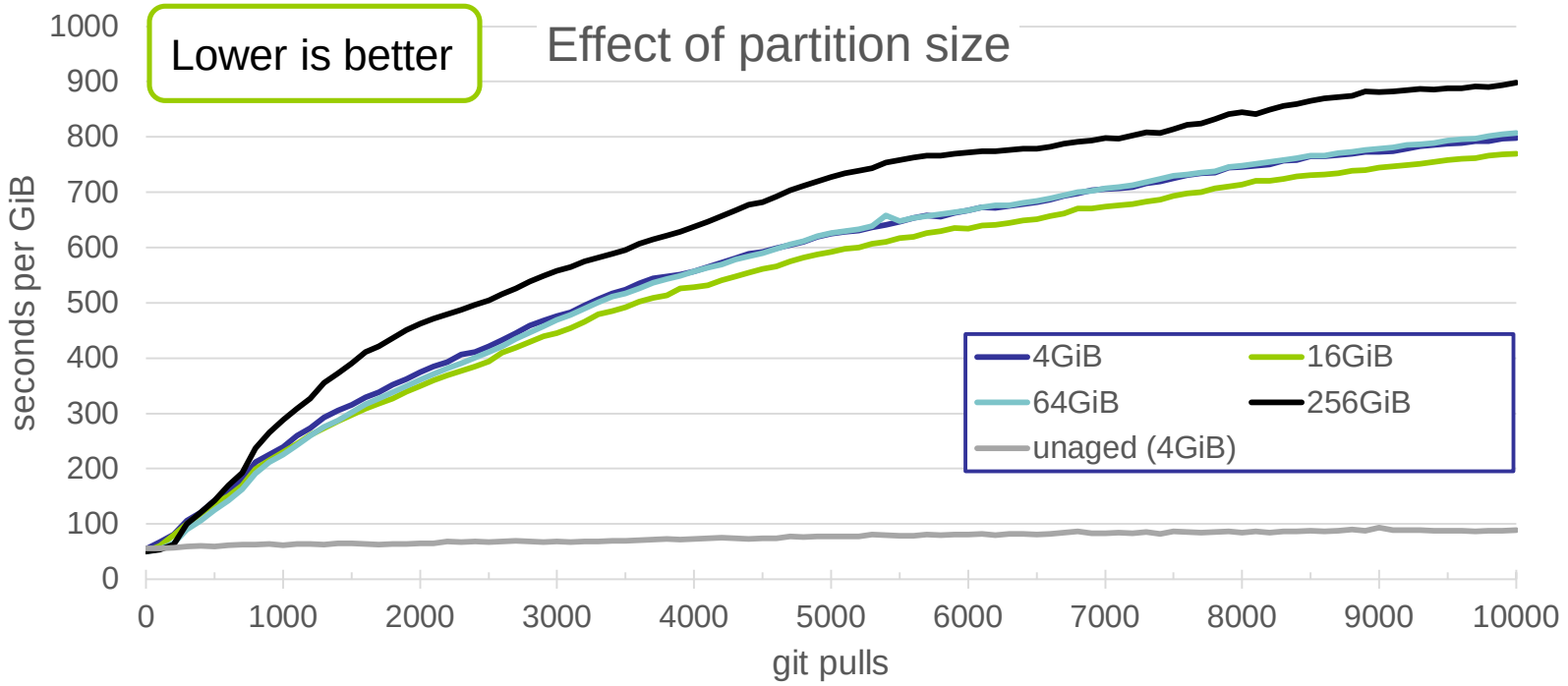
Alternative:
“You probably need a larger hard drive.”

Partition size and aging

- Experiments performed on a 20GB partition.
- Repository size grows from 290MB to 1.2GB.
- The partition was thus at most 6% full.

Aging is not a full disk issue!

Partition size and aging



Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
Variably sized HDD partition - SATA 7200 RPM

Alternative:
This only happens with a cold cache.

What about caching?

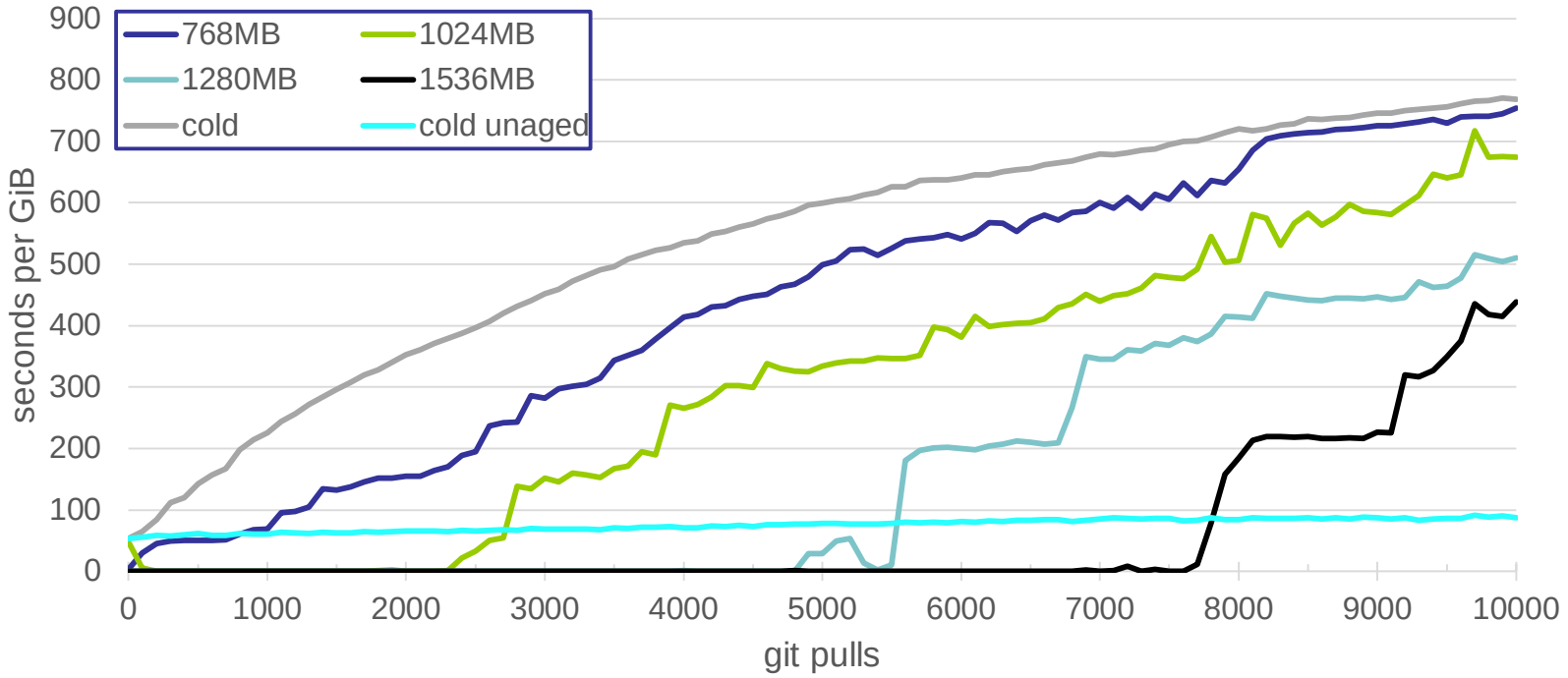
- Write-caching, or deferred allocation, is generally limited to a few seconds in order to maintain reliability. Aging happens on the scale of days or more, so write-caching cannot help.

What about caching?

- Read-caching, or disk cache, can help, but is limited by available memory. When the data set exceeds available memory, aging again becomes a problem.

Git workload – ext4 warm cache

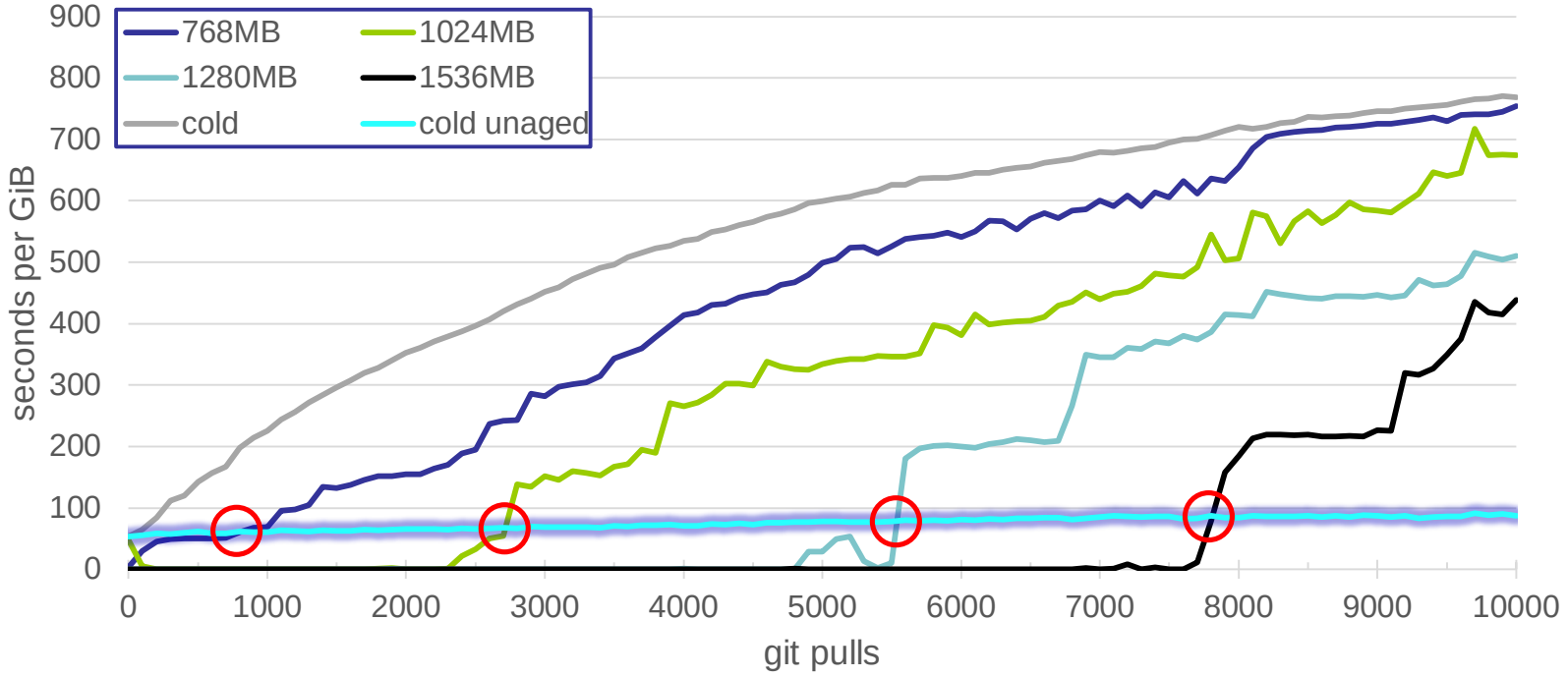
Lower is better



Our Setup: 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Git workload – ext4 warm cache

Lower is better



Note that once the data set doesn't fit in memory, it quickly becomes slower than the unaged version with cold cache

Quick review

- Btrfs, ext4, F2FS, XFS and ZFS all age
  - Up to 22x on HDD
  - Up to 2x on SSD
- Git lets us replay a real development history
  - Induce aging by simulating years of use
  - Takes between 5 hours and 2 days

Quick review

- Versions of the aging scripts used are available at betrfs.org
  - Test your own file system for aging
  - Use them to perform benchmarks on aged file systems

How to prevent aging

How to prevent fragmentation

- Interfile fragmentation: blocks of the same file are not collocated

How to prevent fragmentation

- Interfile fragmentation: blocks of the same file are not collocated
  - Avoid breaking large files into small fragments

How to prevent fragmentation

- Interfile fragmentation: blocks of the same file are not collocated
  - Avoid breaking large files into small fragments
- Intrafile fragmentation: blocks of related files are not collocated

How to prevent fragmentation

- Interfile fragmentation: blocks of the same file are not collocated
  - Avoid breaking large files into small fragments
- Intrafile fragmentation: blocks of related files are not collocated
  - Cluster logically related small files

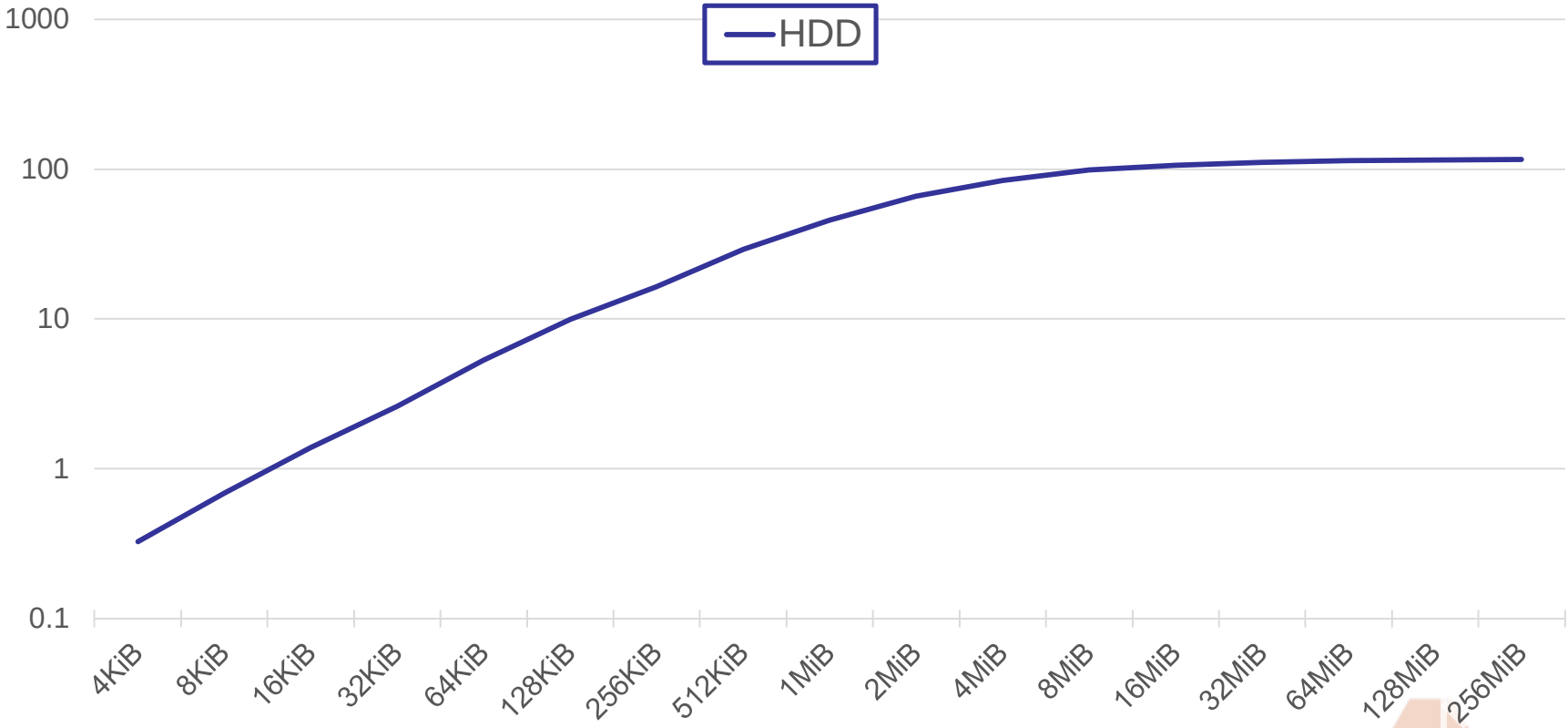
How to prevent fragmentation

- Interfile fragmentation: blocks of the same file are not collocated
  - Avoid breaking large files into small fragments
- Intrafile fragmentation: blocks of related files are not collocated
  - Cluster logically related small files

But what do we mean by small?

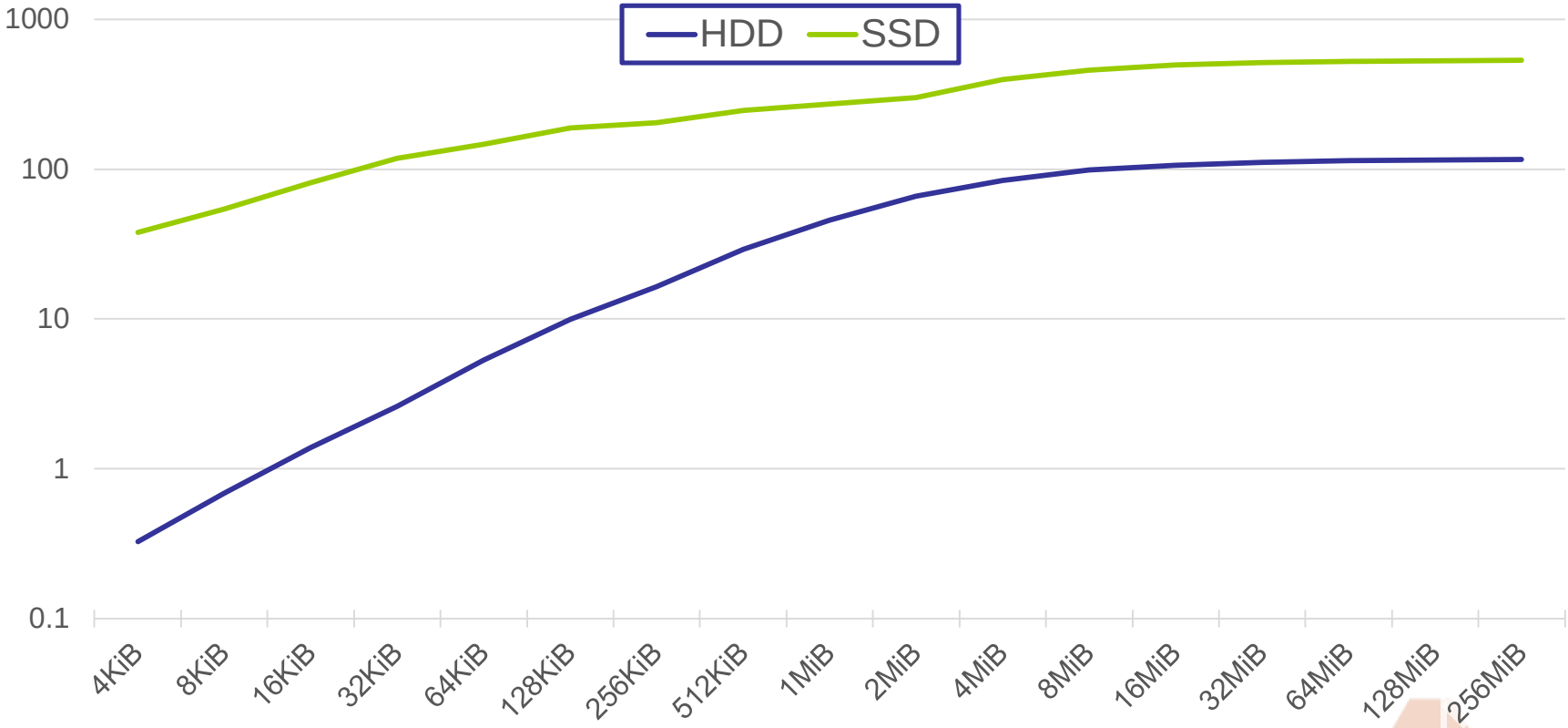
Natural transfer size

Higher is better



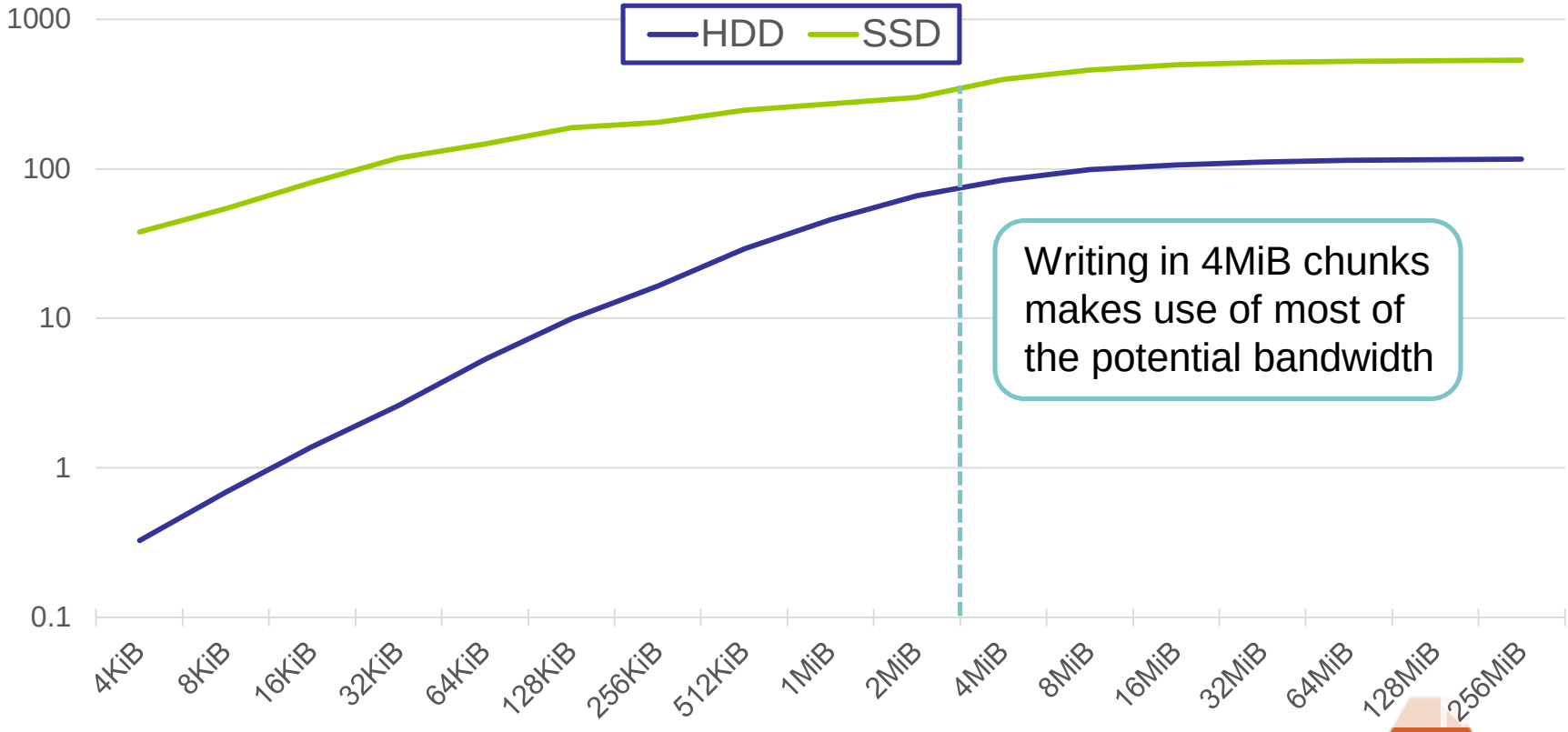
Natural transfer size

Higher is better



Natural transfer size

Higher is better



How to prevent fragmentation

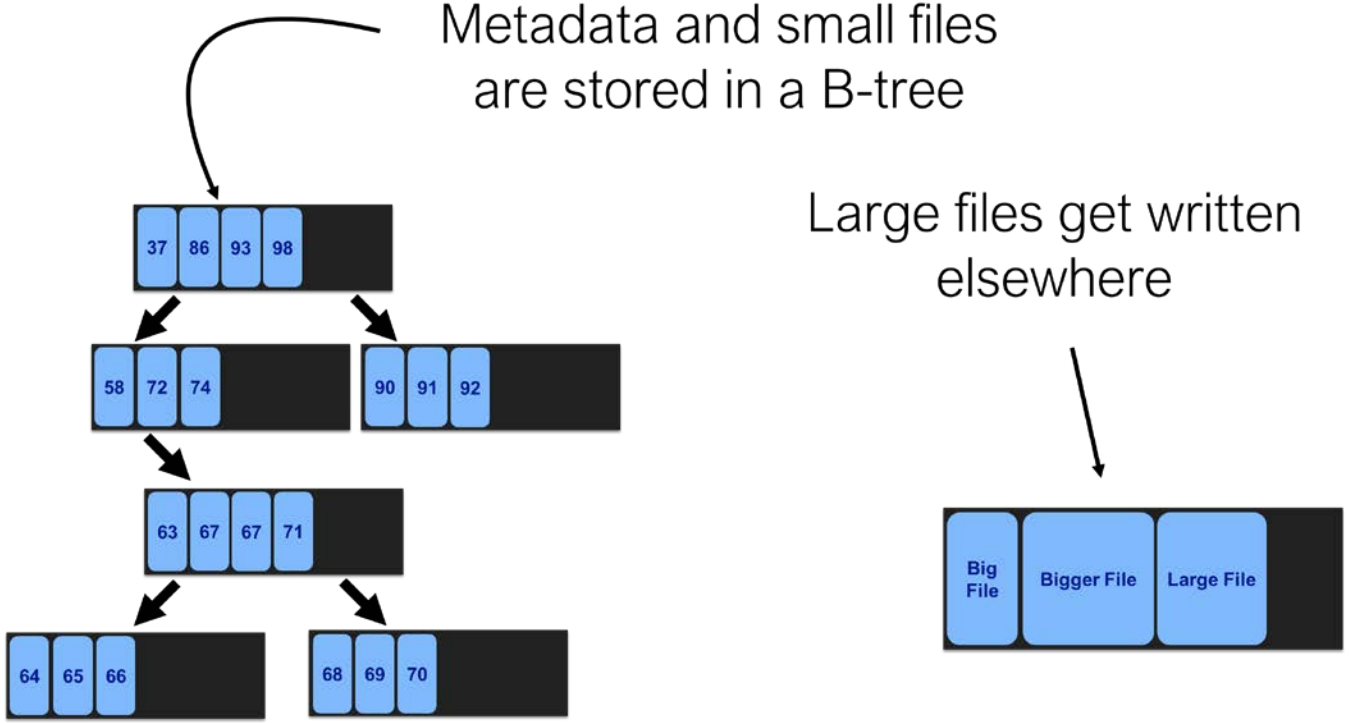
- Interfile fragmentation: blocks of the same file are not collocated
  - Avoid breaking large files into small fragments
- Intrafile fragmentation: blocks of related files are not collocated
  - Cluster logically related small files

Prediction: keeping related data in 4MiB chunks will substantially reduce aging

How to prevent aging (with Btrfs)

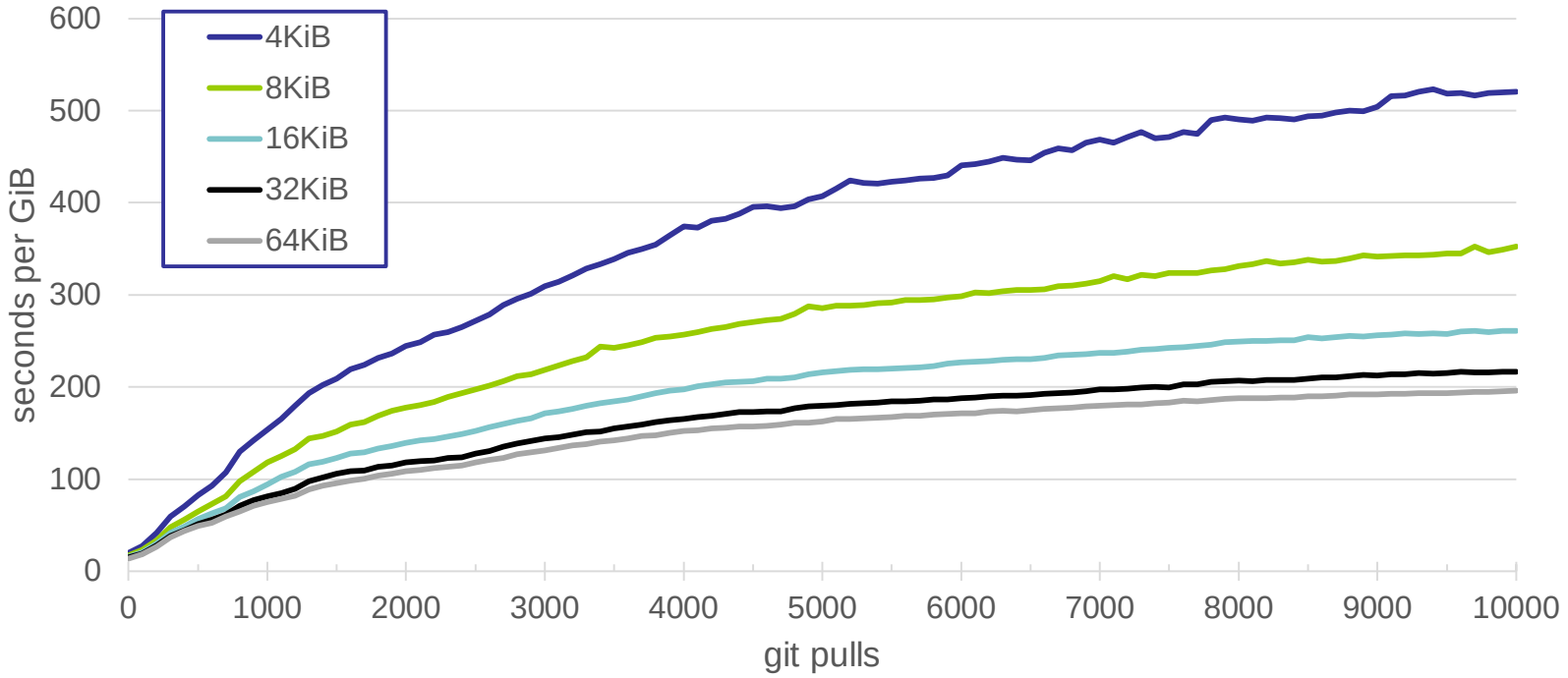
reduce
How to ~~prevent~~ aging (with Btrfs)

How to reduce aging (with Btrfs)



Btrfs: Larger nodes => less aging?

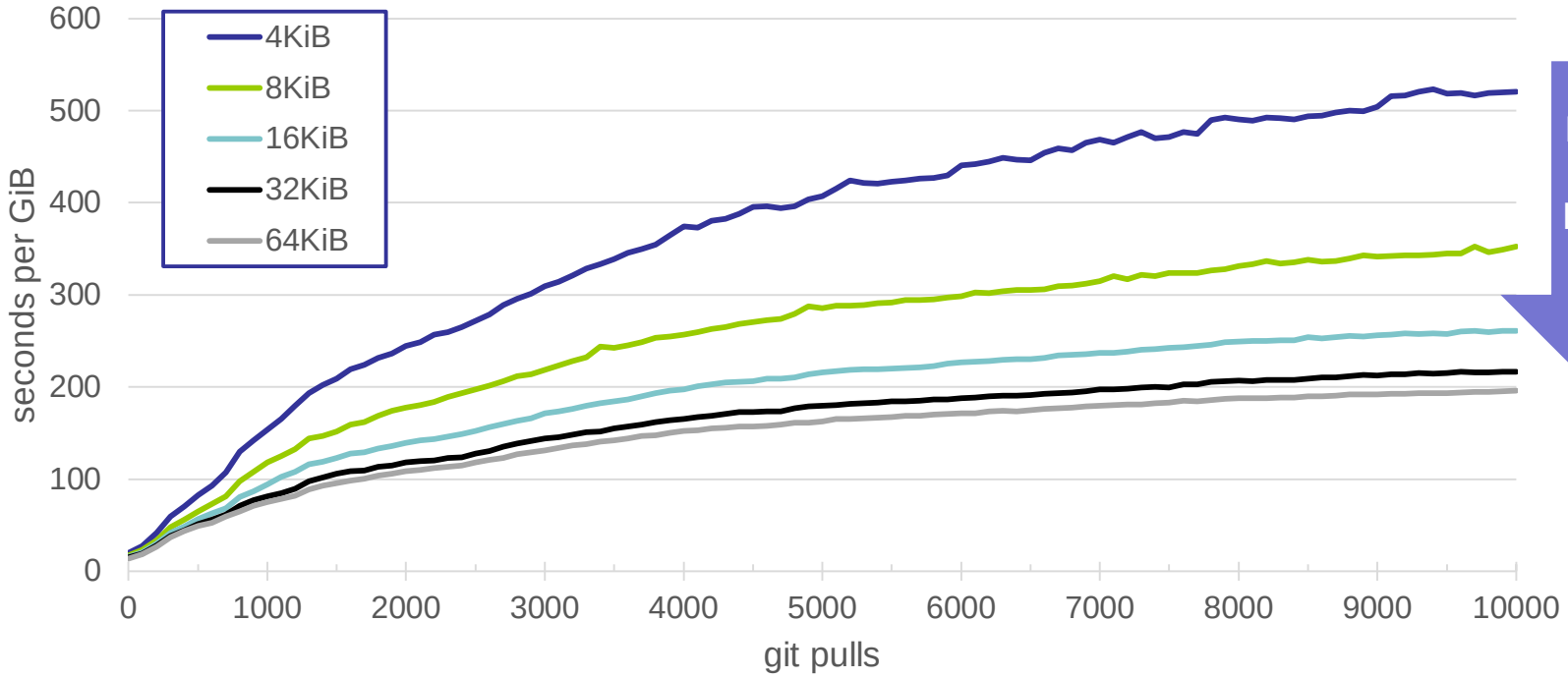
Lower is better



Our Setup: 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Btrfs: Larger nodes => less aging?

Lower is better



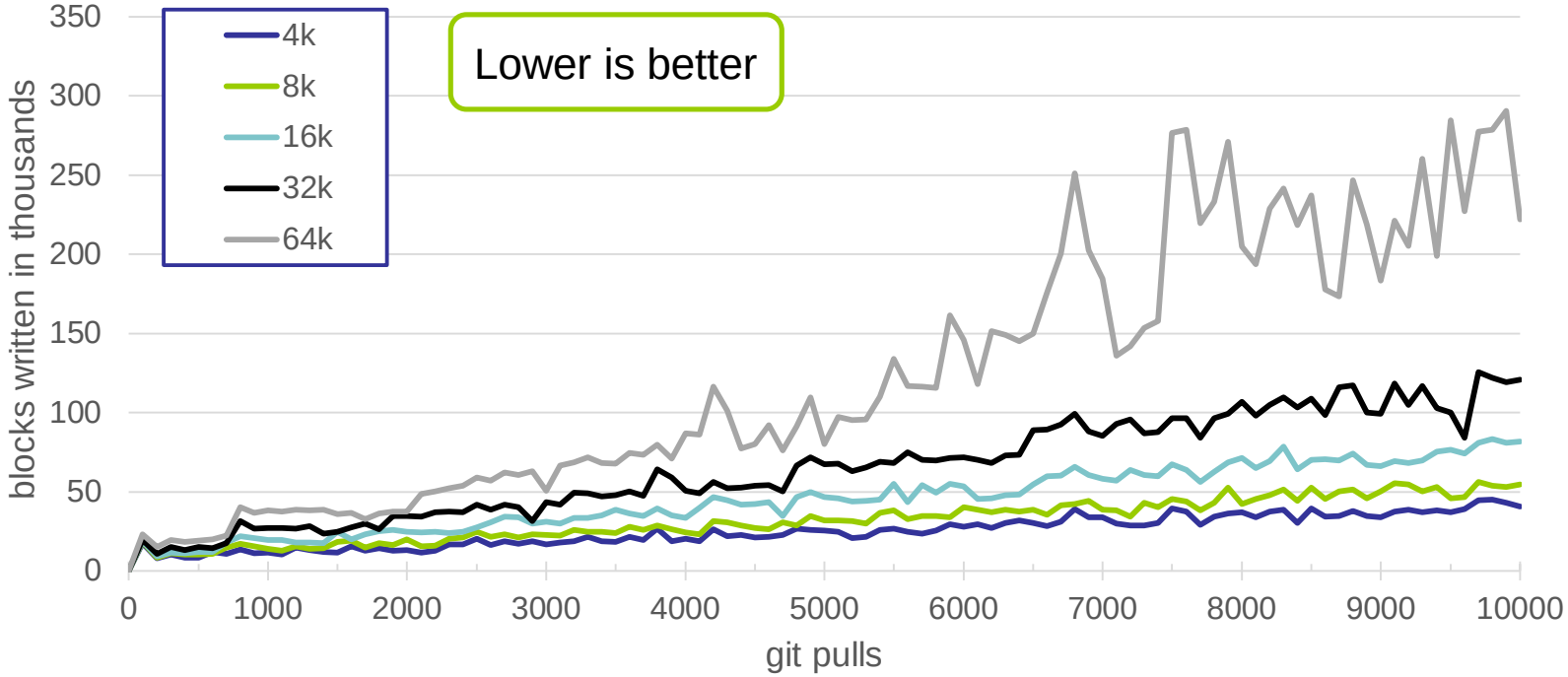
Our Setup: 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Btrfs: Larger nodes => more writing?

$$\text{Write Amplification} = \frac{|\text{I/Os}|}{|\text{Workload}|}$$

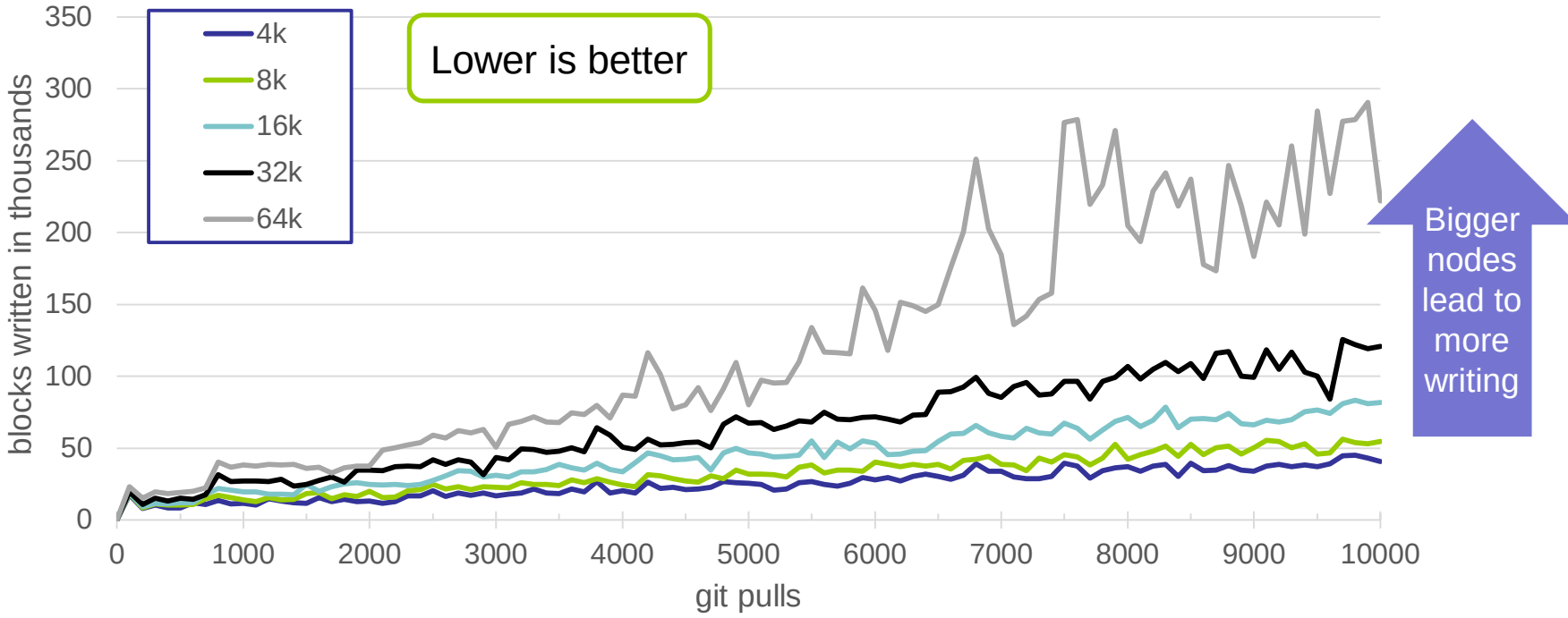
For a B-tree, bigger leaves will mean more write-amplification

Btrfs: Larger nodes => more writing?



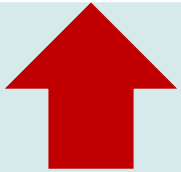
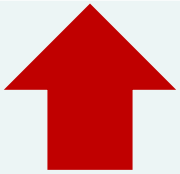
Our Setup: 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Btrfs: Larger nodes => more writing?

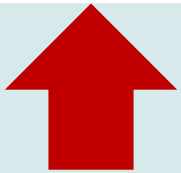
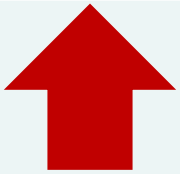


Our Setup: 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

B-tree performance tradeoff

Bigger nodes	Smaller nodes
 Less aging 	 More aging 
 More writing 	 Less writing 

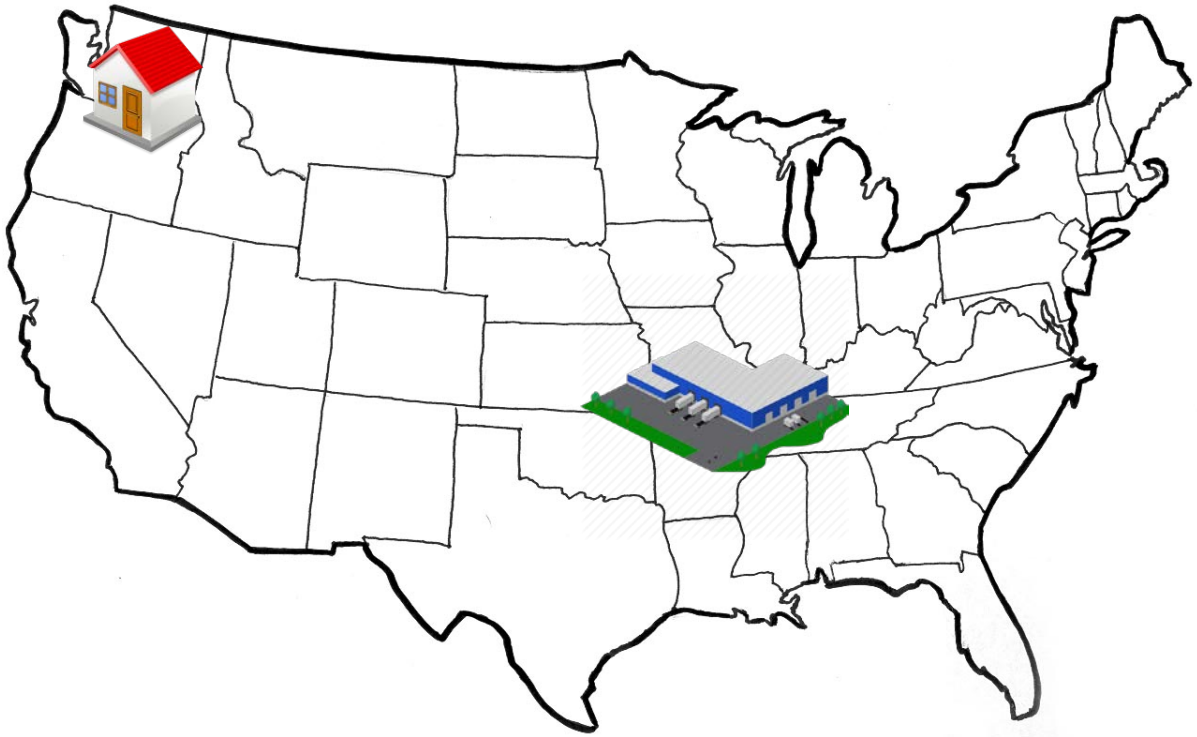
B-tree performance tradeoff

Bigger nodes	Smaller nodes
 Less aging 	 More aging 
 More writing 	 Less writing 

This tradeoff is inherent to B-trees

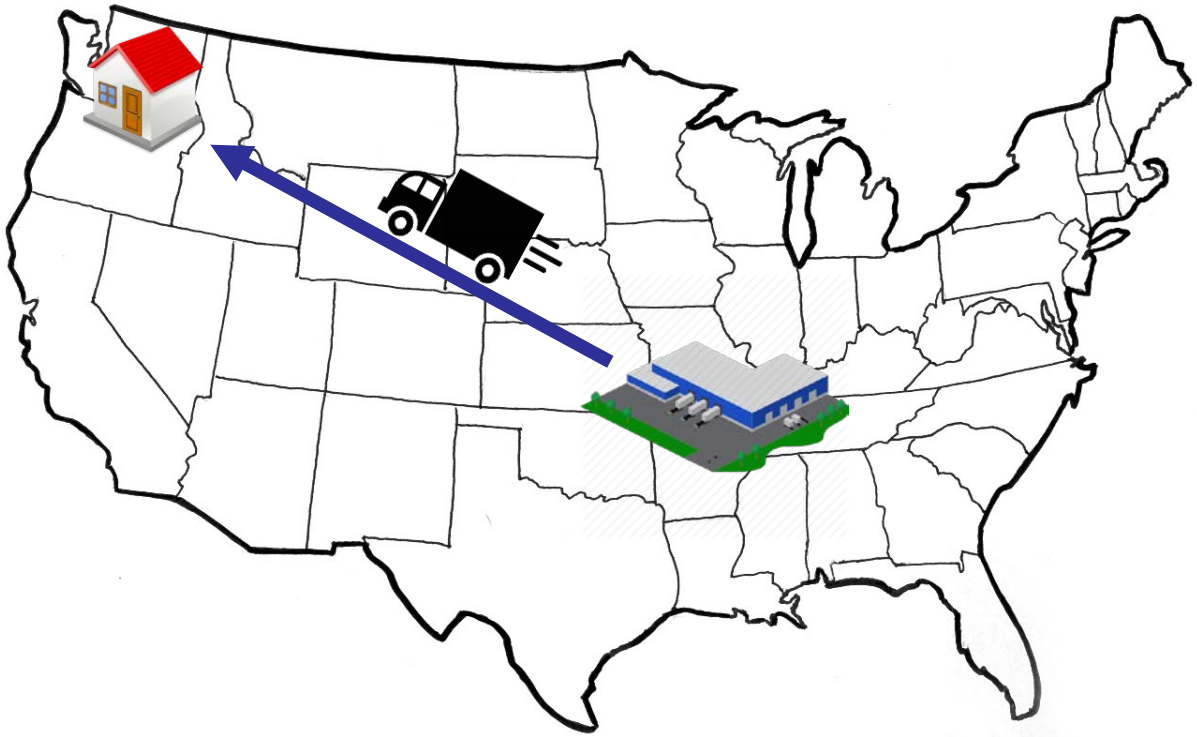
How to prevent aging (with Betrfs)

B^{ε}-trees are B-trees with buffers



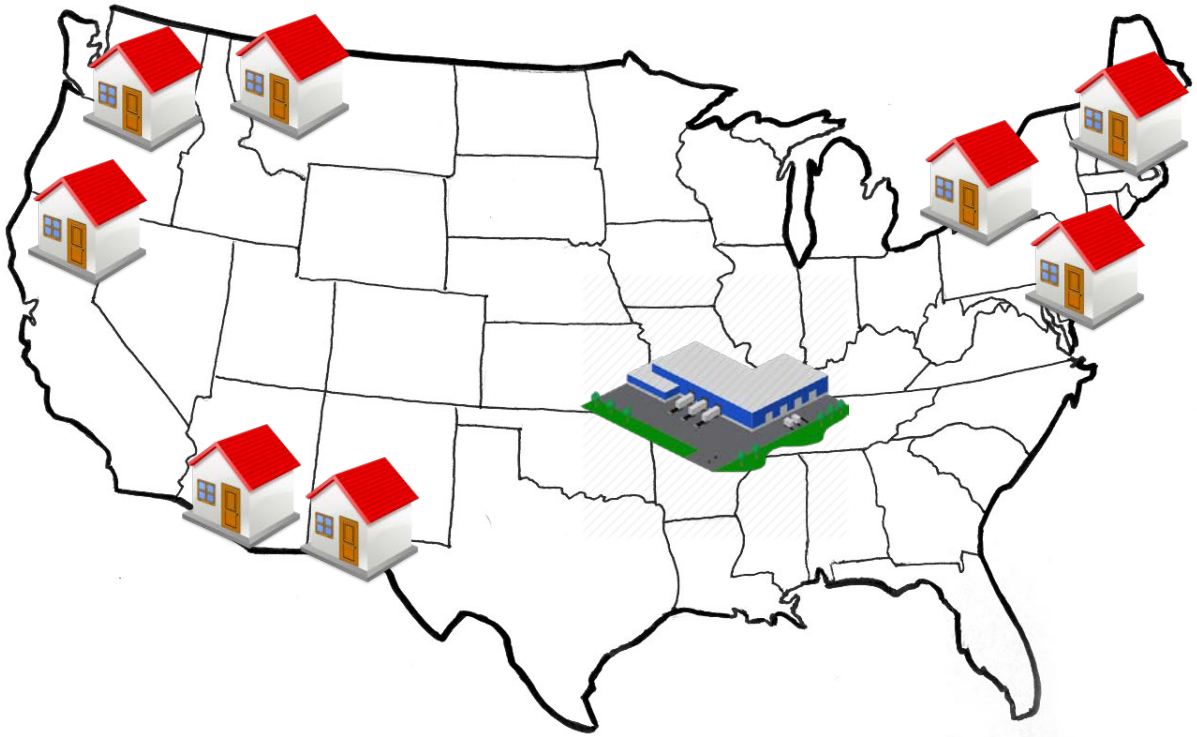
In a B-tree, when changes are made to a leaf, the whole leaf must be read, modified and written out

B^{ε}-trees are B-trees with buffers



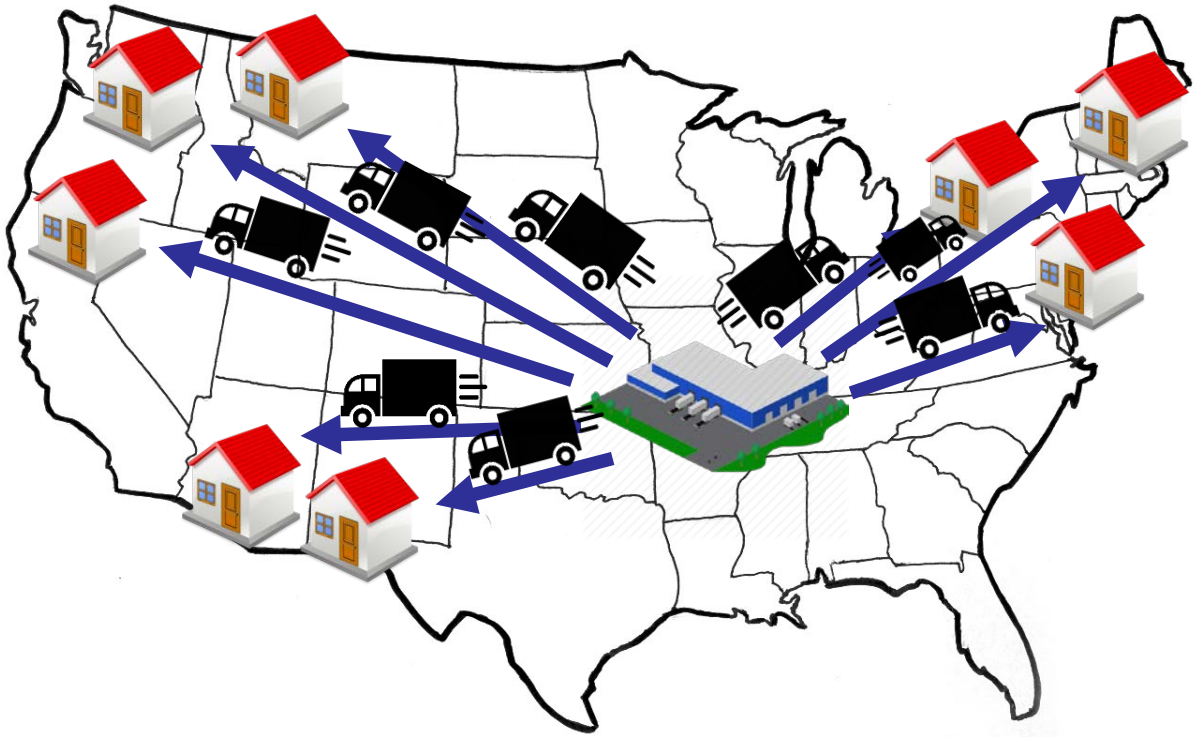
In a B-tree, when changes are made to a leaf, the whole leaf must be read, modified and written out

B^{ε}-trees are B-trees with buffers



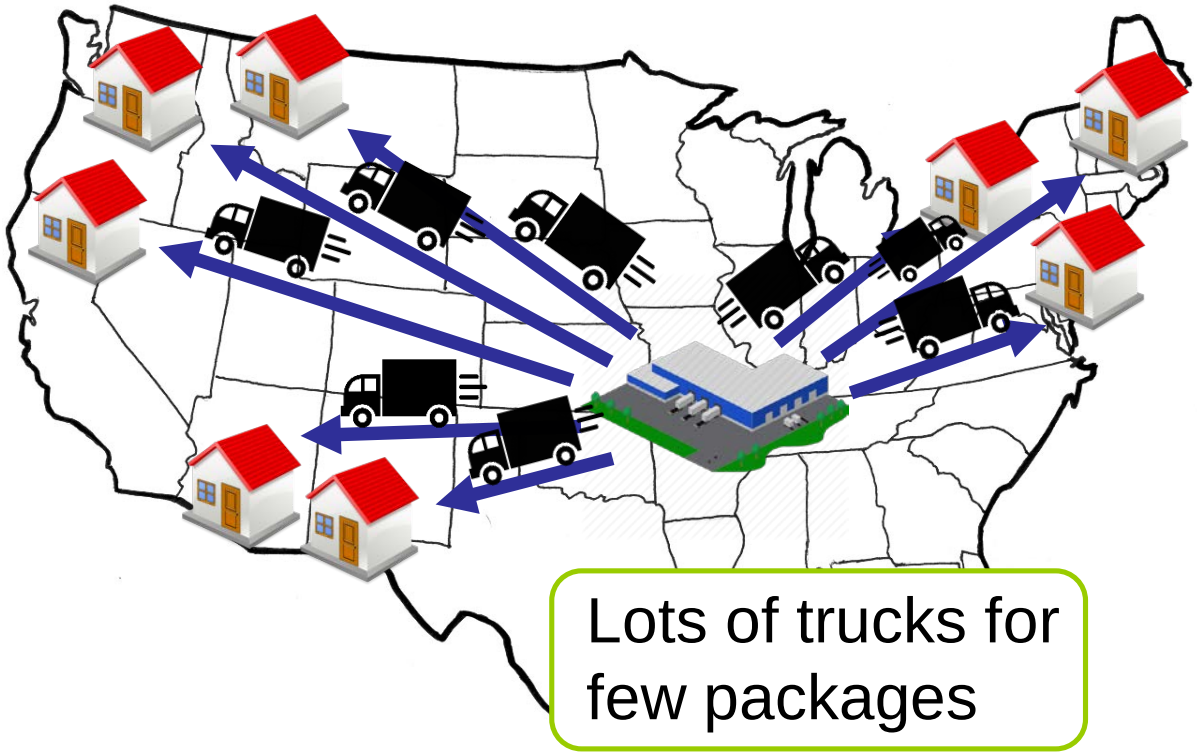
In a B-tree, when changes are made to a leaf, the whole leaf must be read, modified and written out

B^{ε}-trees are B-trees with buffers



In a B-tree, when changes are made to a leaf, the whole leaf must be read, modified and written out

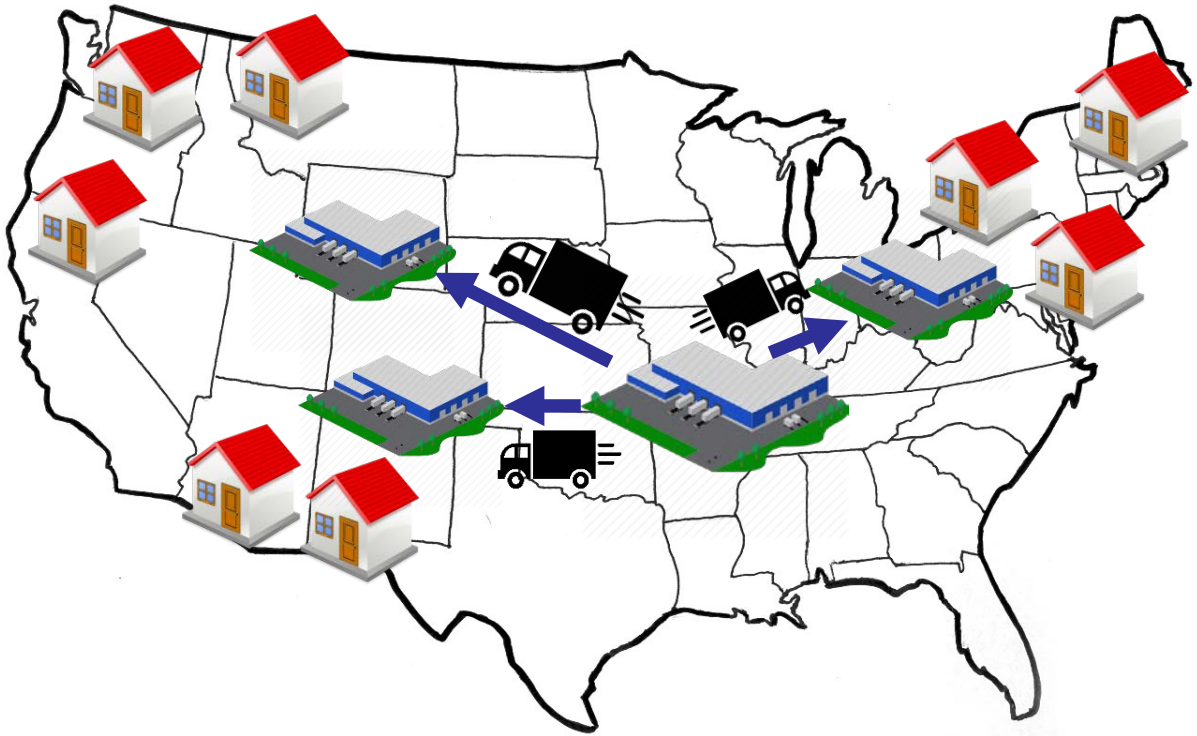
B^{ε}-trees are B-trees with buffers



Lots of trucks for few packages

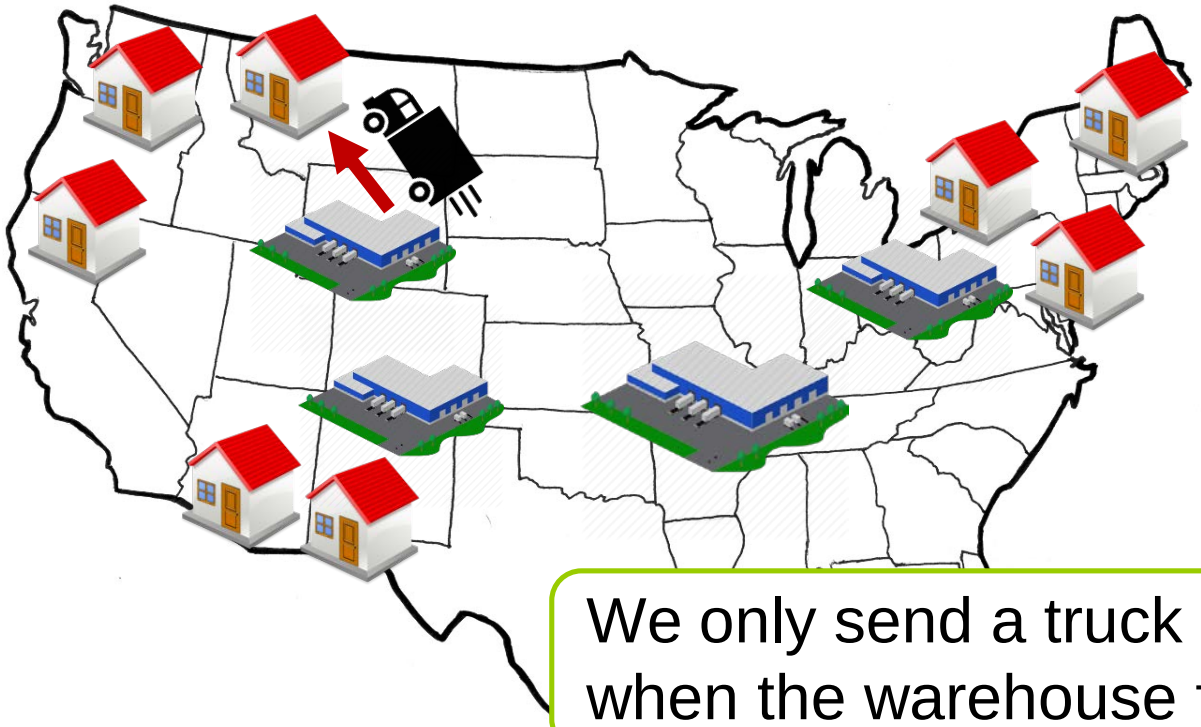
In a B-tree, when changes are made to a leaf, the whole leaf must be read, modified and written out

B^{ε}-trees are B-trees with buffers



In a B^{ε}-trees, intermediate nodes have buffers, aggregating changes en route to the leaves.

B^{ε}-trees are B-trees with buffers



In a B^{ε}-tree, intermediate nodes have buffers, aggregating changes en route to the leaves.

We only send a truck when the warehouse fills

There is no tradeoff!

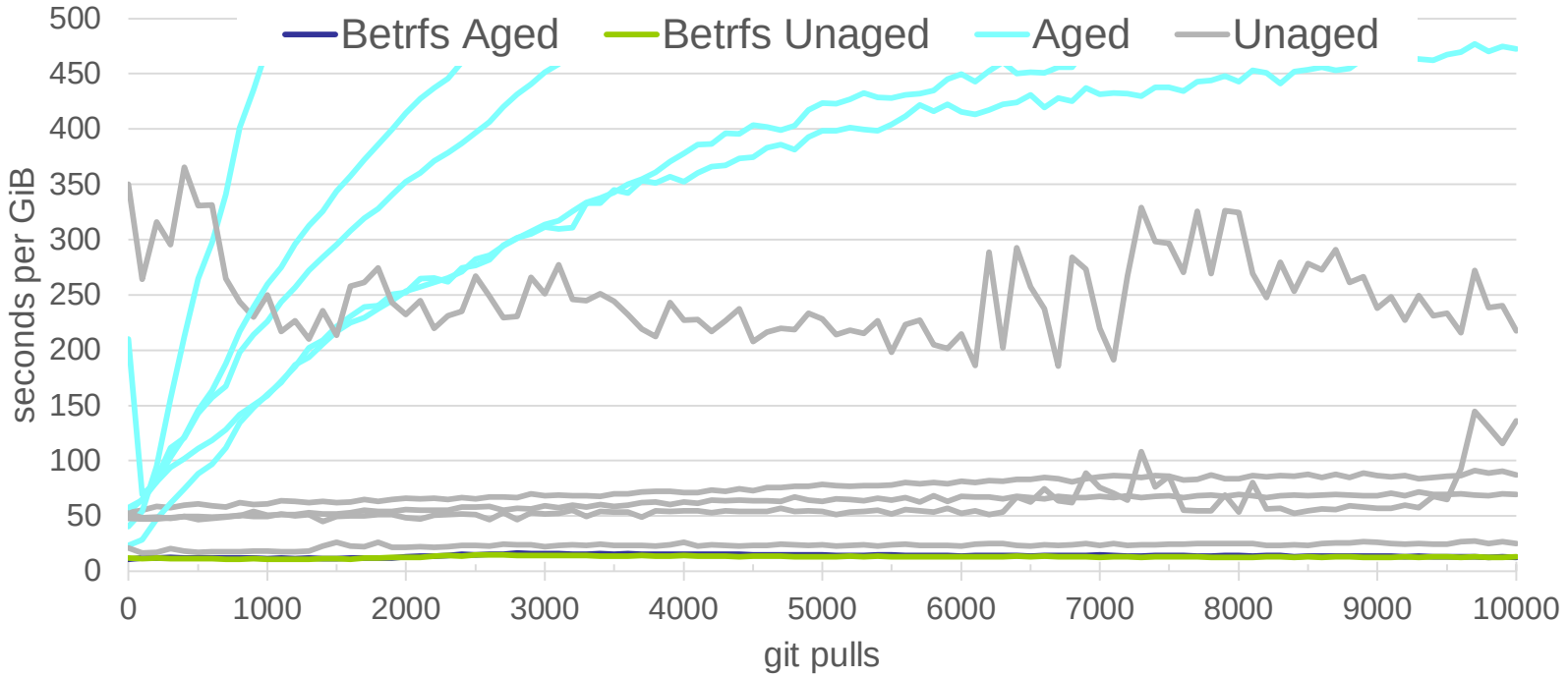
- With a B^{ε}-tree, we only write a node when we've aggregated enough data.
- So we can have big leaves without write-amplification!

Git workload – Betrfs HDD

Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Git workload – Betrfs HDD

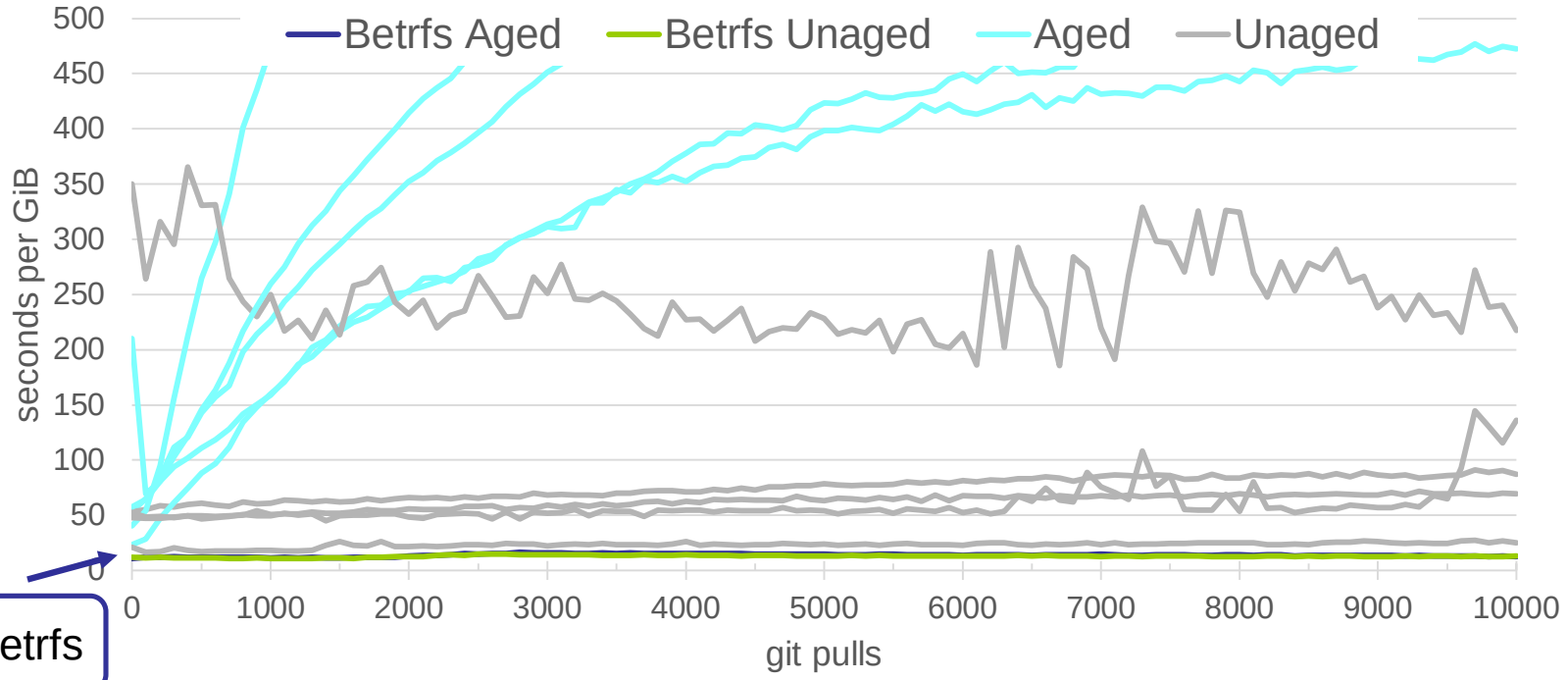
Lower is better



Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Git workload – Betrfs HDD

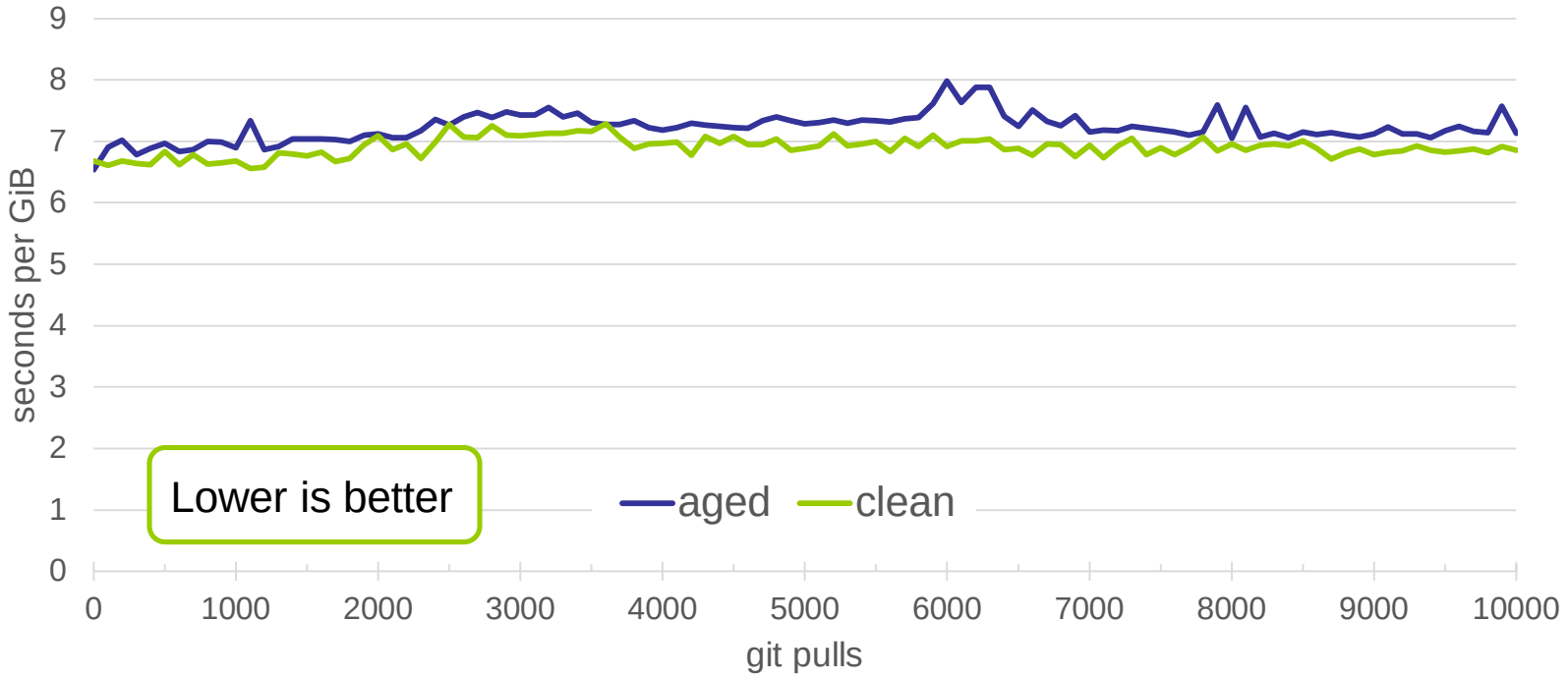
Lower is better



Betrfs

Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

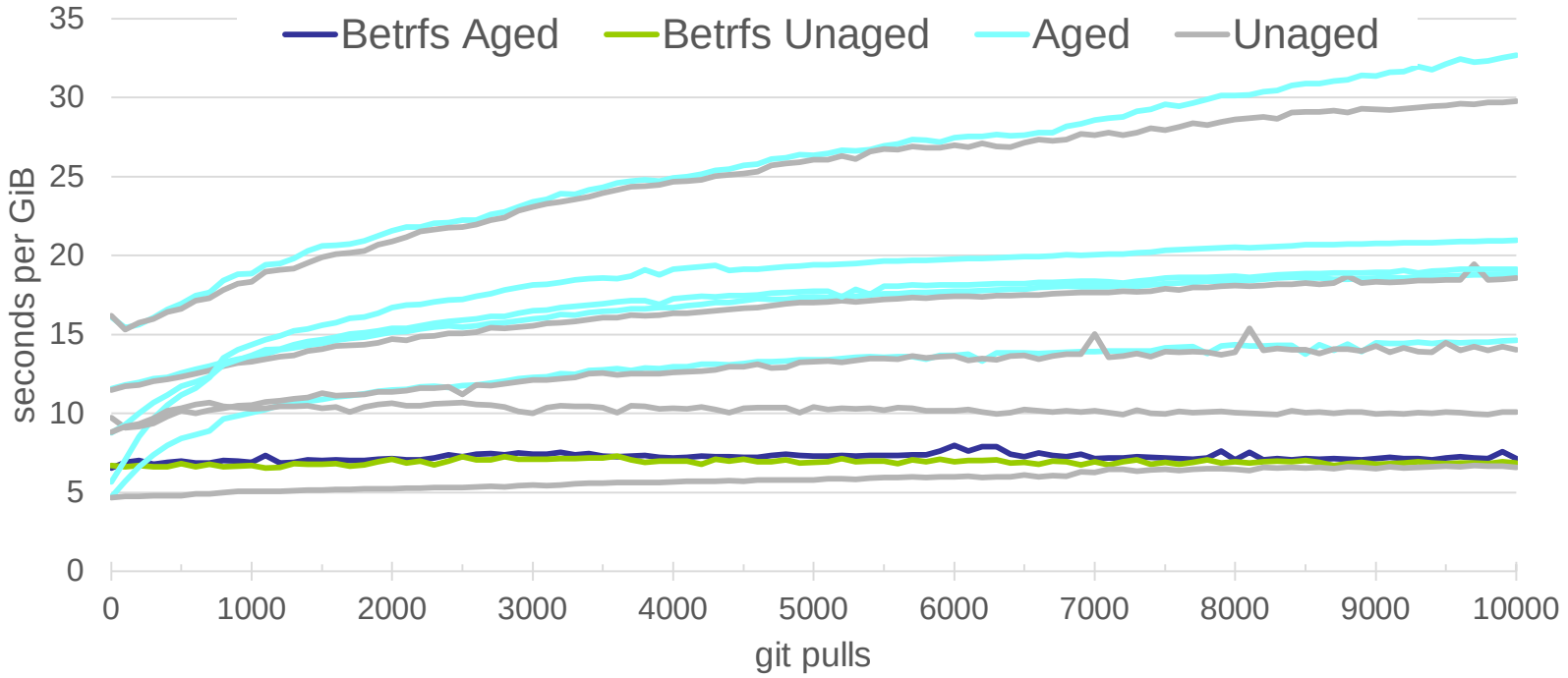
Git workload – Betrfs SSD



Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Git workload – Betrfs HDD

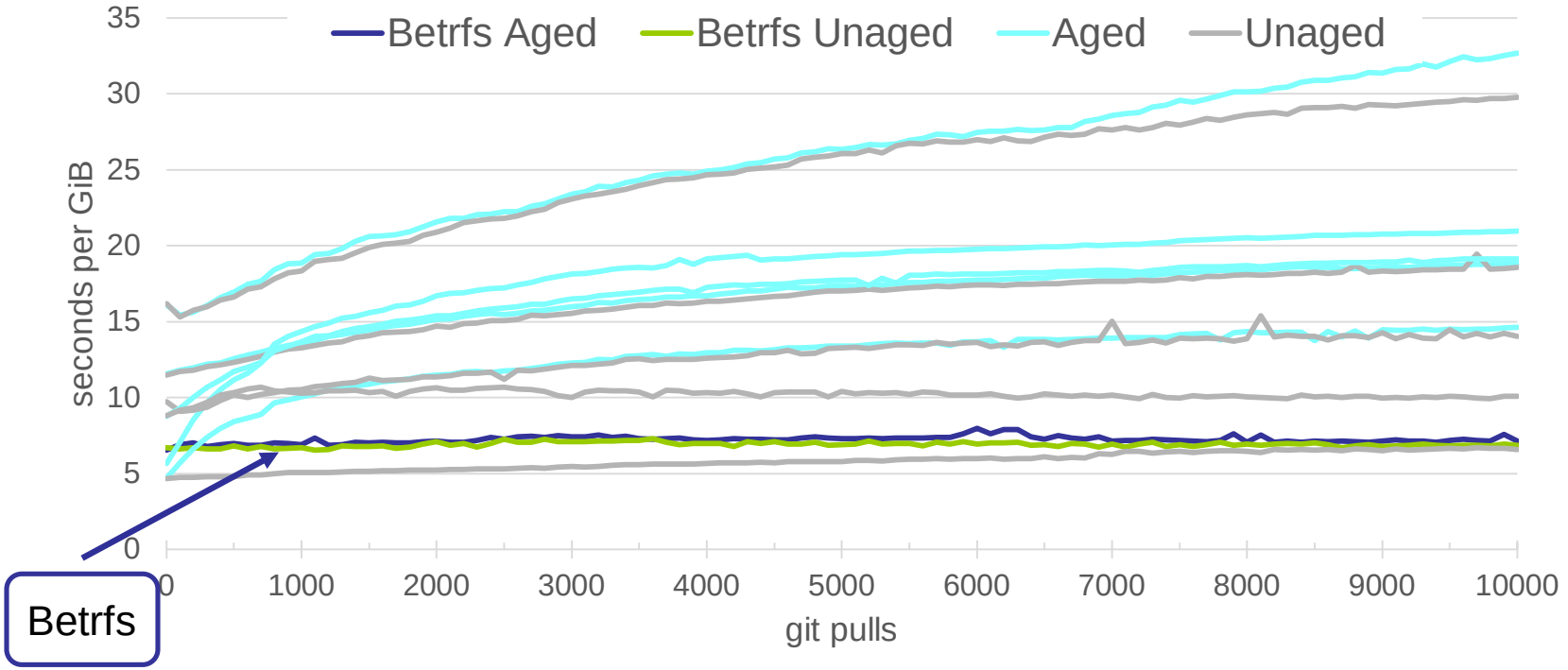
Lower is better



Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Git workload – Betrfs HDD

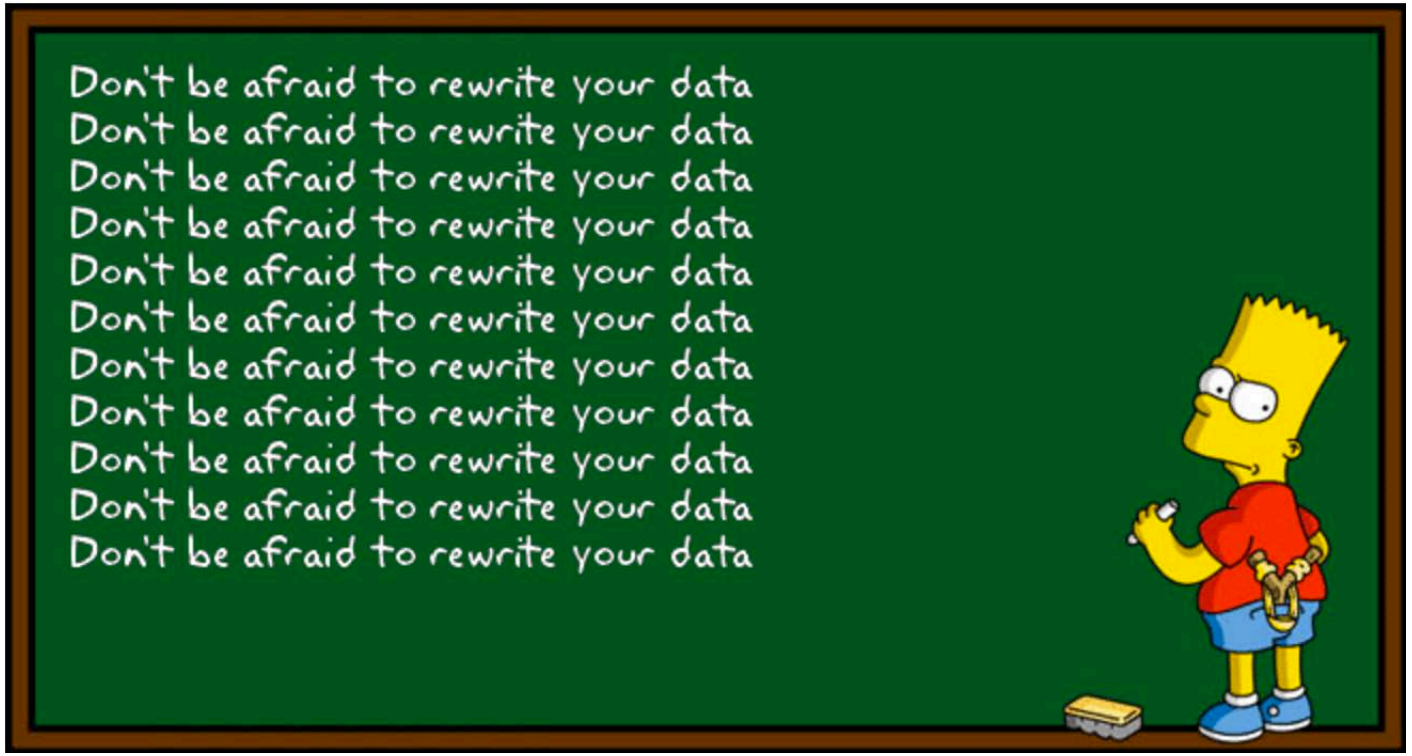
Lower is better



Betrfs

Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Thank you!



Alex Conway
betrfs.org
alexander.conway@rutgers.edu