







File Systems Fated for Senescence? Nonsense, Says Science!

Alex Conway^{}, Ainesh Bakshi^{}, Yizheng Jiao^{}, Yang Zhan^{}, Michael A. Bender^{}, William Jannen^{}, Rob Johnson^{}, Bradley C. Kuszmaul^{}, Donald E. Porter^{}, Jun Yuan^{} and Martin Farach-Colton^{}

^{}Rutgers University, ^{}The University of North Carolina at Chapel Hill,
^{}Stony Brook University, ^{}Oracle Corporation and Massachusetts Institute of Technology,
^{}Farmingdale State College of SUNY

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

File Systems Fated for
Senescence? Nonsense,

Says Science;

old age

The Essence of

Semperjuvenescence is

Coalescence!

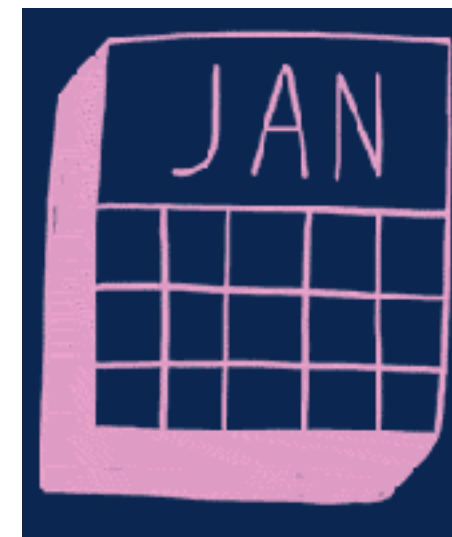
merging together

being young forever

File System Aging

Aging is fragmentation over time

Performance



In this talk

Do file systems age?

What can we do about it?

Is aging a problem?

Is aging a problem?



does my file system need defragmentation



All

News

Videos

Images

Shopping

More

Settings

Tools

About 409,000 results (0.87 seconds)

Why Linux Doesn't Need Defragmenting - How-To Geek

<https://www.howtogeek.com/.../htg-explains-why-linux-doesnt-need-defragmenting/> ▼

May 30, 2012 - To understand why Linux **file systems** don't **need defragmenting** in normal use – and Windows ones **do** – you'll **need** to understand why ...

You visited this page on 2/20/17.

File Systems - Which Need Defragmenting? - PCMech

<https://www.pcmech.com/article/file-systems-which-need-defragmenting/> ▼

Nov 30, 2007 - The FAT **file system** is particularly susceptible to **fragmentation** by its very design. More information about FAT **can** be found on Wikipedia.

What doesn't need defragmentation? Linux or the ext2 ext3 FS?

unix.stackexchange.com/.../what-doesnt-need-defragmentation-linux-or-the-ext2-ext3... ▼

May 13, 2013 - Because it's using the ext2/ext3 **file system**, or because it's Linux? ... And they also **have** an article asking "**Do** you really **need** to **defrag**?" I'm kind of bad to revise my language without correcting any problems the revision ...

You visited this page on 2/20/17.

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.”

Is aging a problem?

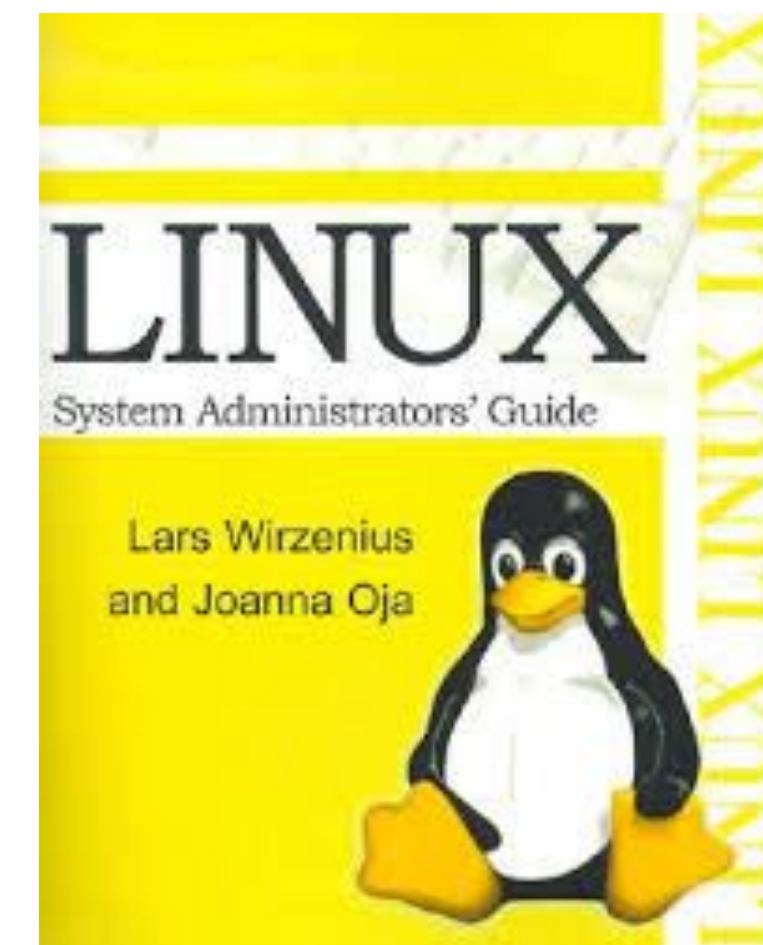
I'm Feeling Lucky

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

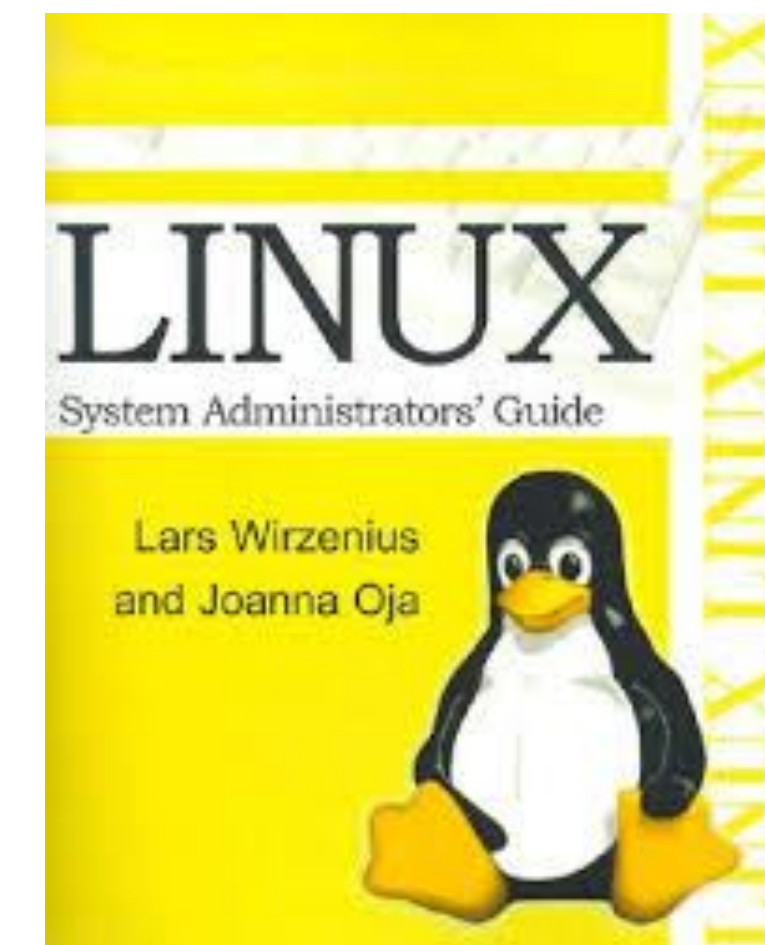
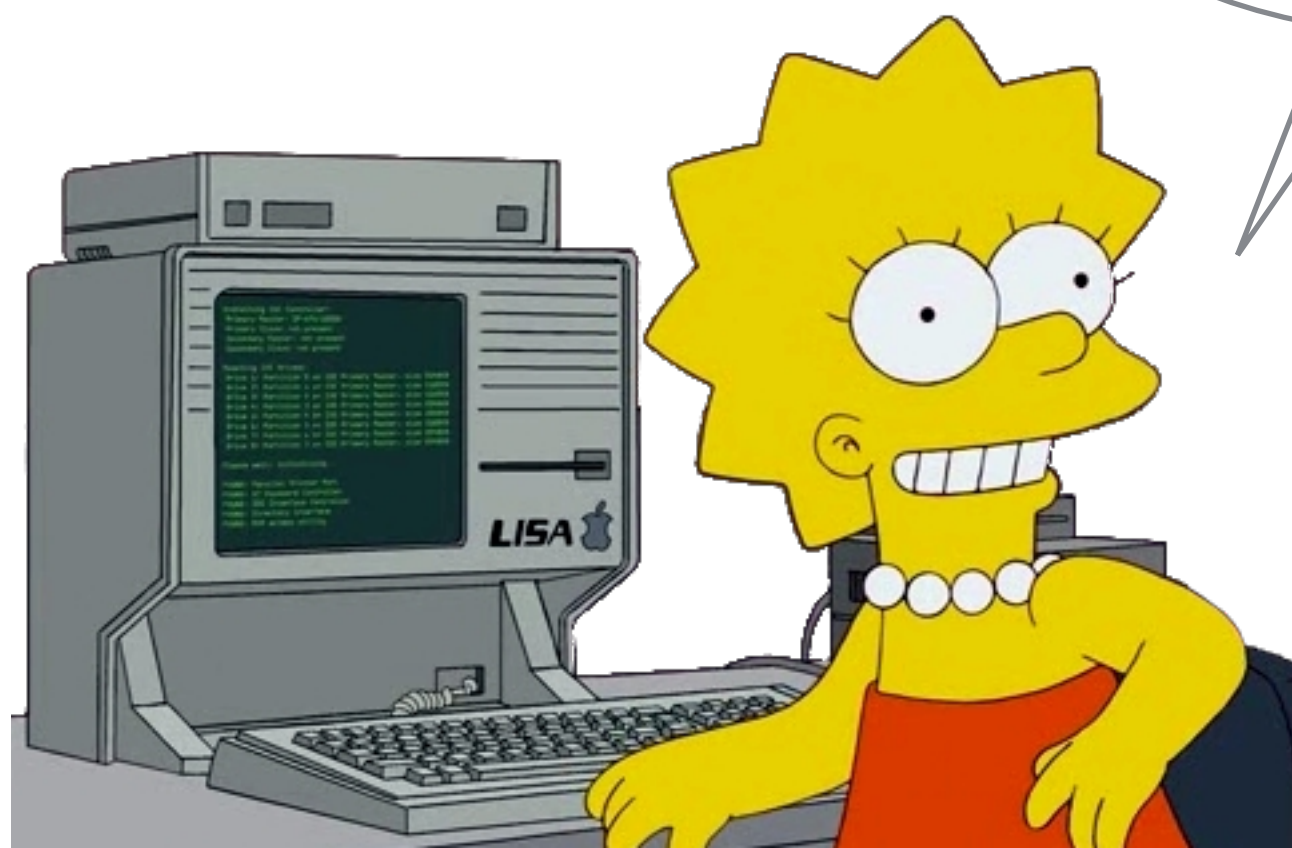
Chris Hoffman at howtogeek.com says:

“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.”

Nope

“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?



file system aging



Scholar

About 2,340,000 results (0.07 sec)

Articles

Case law

My library

File system aging—increasing the relevance of **file system** benchmarks

[KA Smith, MI Seltzer - ACM SIGMETRICS Performance Evaluation ...](#), 1997 - [dl.acm.org](#)

Abstract Benchmarks are important because they provide a means for users and researchers to characterize how their workloads will perform on different systems and different **system** architectures. The field of **file system** design is no different from other areas

[Cited by 131](#) [Related articles](#) [All 15 versions](#) [Cite](#) [Save](#)

Is aging a problem?

Aging happens in real filesystems

- Smith and Seltzer ('97)

Benchmarks should incorporate aging

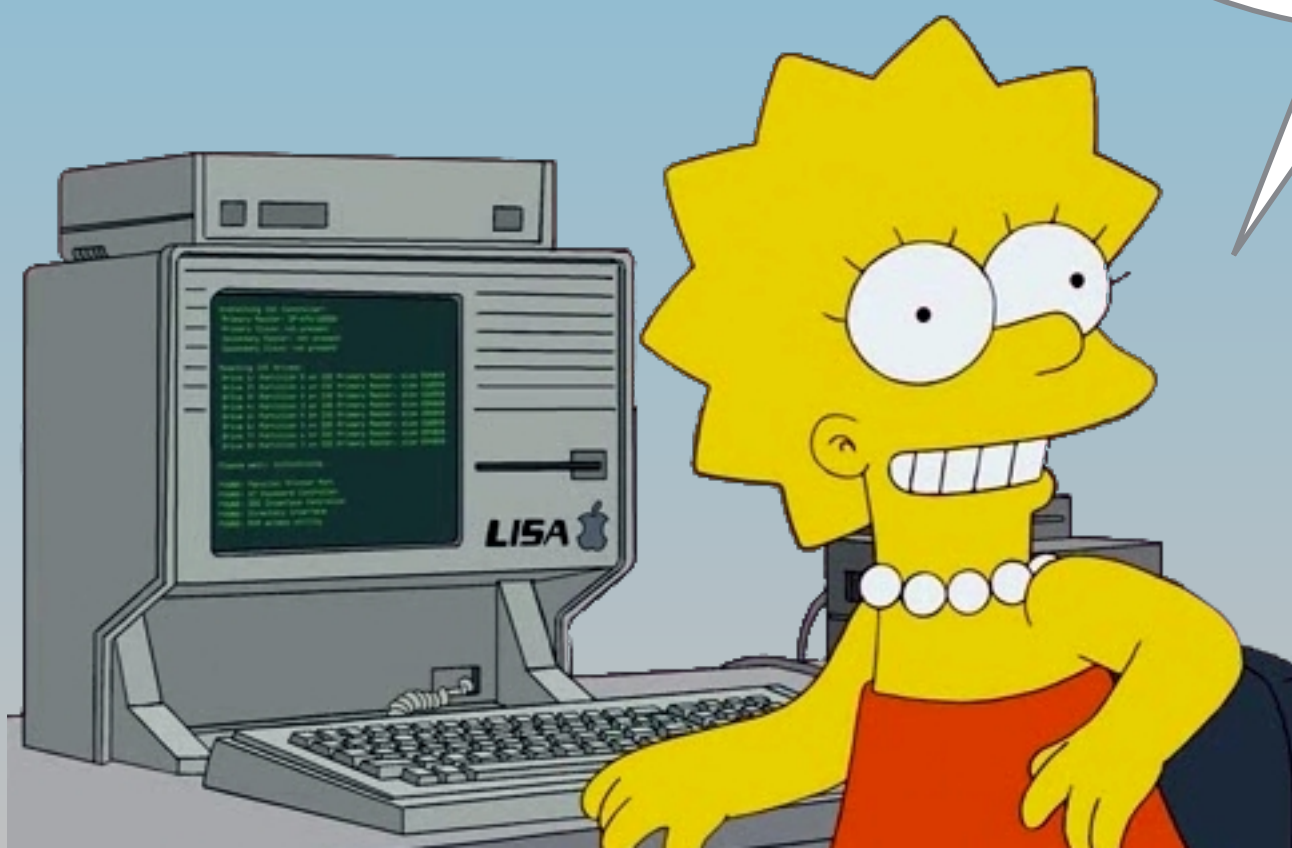
- Zhu, Chen and Chiueh ('05)
- Agrawal, A. Arpaci-Dusseau and R. Arpaci-Dusseau ('09)

Yep



Is aging a problem?

Nope



Yep



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

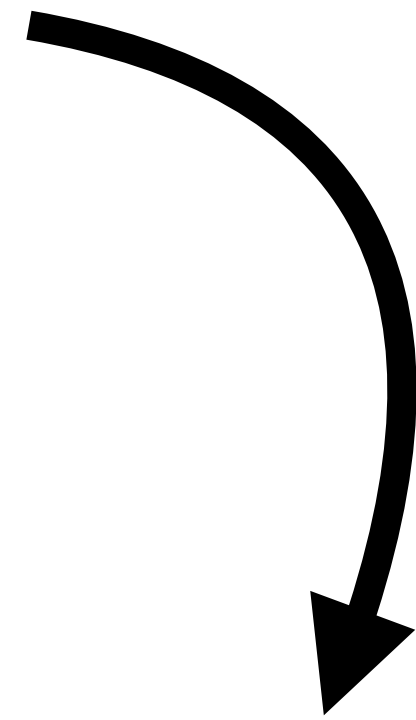
Simulating a Developer

Simulating a Developer



get coffee

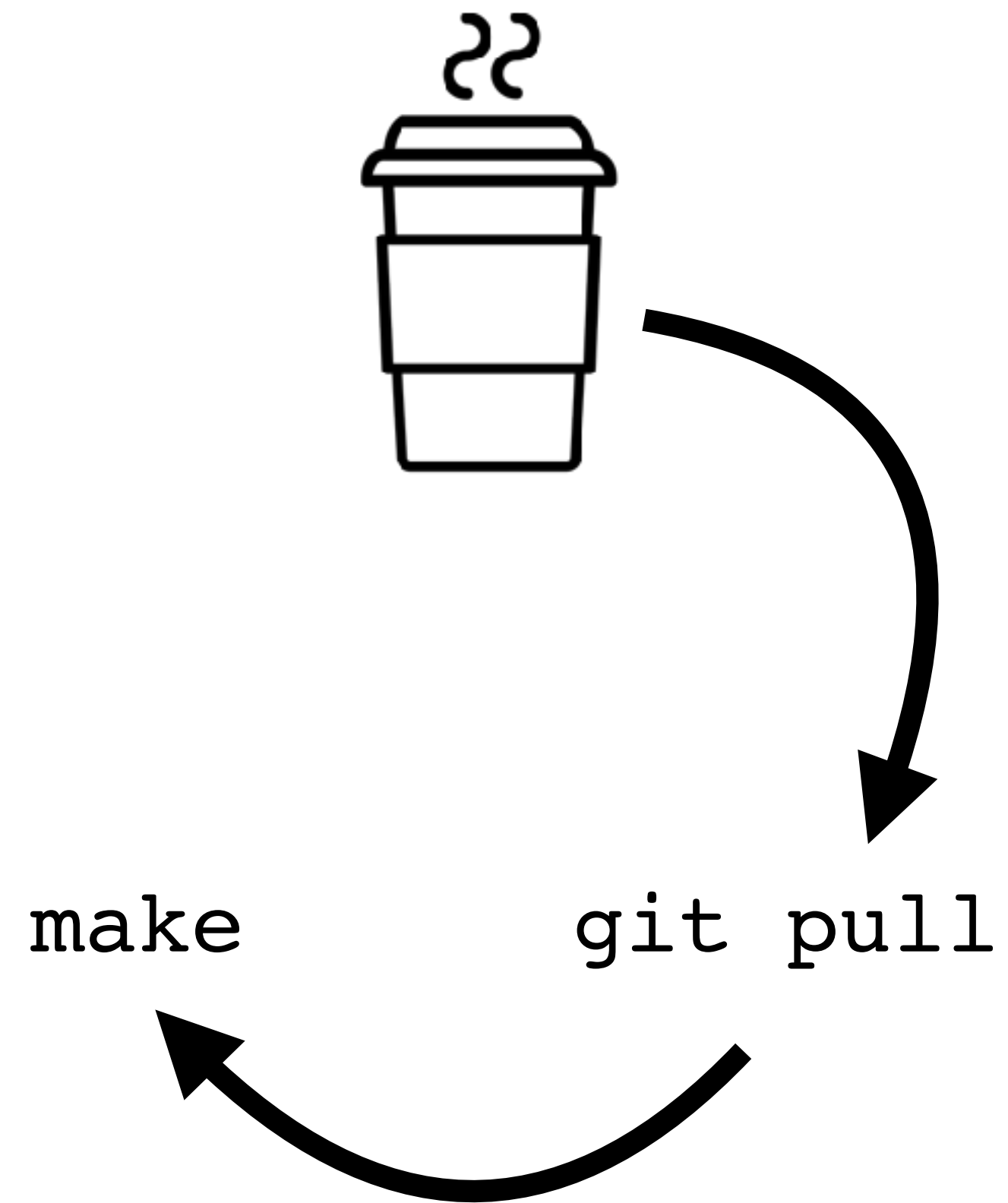
Simulating a Developer



git pull

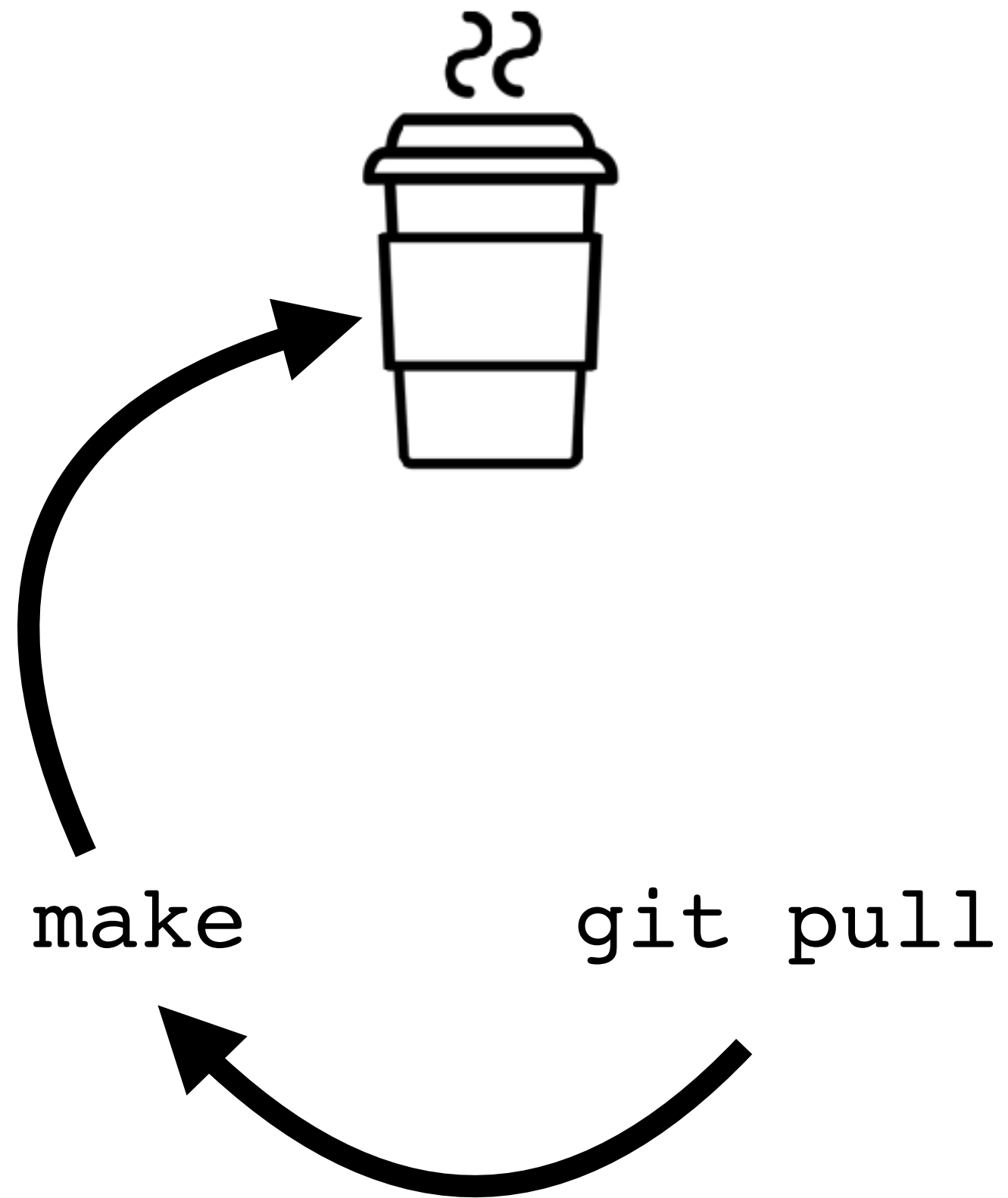
get coffee
git pull

Simulating a Developer



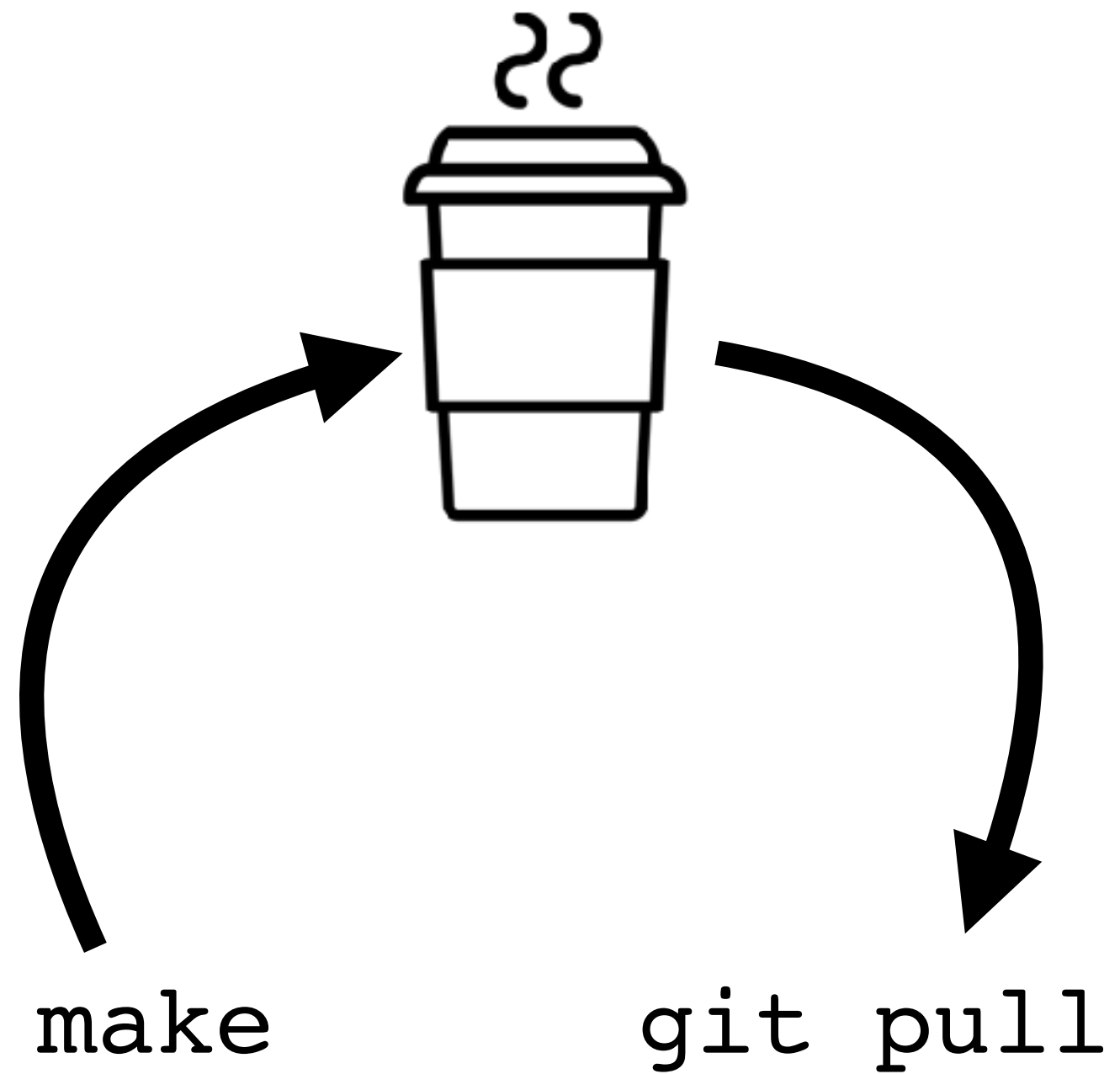
get coffee
git pull
make

Simulating a Developer



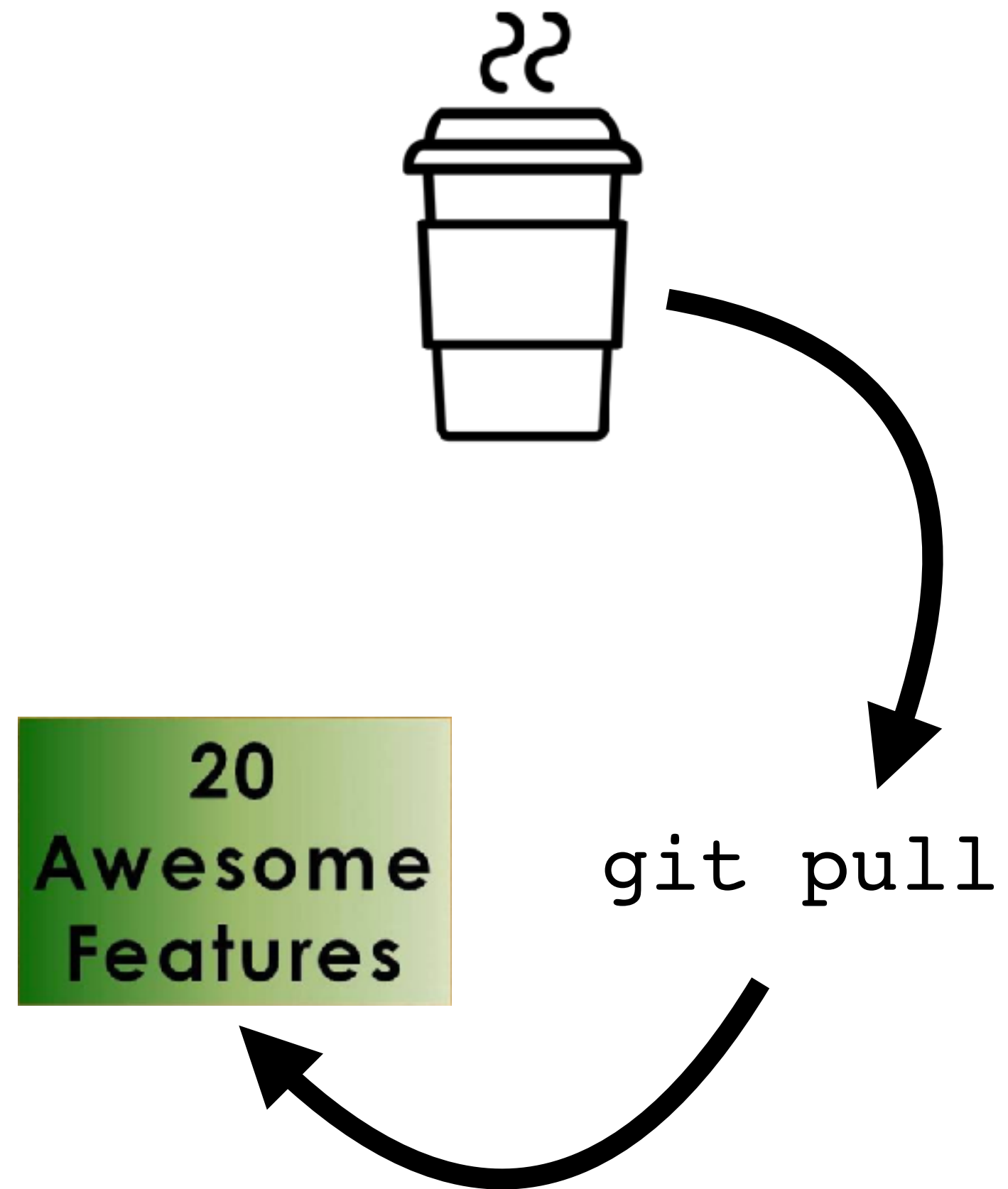
```
get coffee  
git pull  
make  
get coffee
```

Simulating a Developer



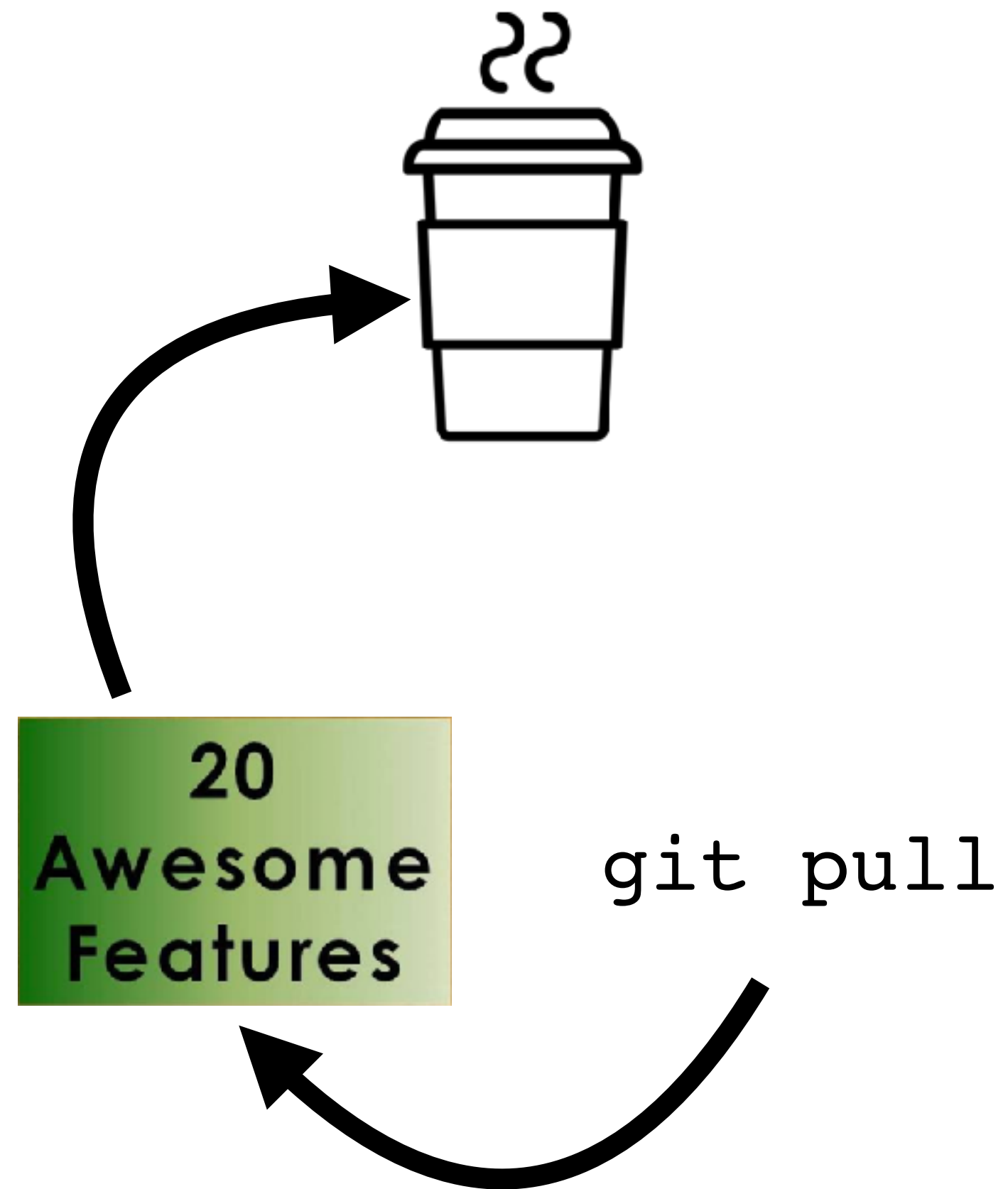
```
get coffee  
git pull  
make  
get coffee  
git pull
```

Simulating a Developer



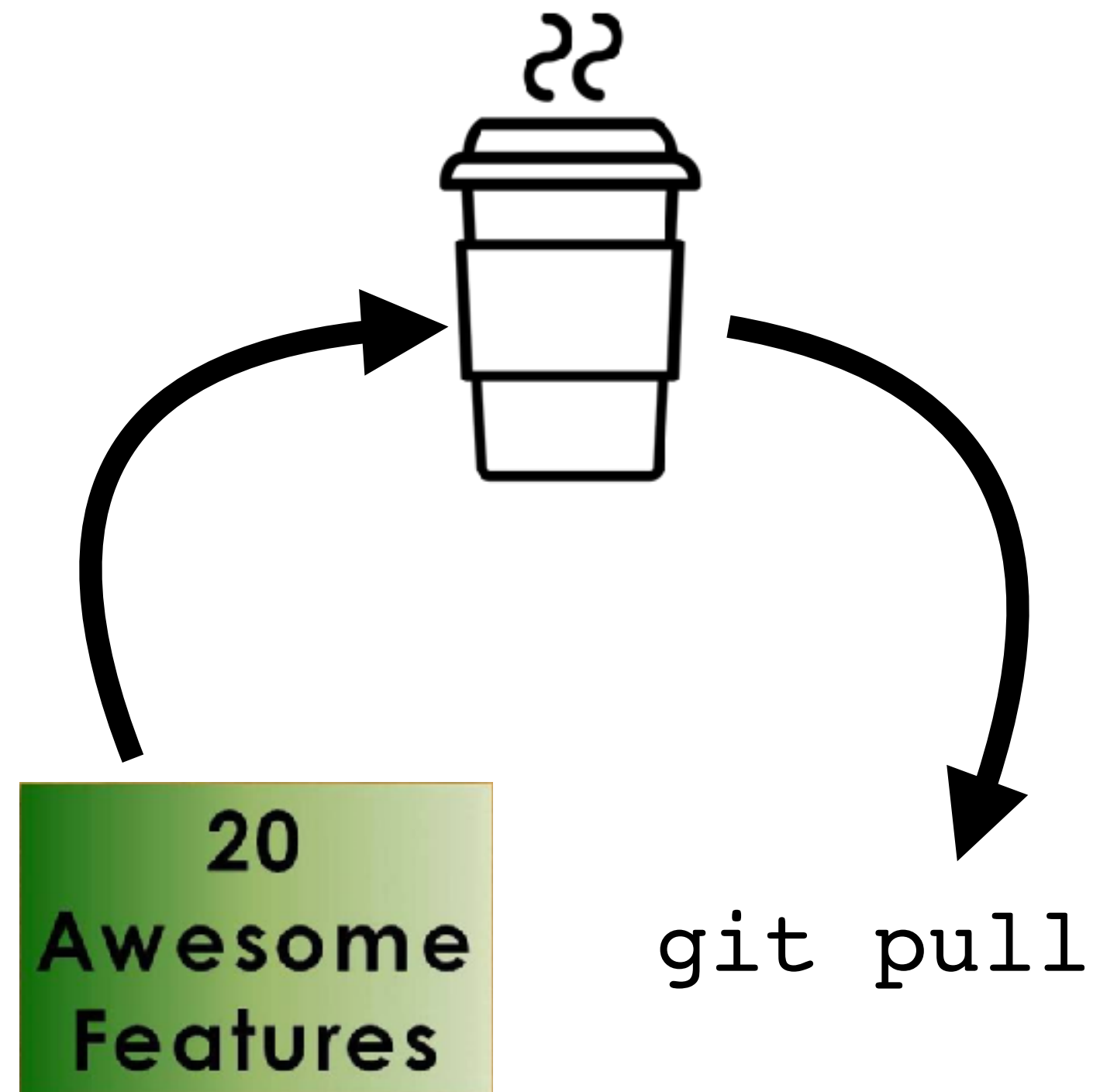
```
get coffee
git pull
make
get coffee
git pull
add awesome features
```


Simulating a Developer



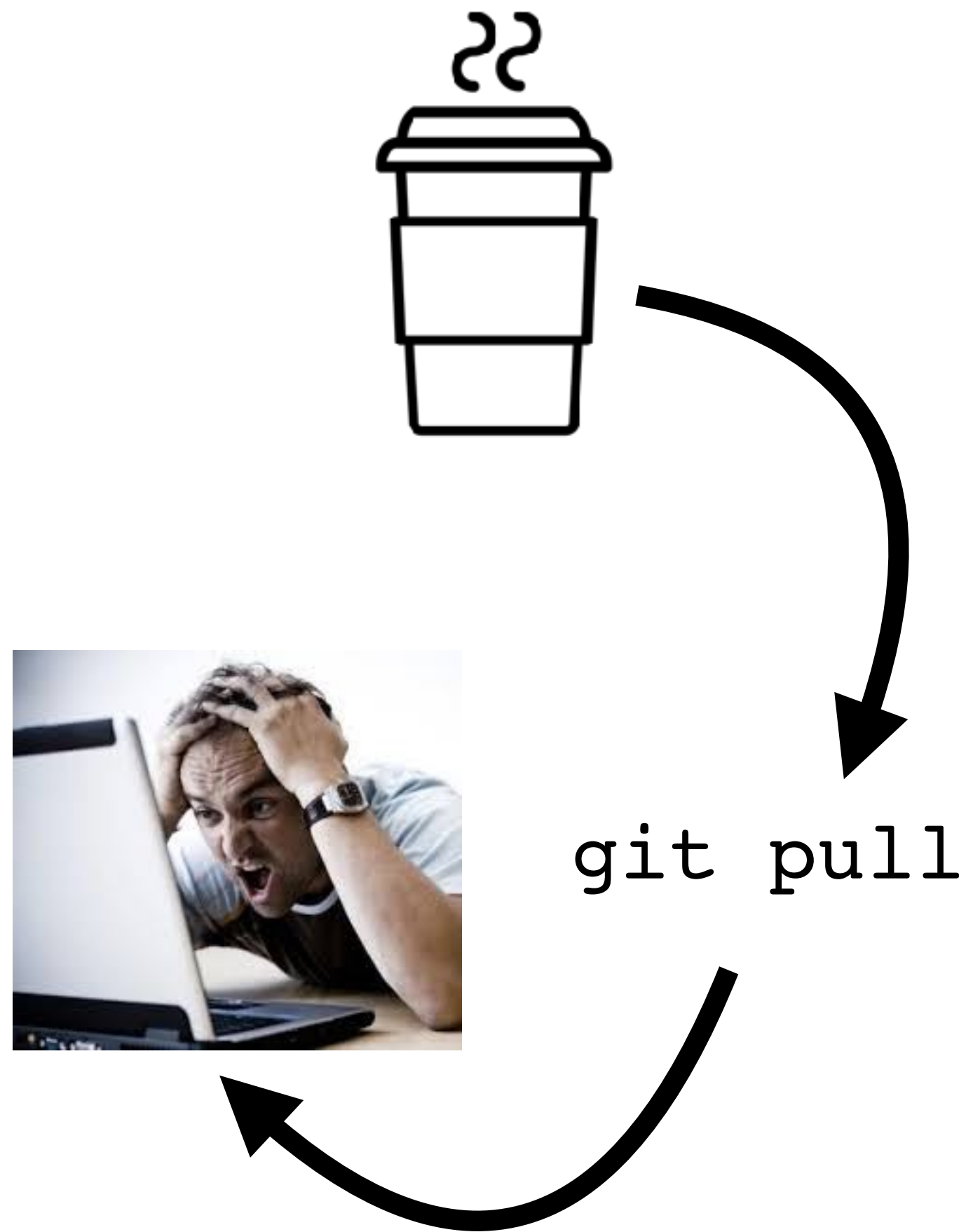
```
get coffee
git pull
make
get coffee
git pull
add awesome features
get coffee
```

Simulating a Developer



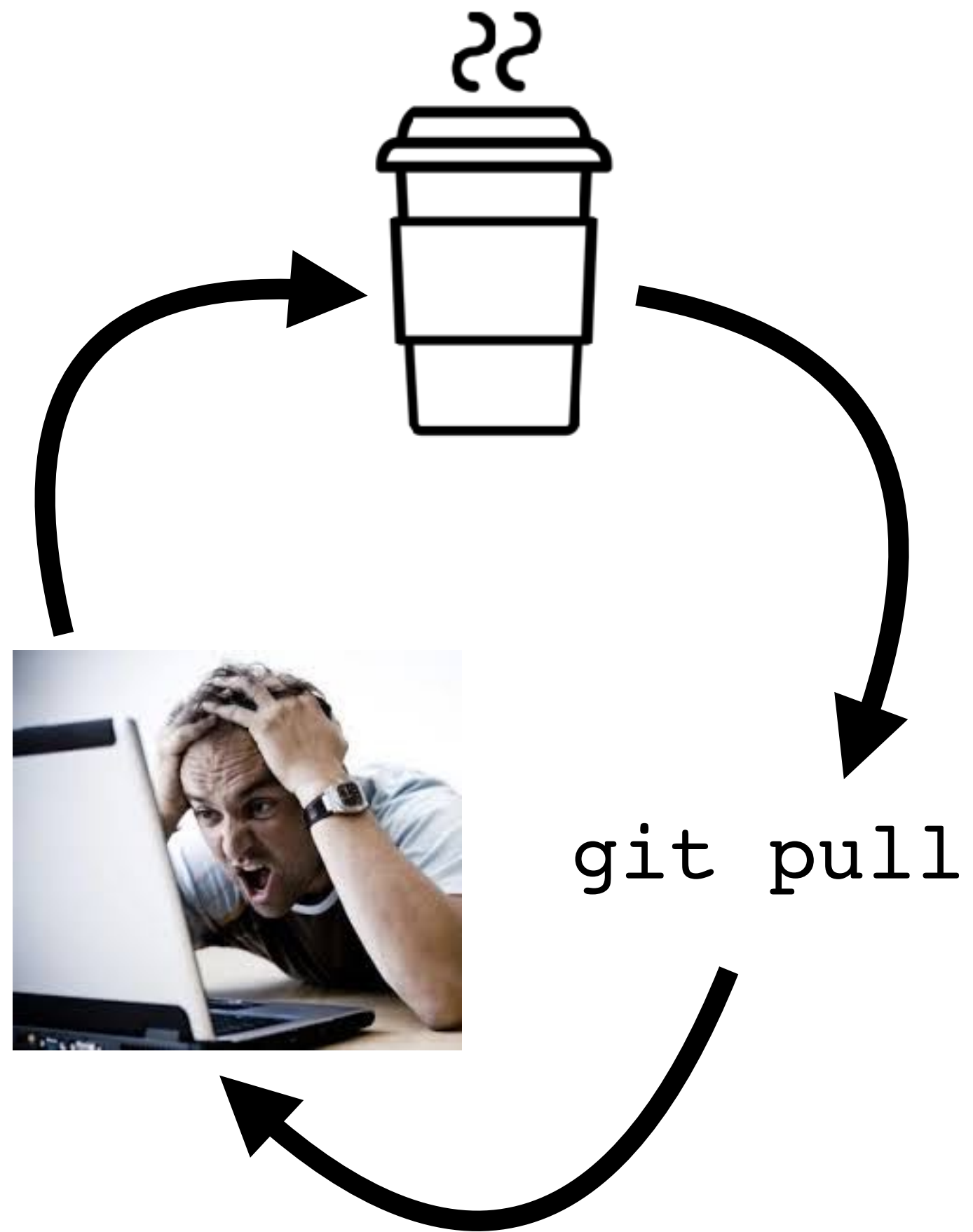
```
get coffee
git pull
make
get coffee
git pull
add awesome features
get coffee
git pull
```

Simulating a Developer



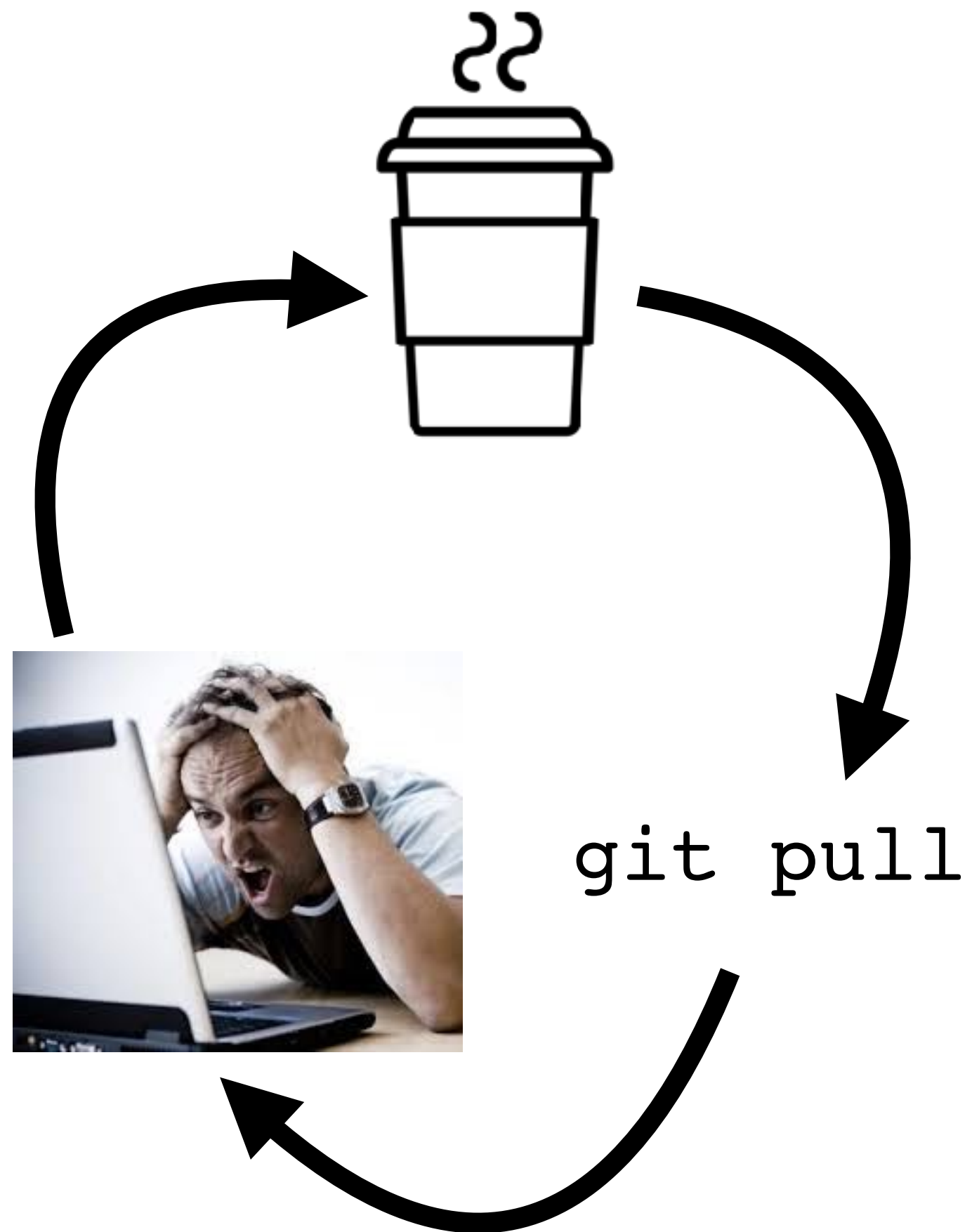
```
get coffee  
git pull  
make  
get coffee  
git pull  
add awesome features  
get coffee  
git pull  
fix bugs
```

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  
...
```

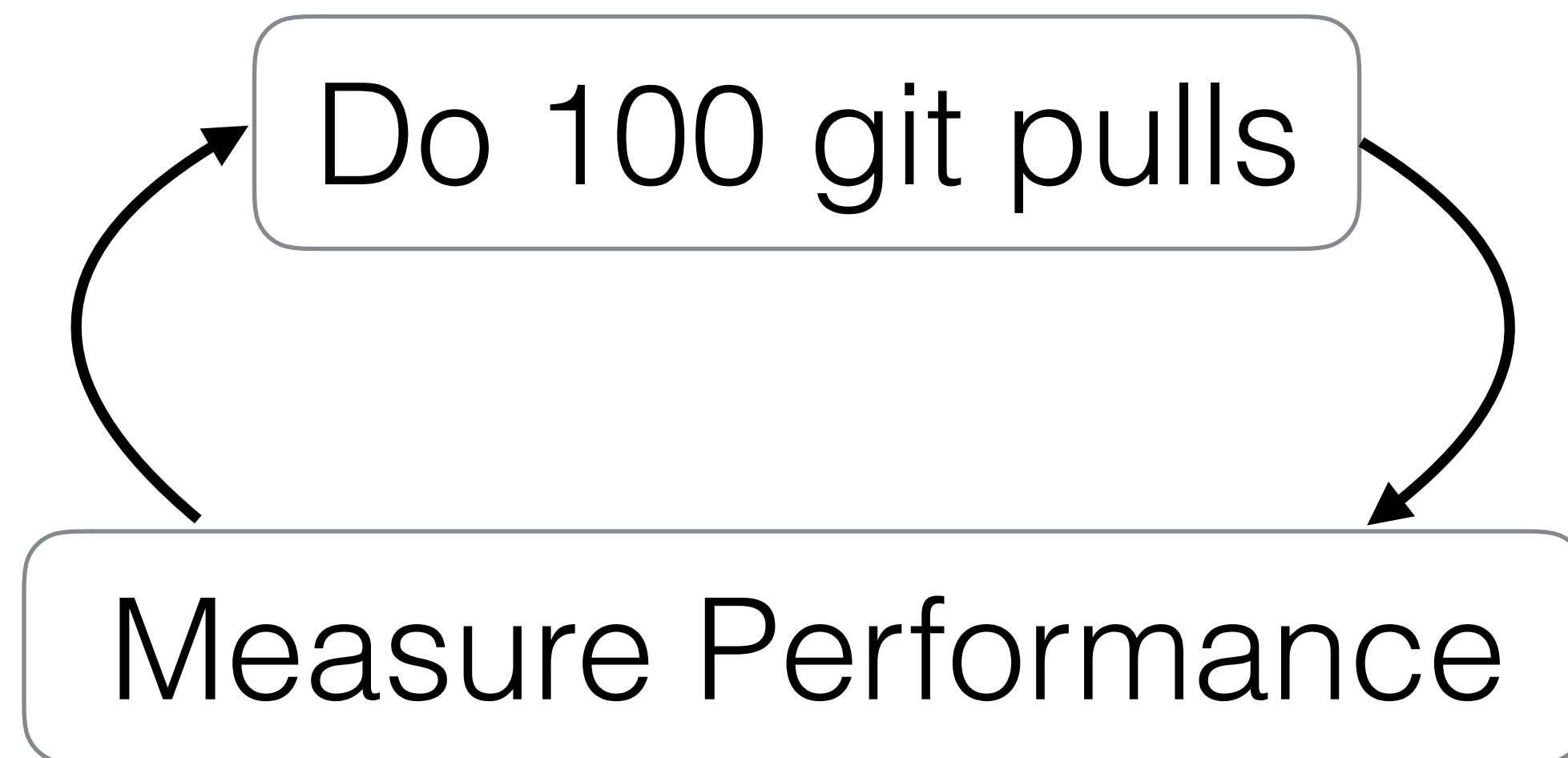
We can simulate a developer by replaying Git histories

Simulating a Developer



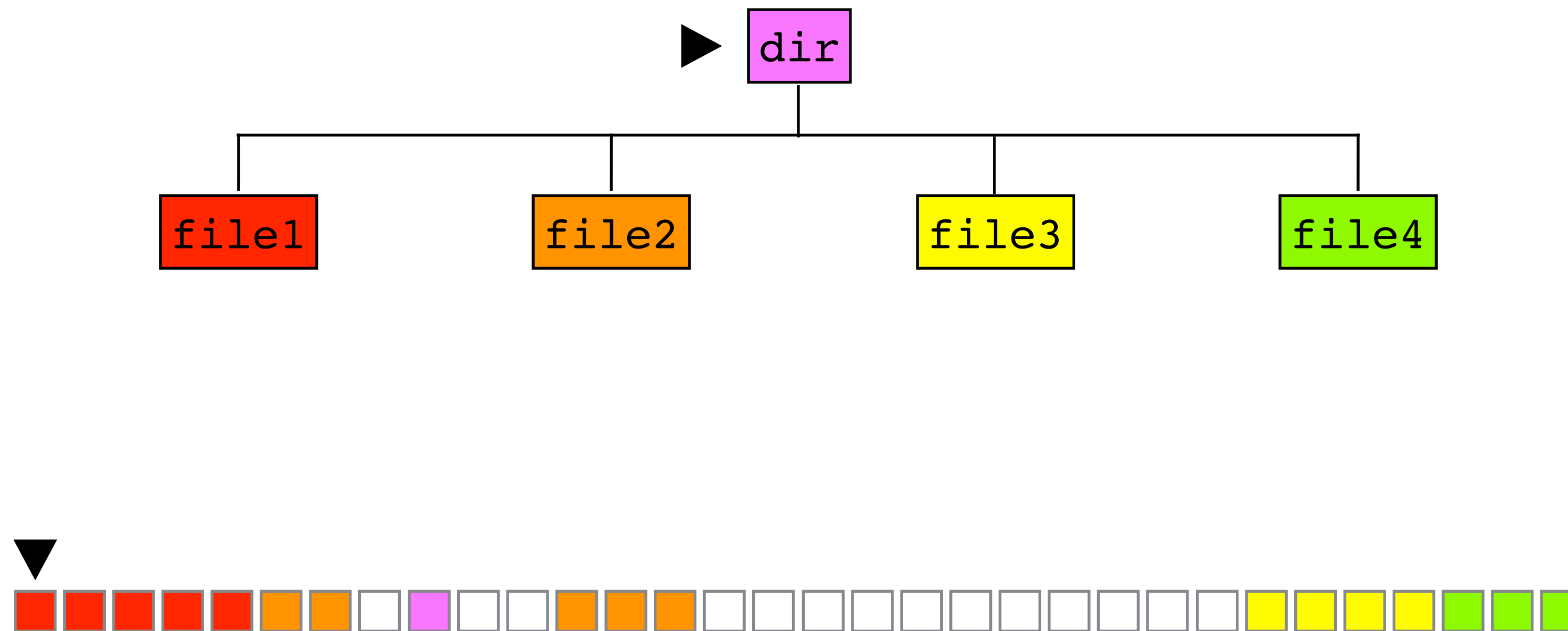
Simulating a Developer

Use the Linux kernel repo from github.com



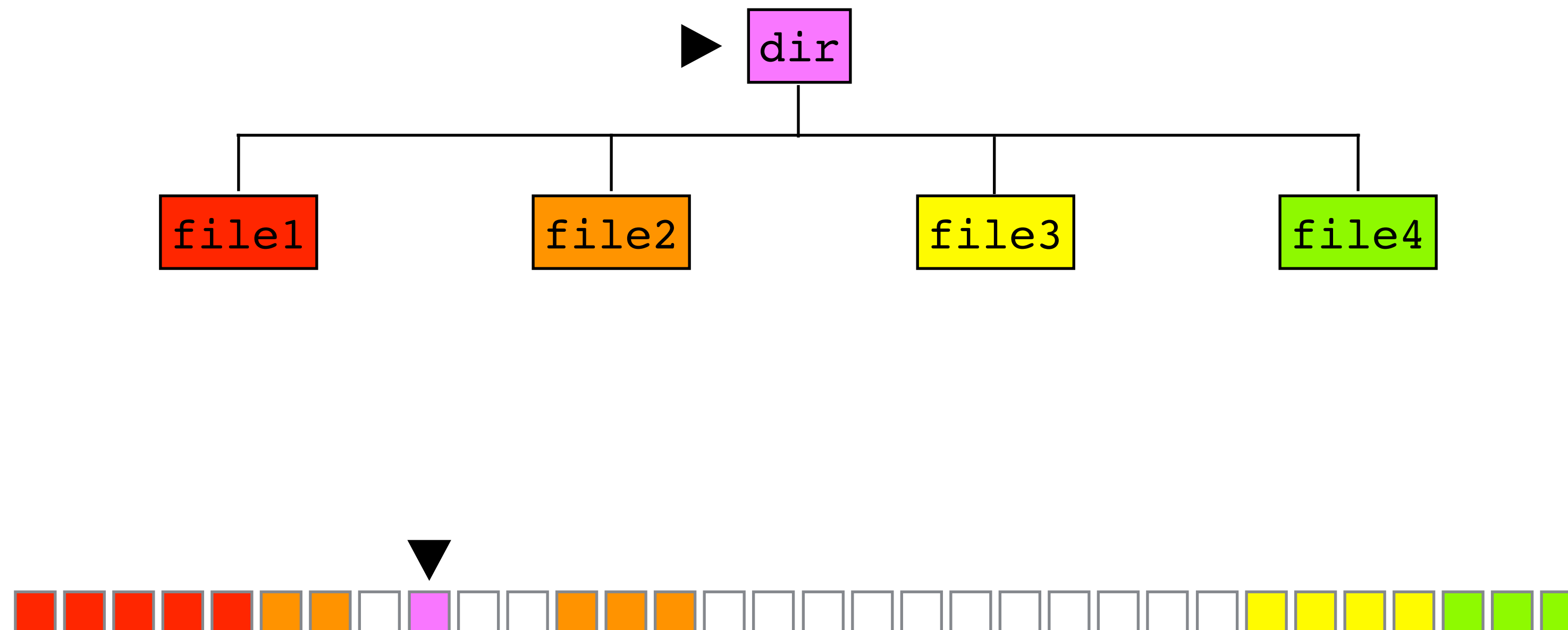
Measuring Aging

```
time grep -r random_string /path/to/filesystem
```



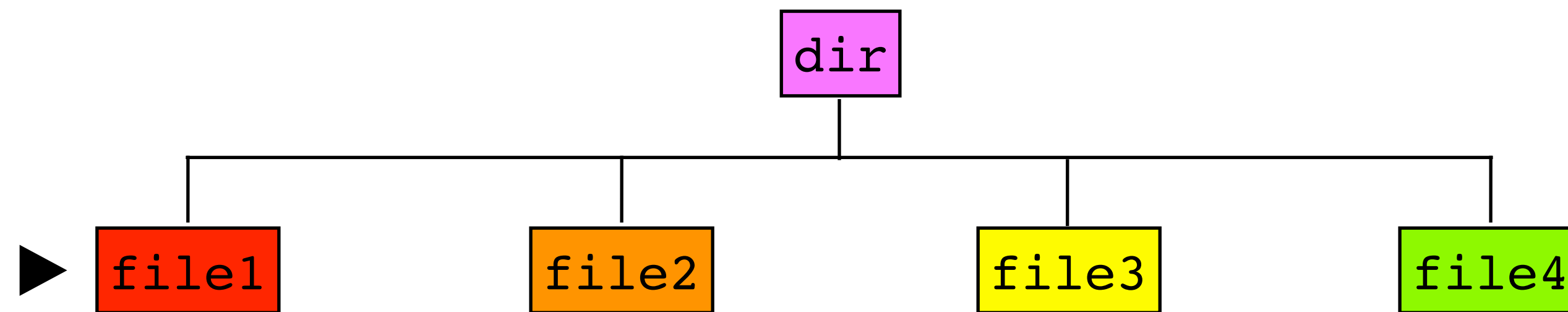
Measuring Aging

```
time grep -r random_string /path/to/filesystem
```



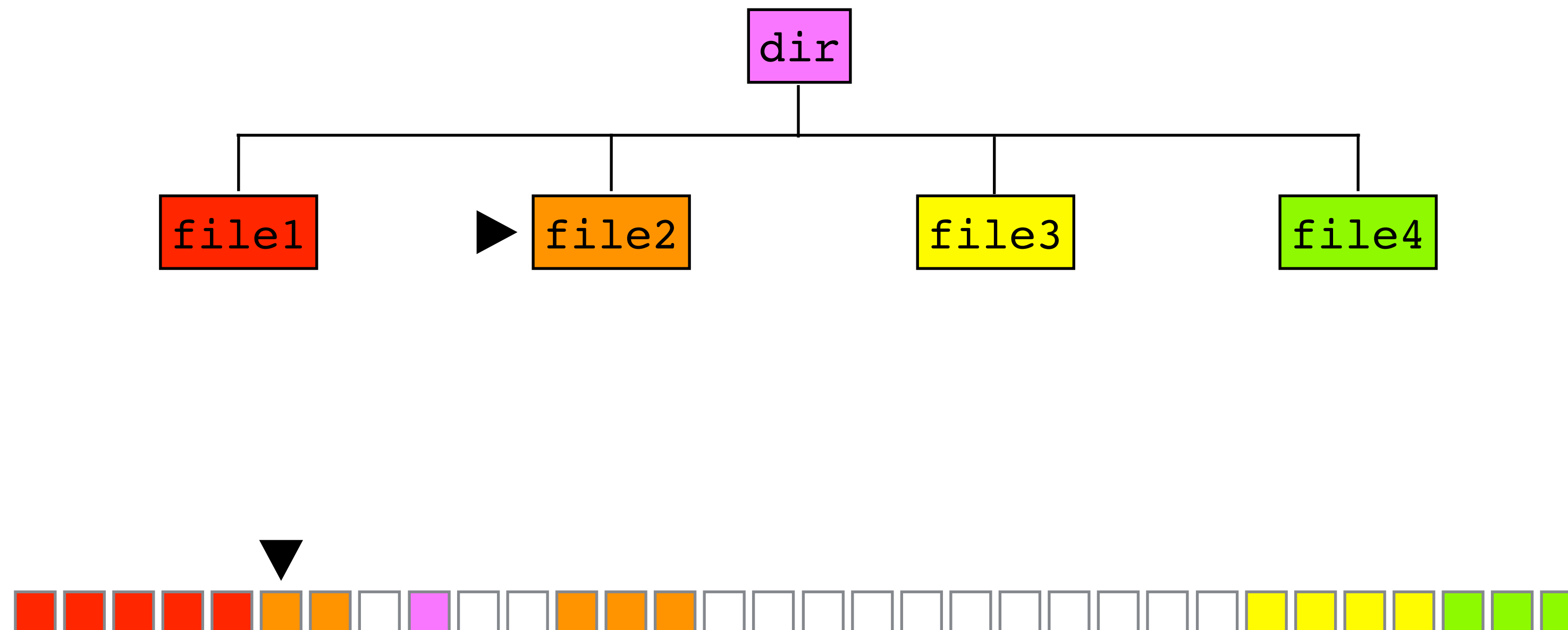
Measuring Aging

```
time grep -r random_string /path/to/filesystem
```



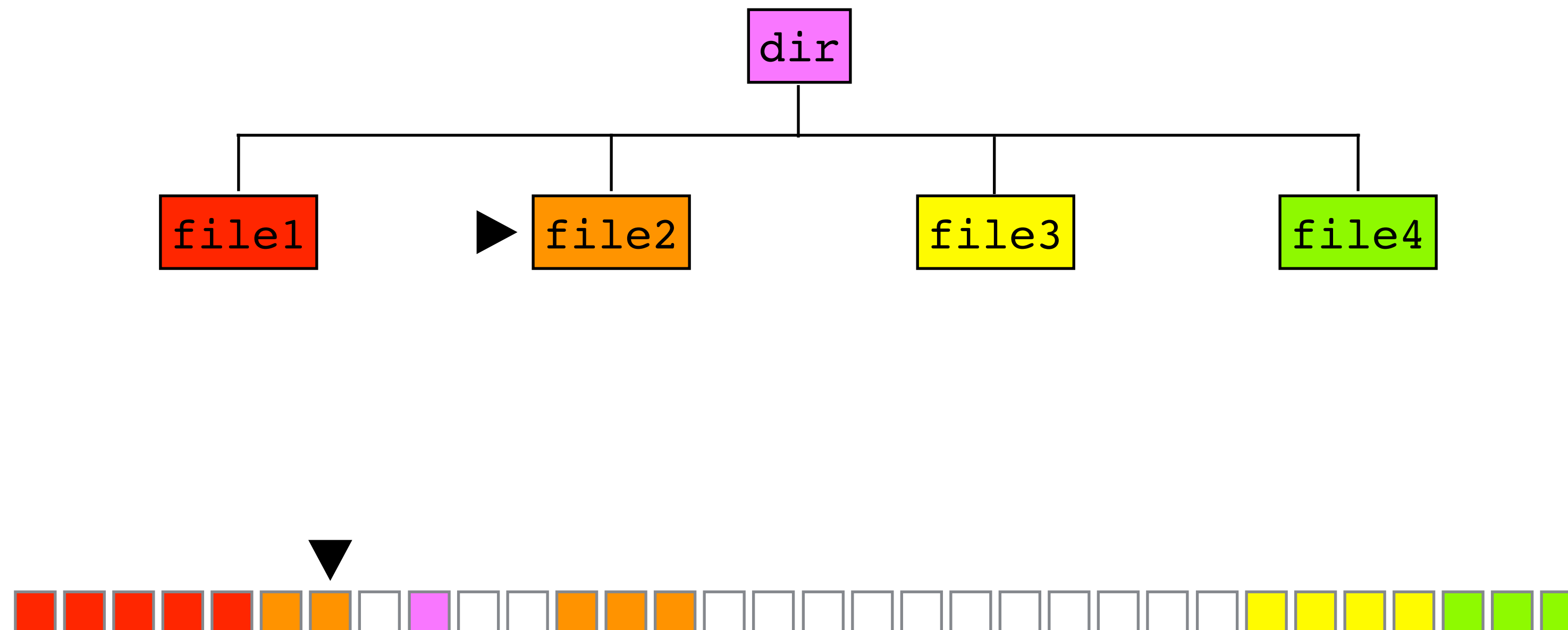
Measuring Aging

```
time grep -r random_string /path/to/filesystem
```



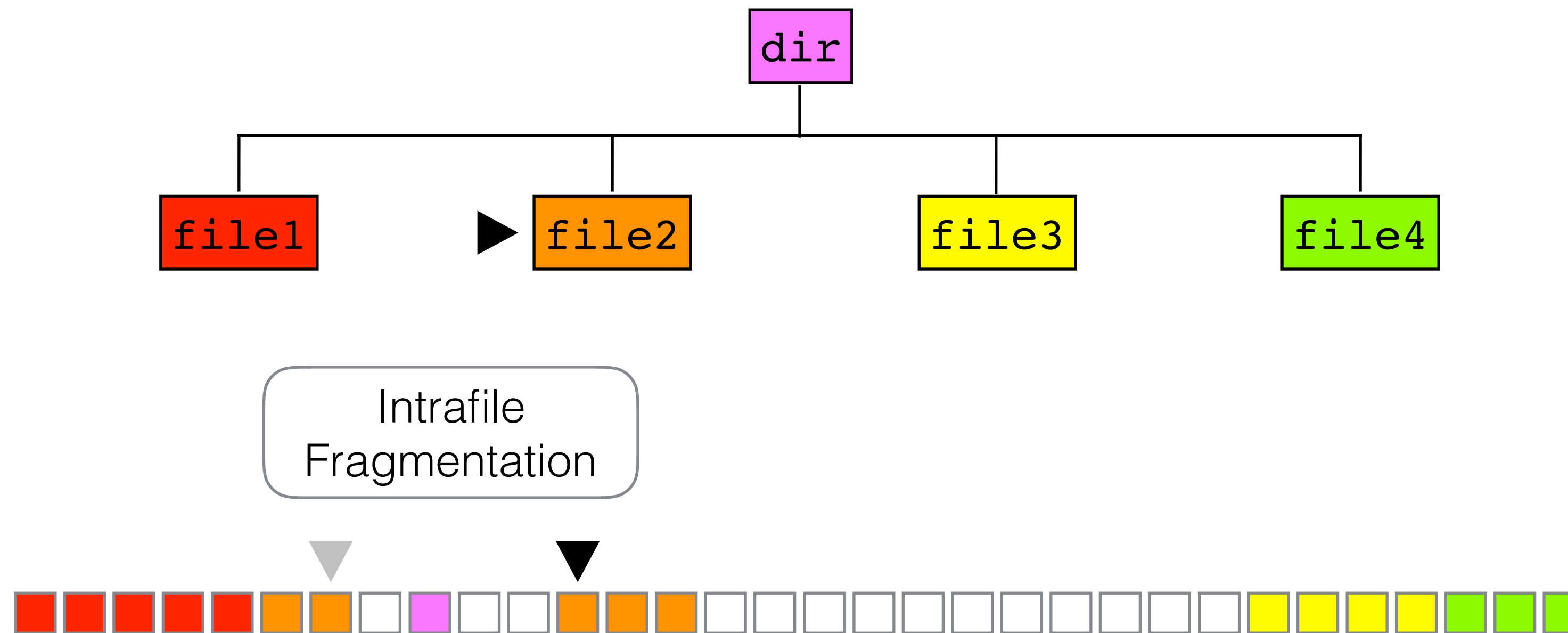
Measuring Aging

```
time grep -r random_string /path/to/filesystem
```



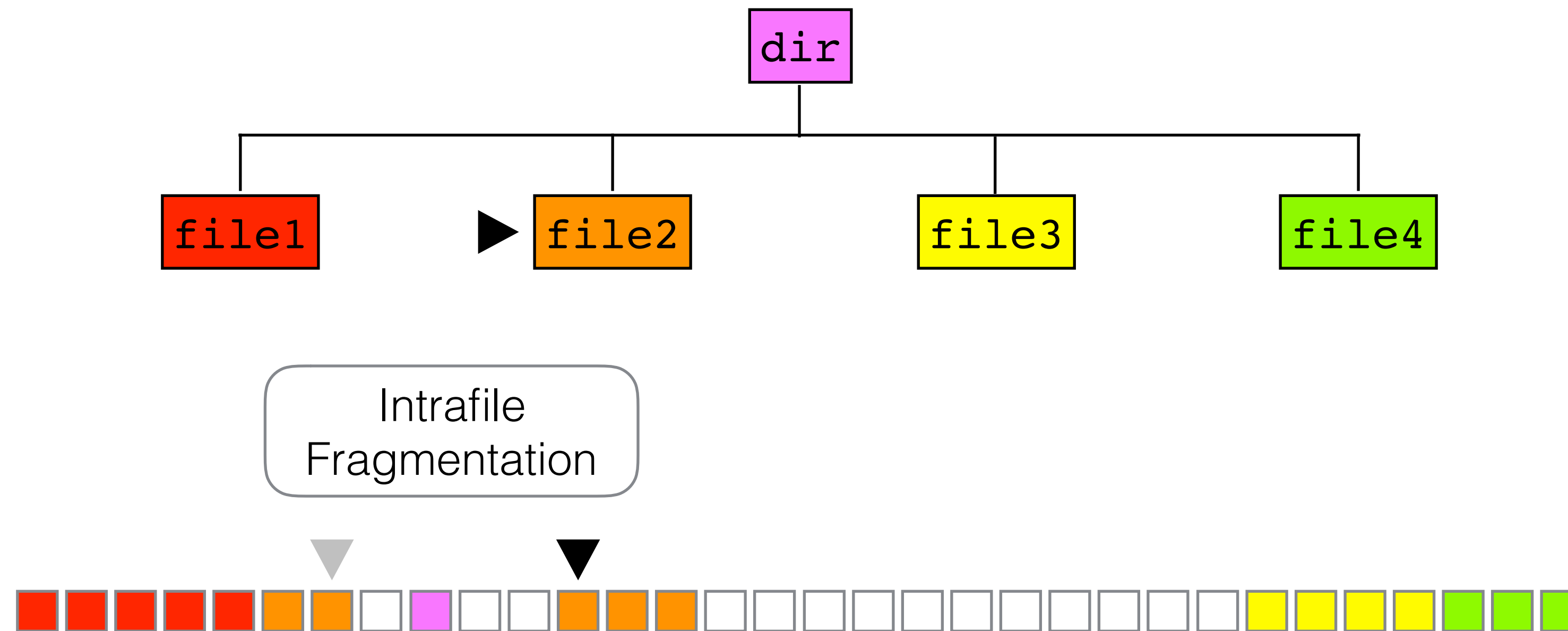
Measuring Aging

```
time grep -r random_string /path/to/filesystem
```



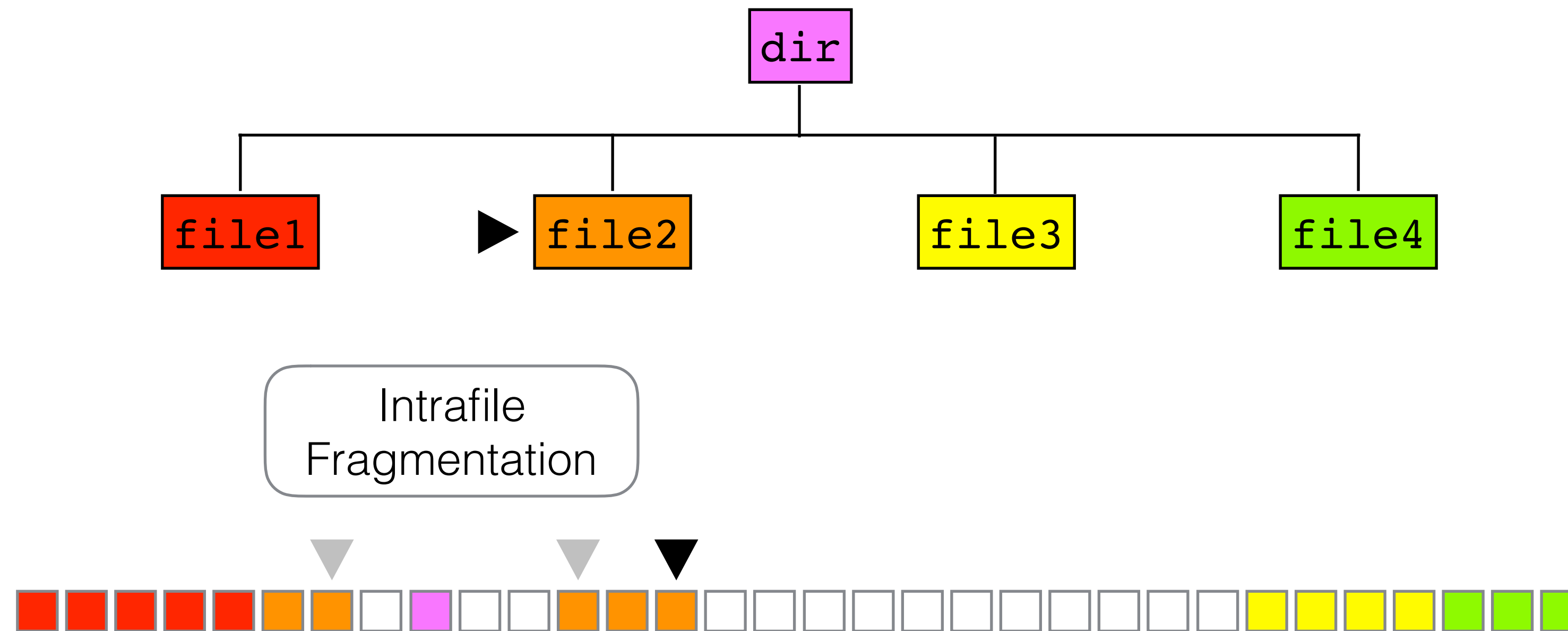
Measuring Aging

```
time grep -r random_string /path/to/filesystem
```



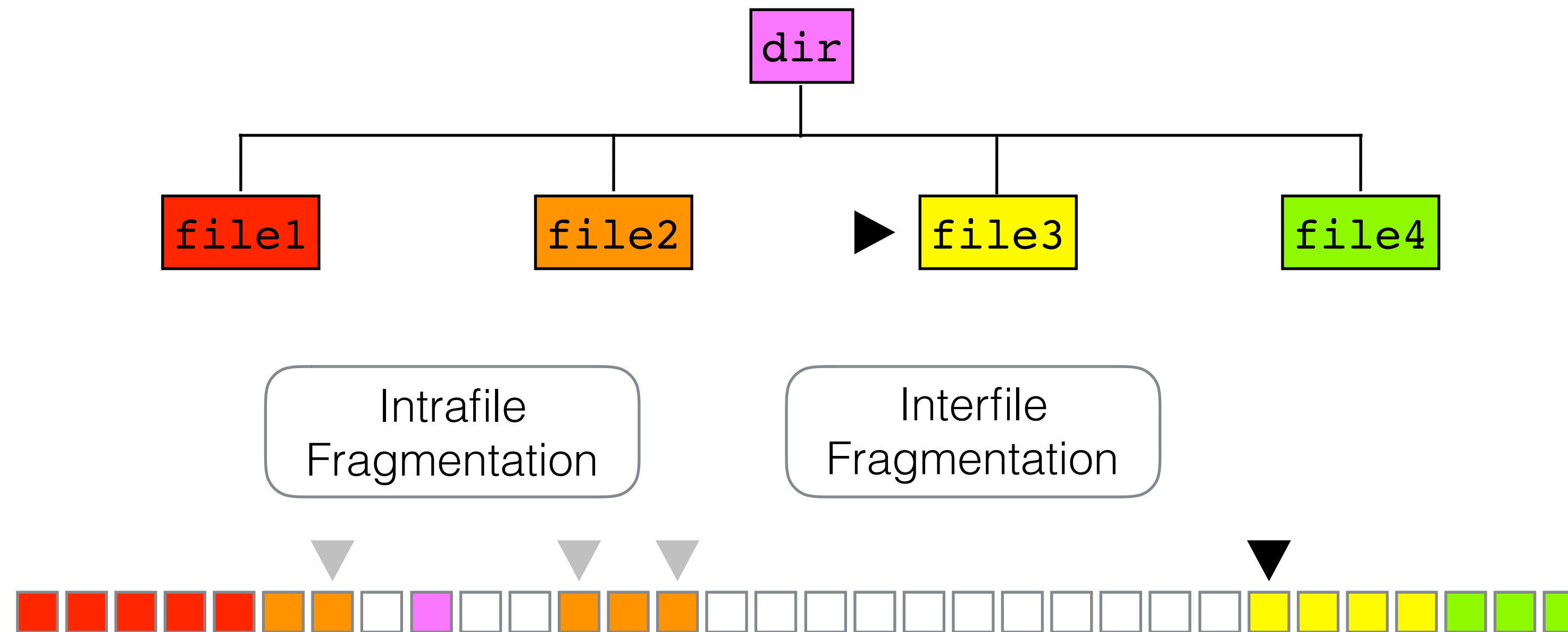
Measuring Aging

```
time grep -r random_string /path/to/filesystem
```



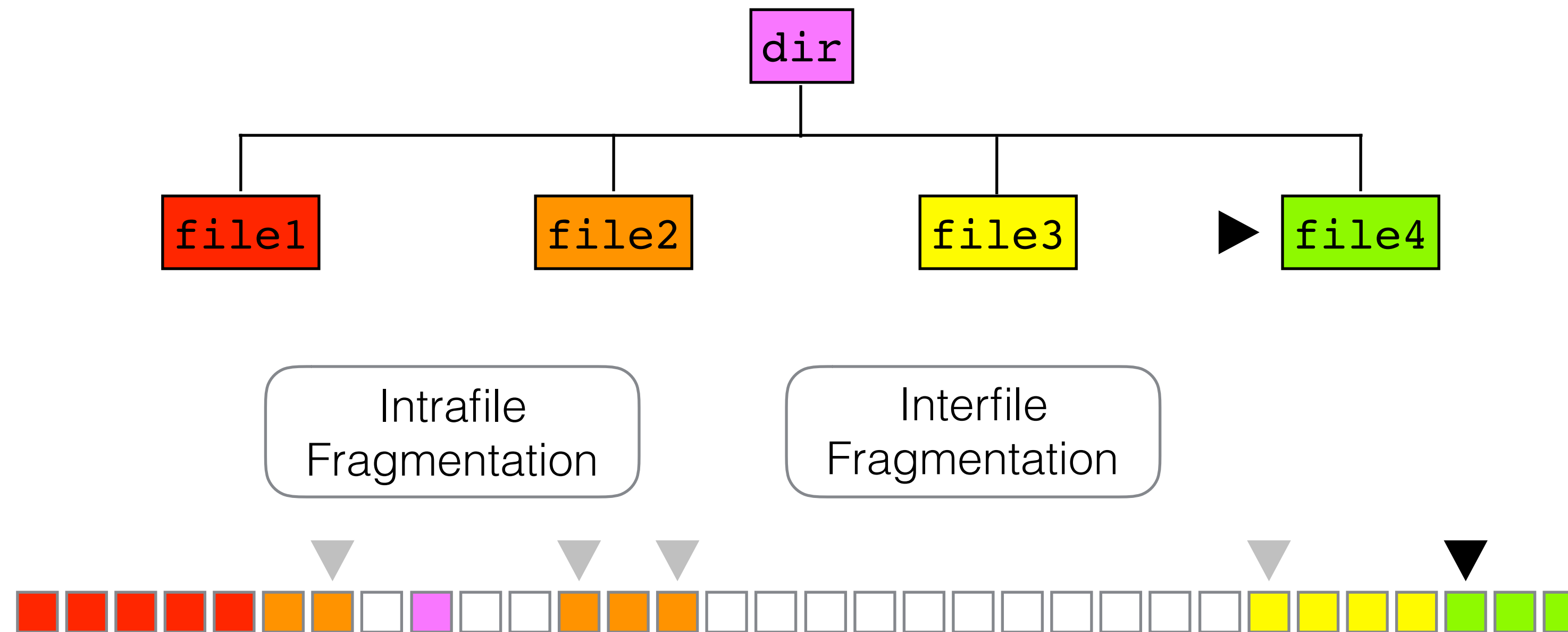
Measuring Aging

```
time grep -r random_string /path/to/filesystem
```



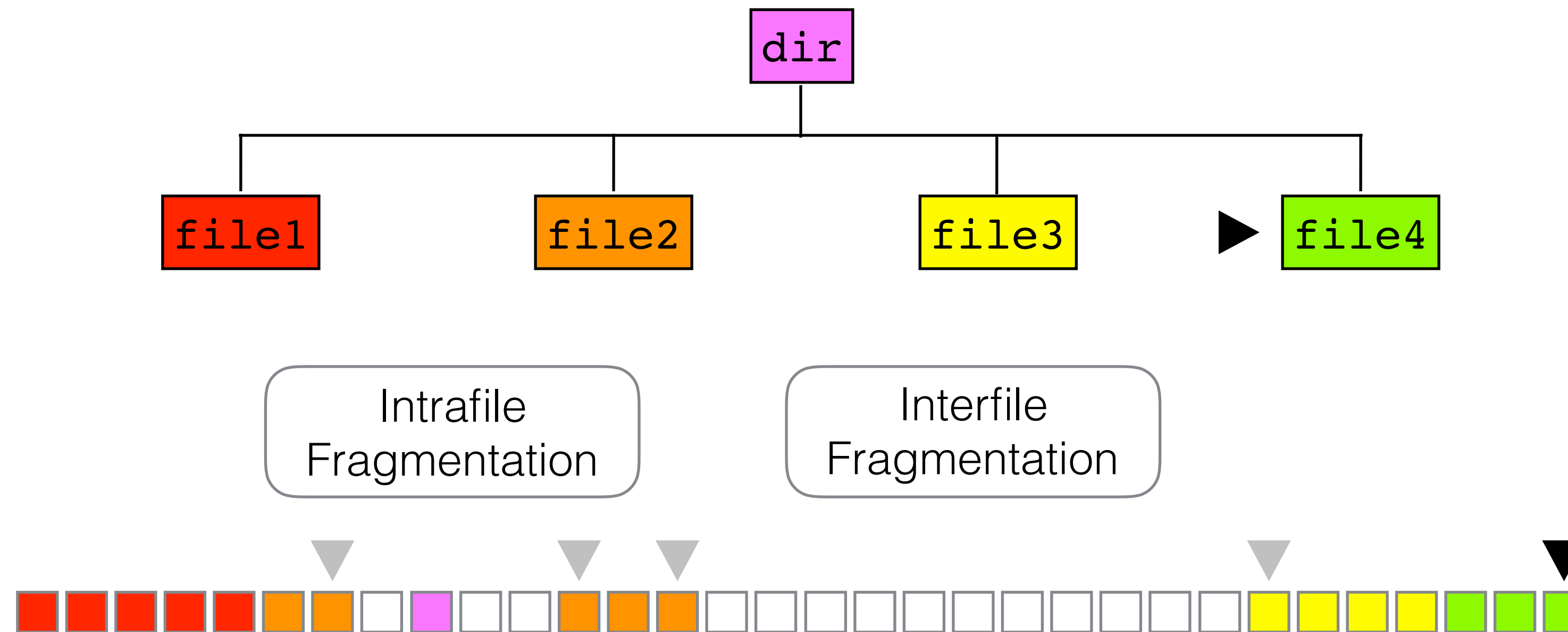
Measuring Aging

```
time grep -r random_string /path/to/filesystem
```



Measuring Aging

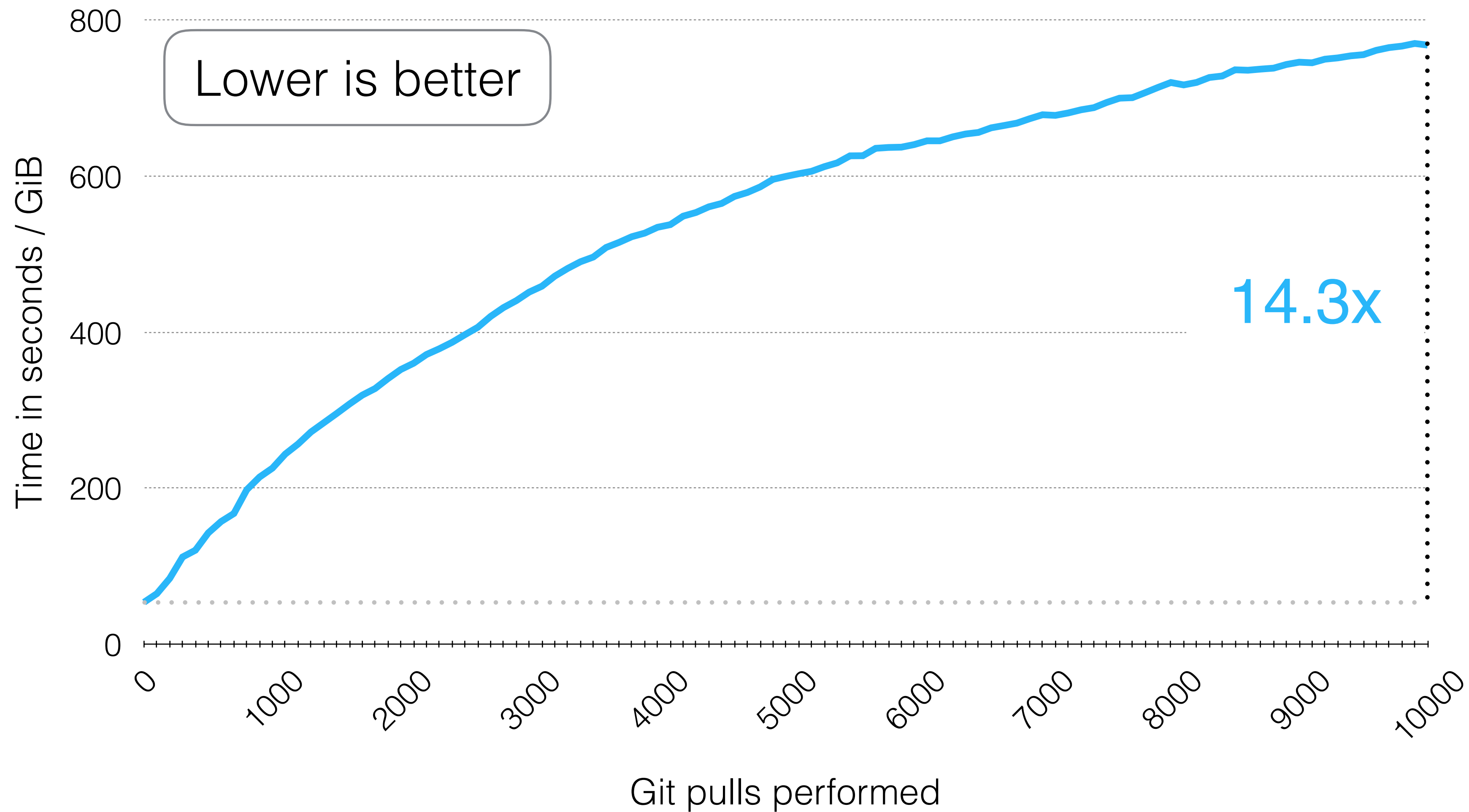
```
time grep -r random_string /path/to/filesystem
```



Then normalize per gigabyte read

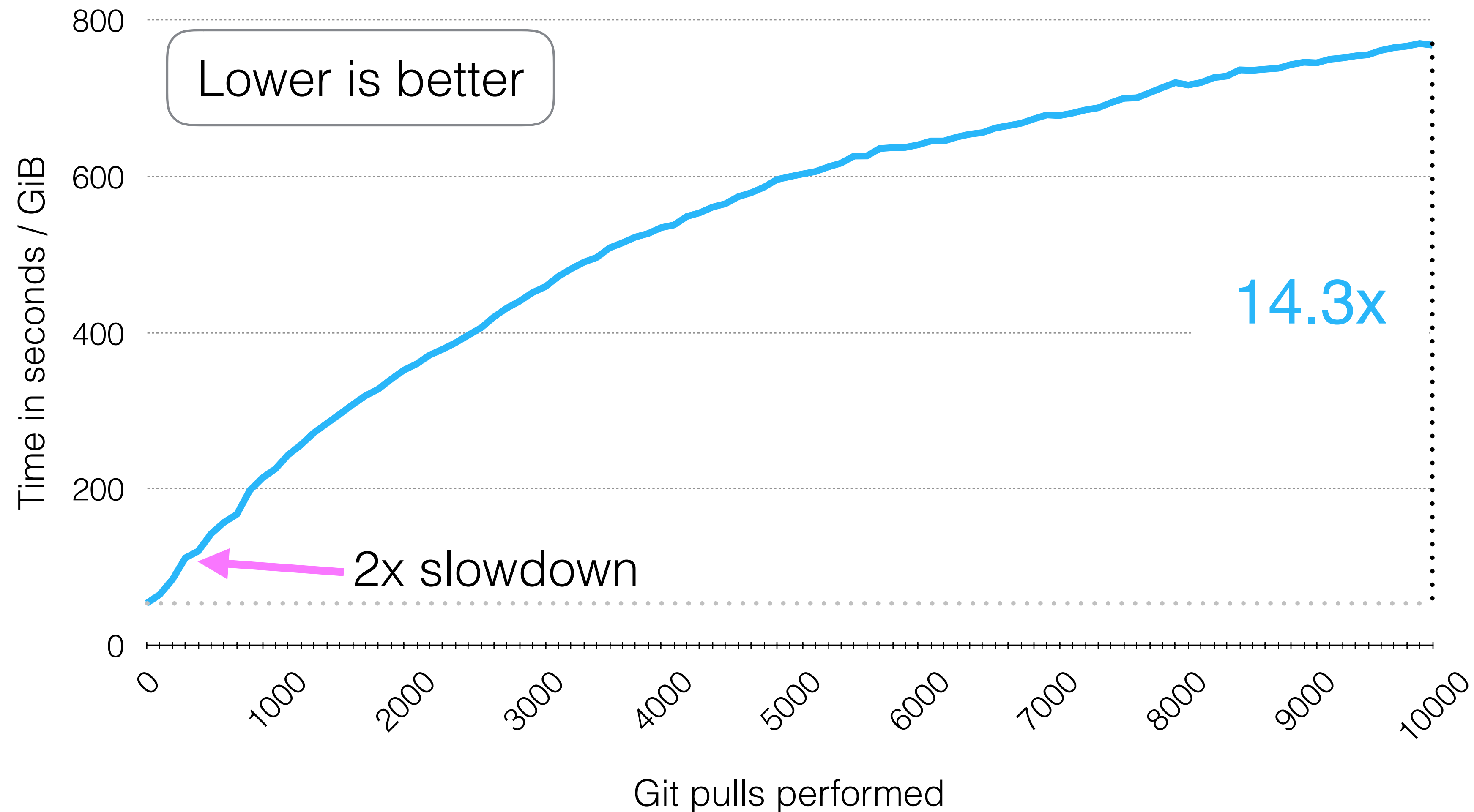
Do modern file
systems age?

Git Workload on ext4 on HDD



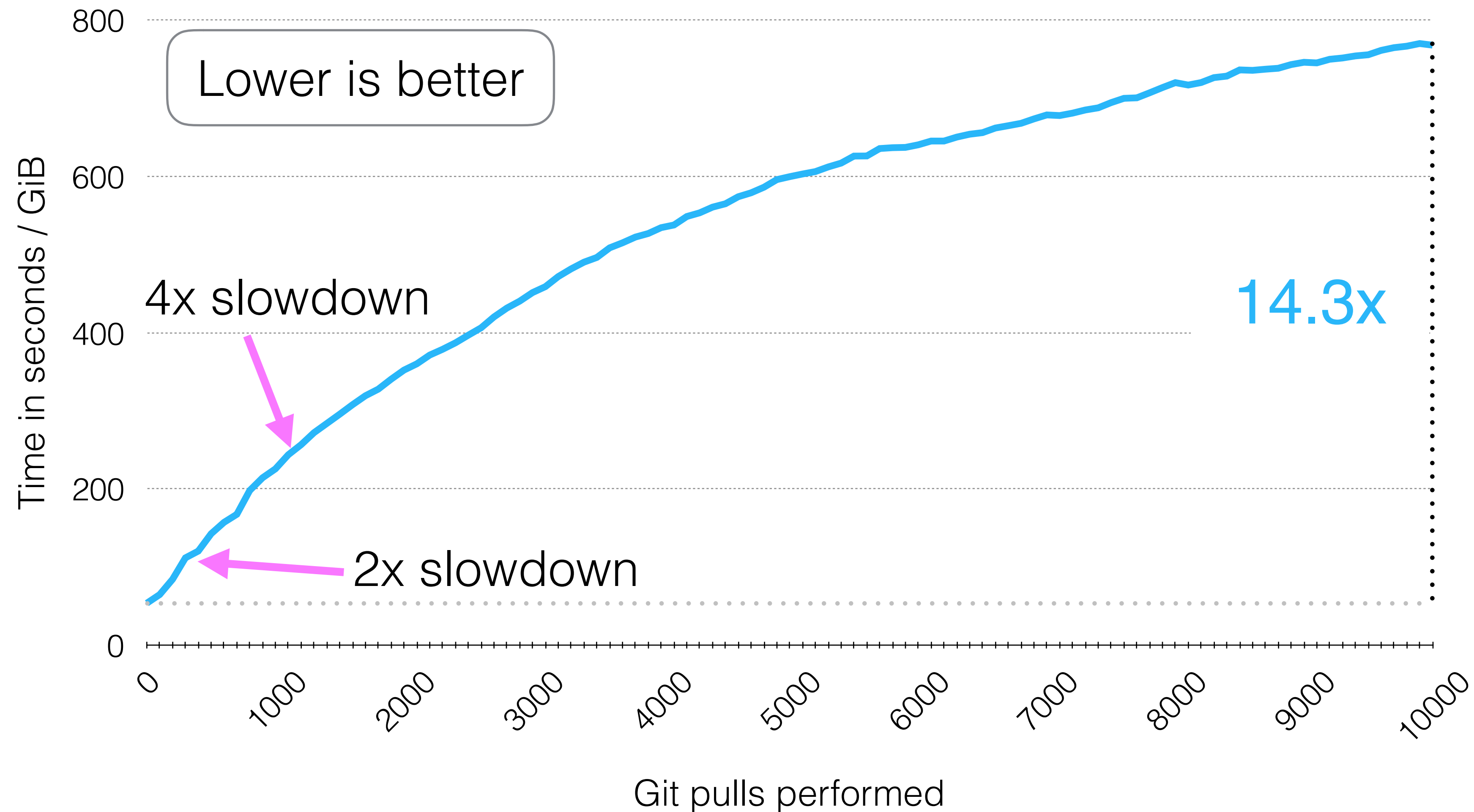
Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Git Workload on ext4 on HDD



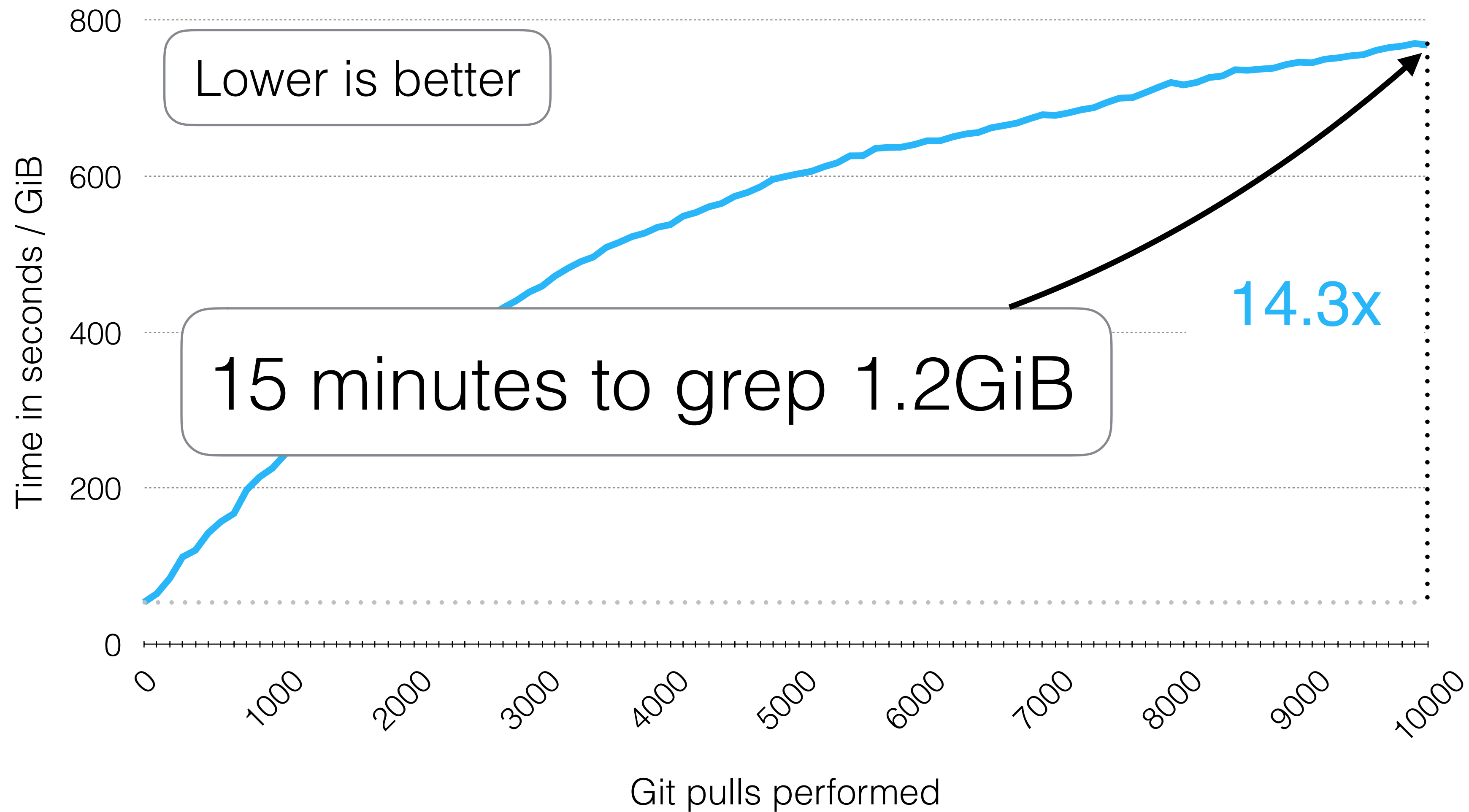
Our Setup: Cold Cache, 3.4 GHz Quad Core, 4GiB RAM,
20 GiB HDD partition - SATA 7200 RPM

Git Workload on ext4 on HDD



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

Git Workload on ext4 on 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
slowdown is due to aging?

How can we be sure this slowdown is due to aging?



I'm not old. My
directory structure
is different!

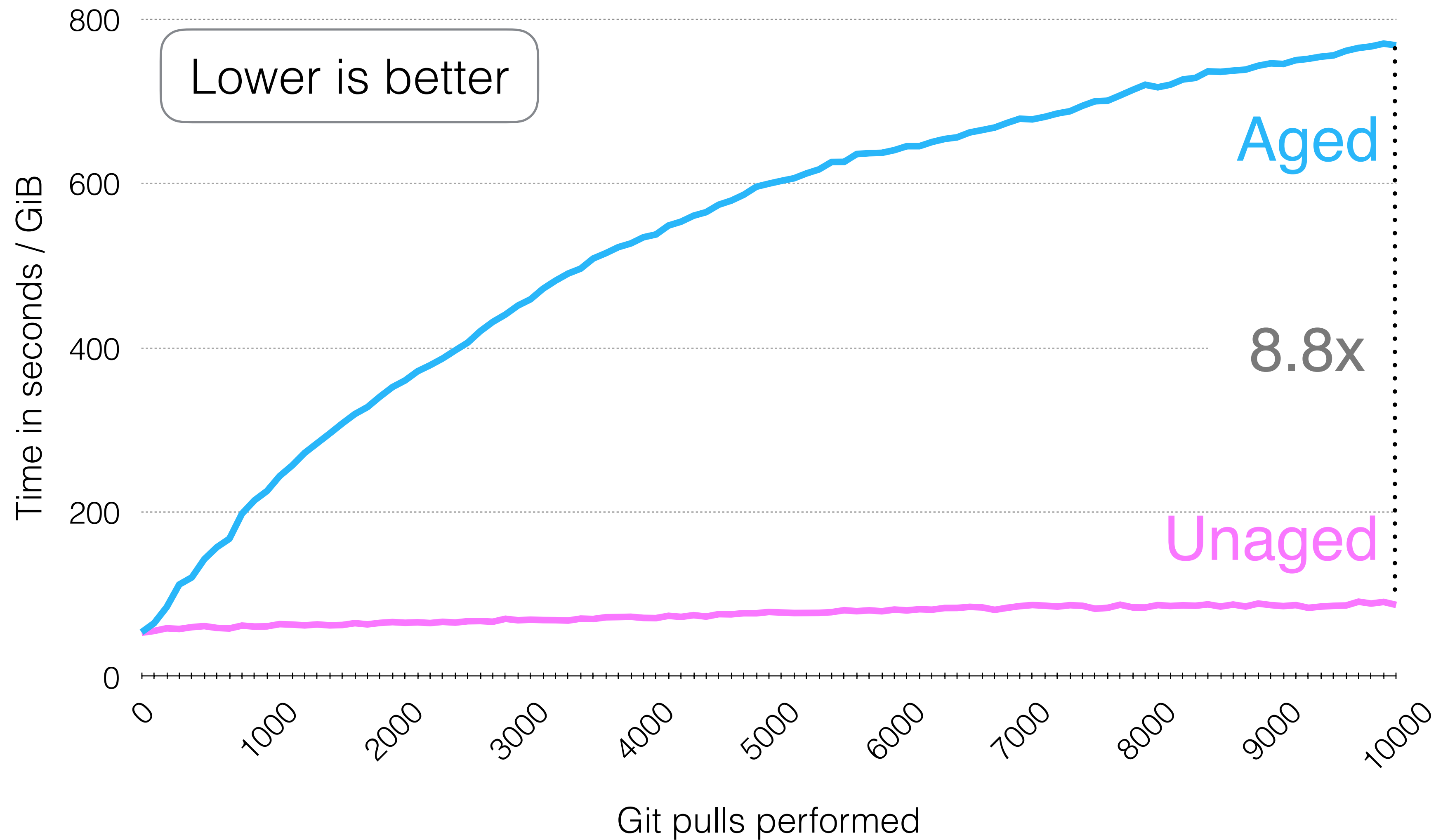
File System Rejuvenation

Idea: Copy same logical state to a new file system

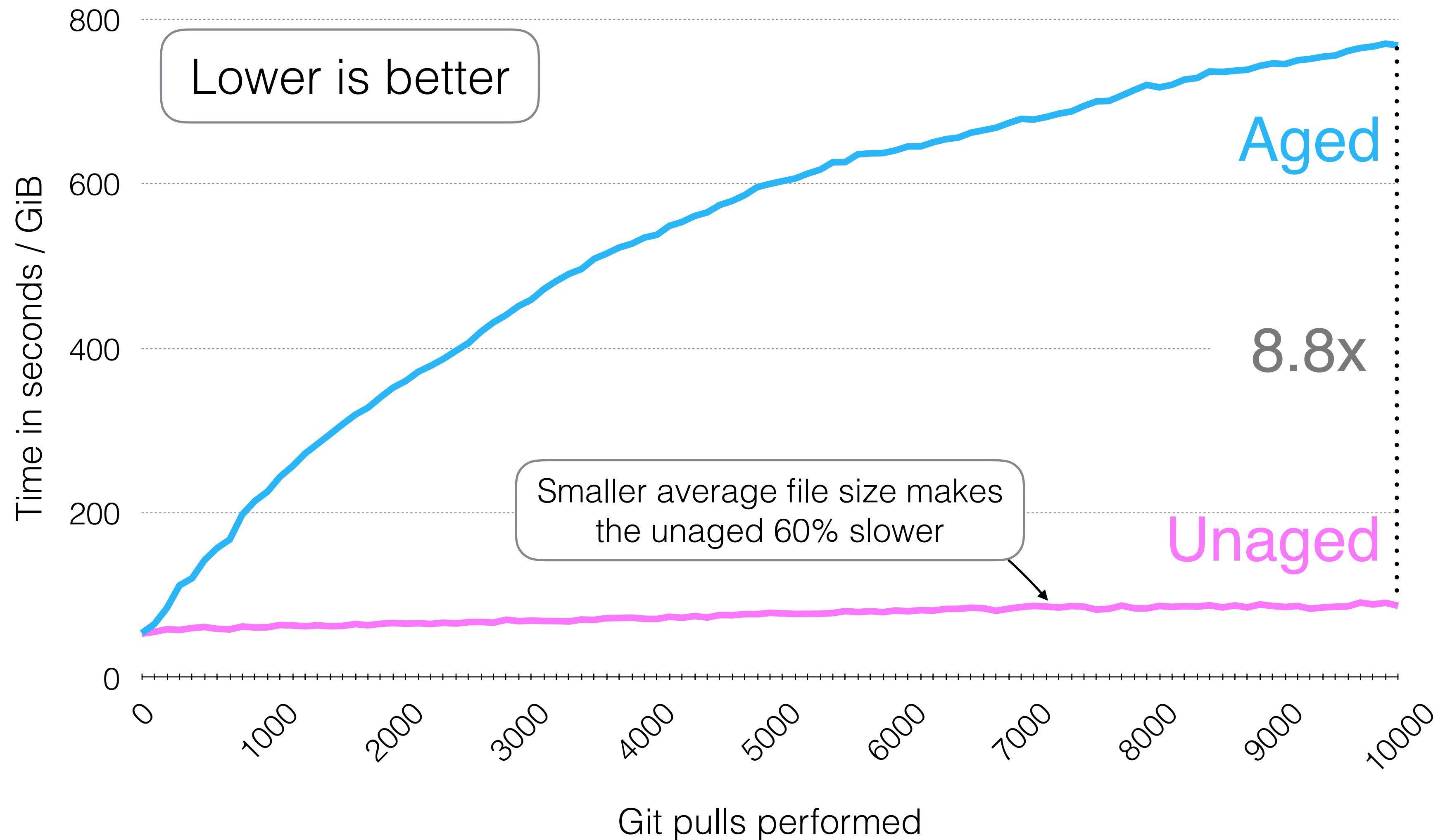
- After each 100 pulls
- Compare grep cost



Aging ext4 with Git on HDD



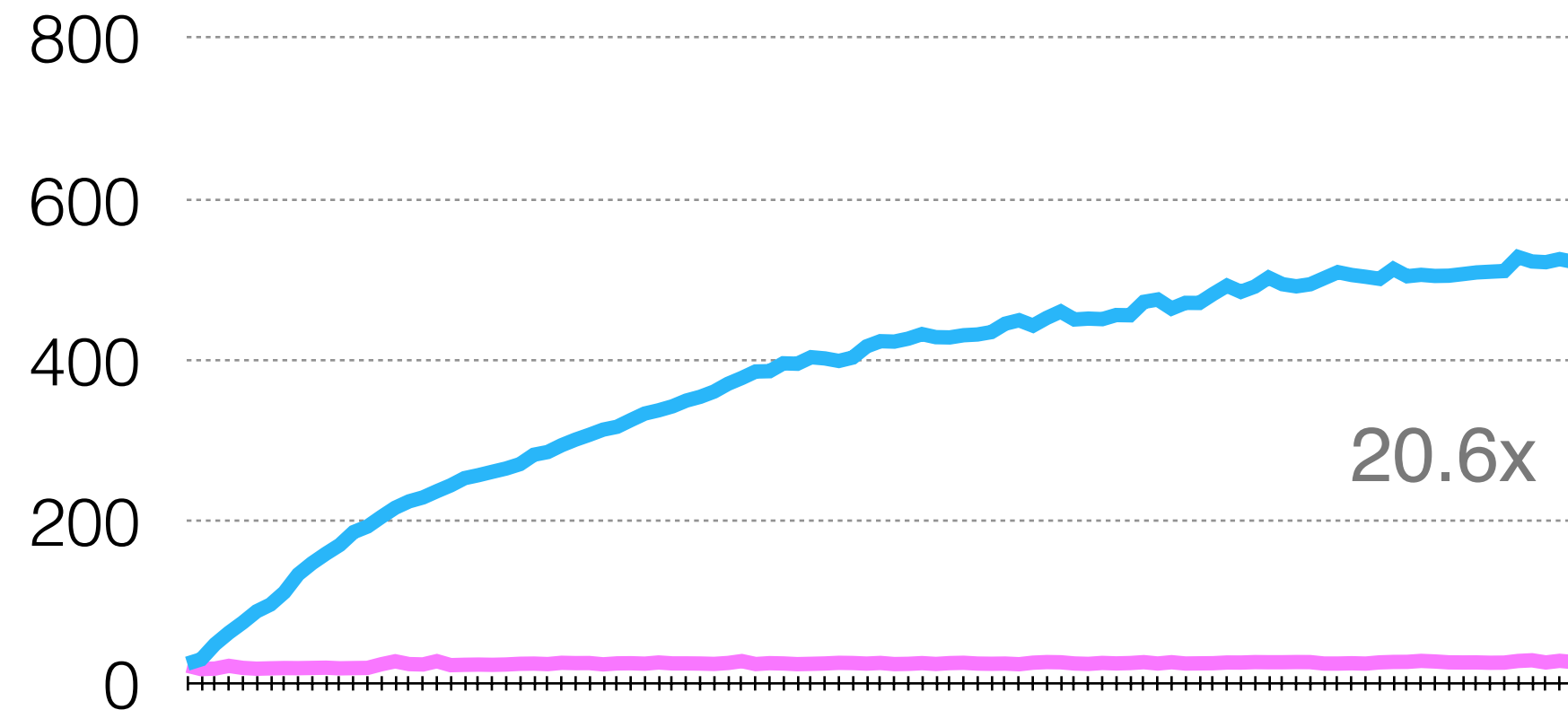
Aging ext4 with Git on HDD



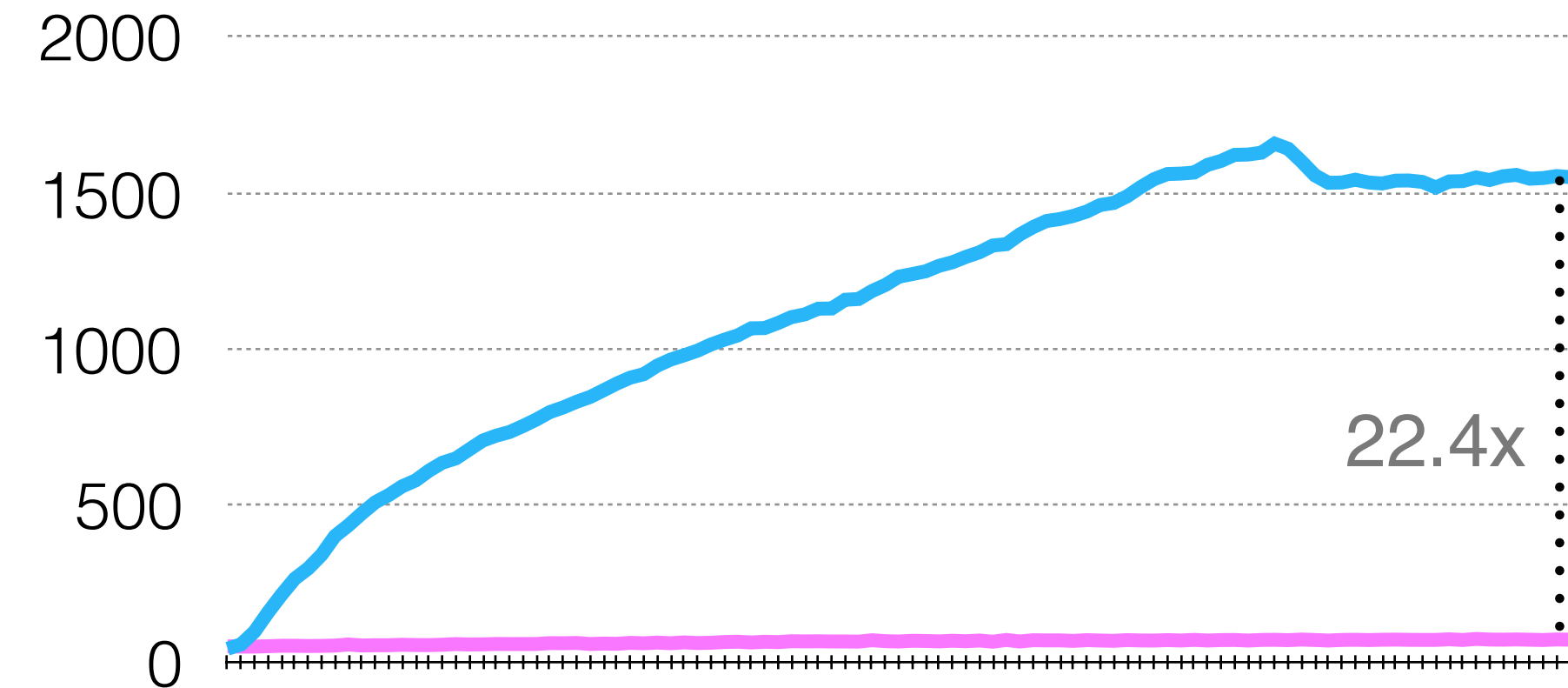
Is this specific to ext4?

Aging other file systems with Git on HDD

Btrfs

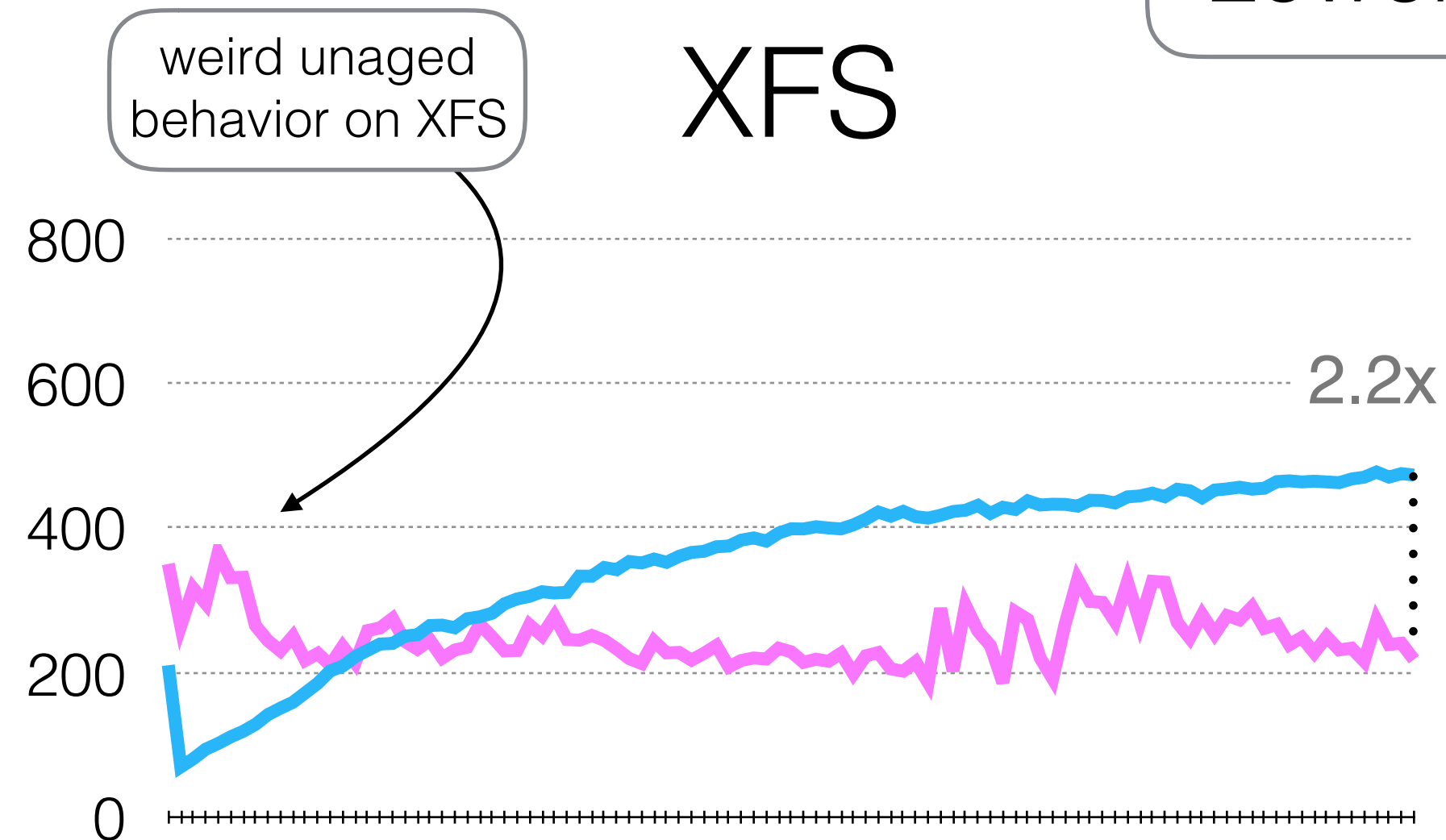


F2FS

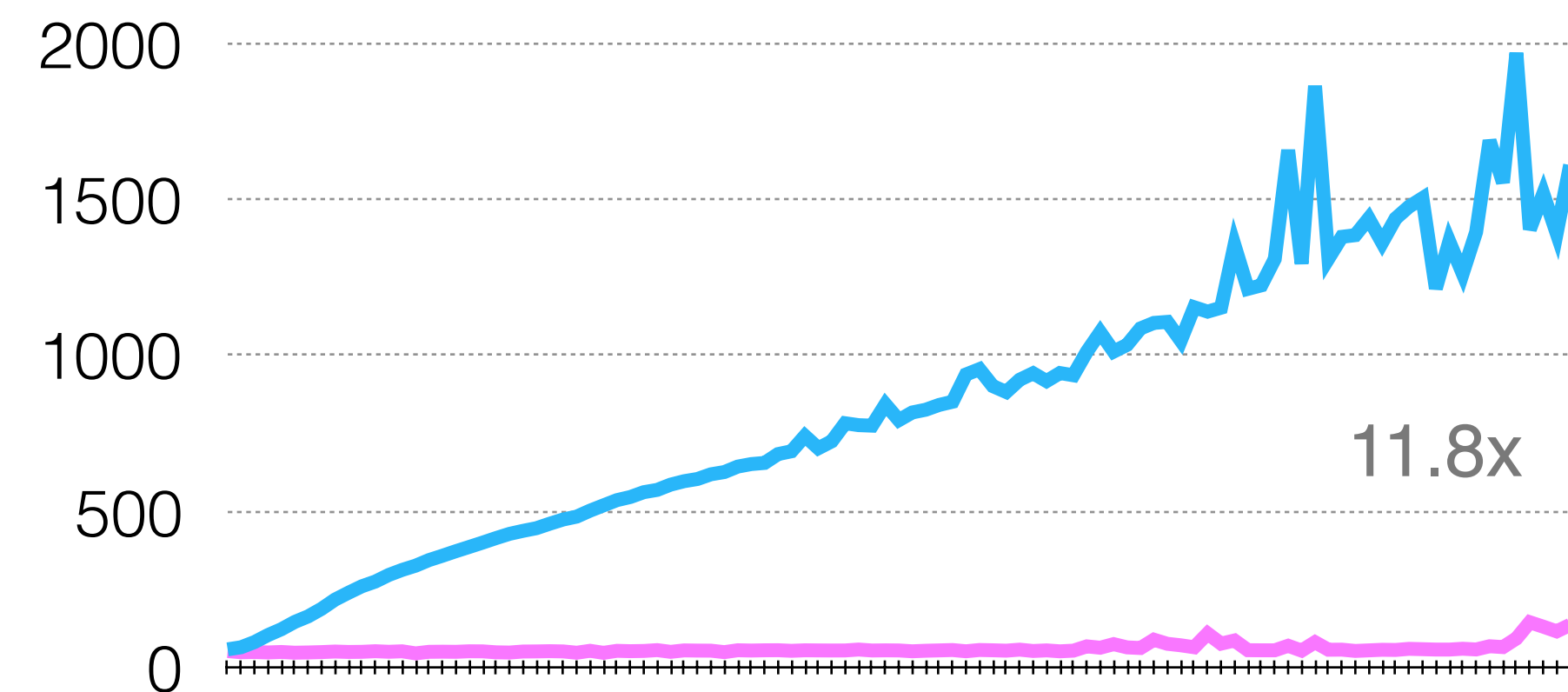


Lower is better

XFS



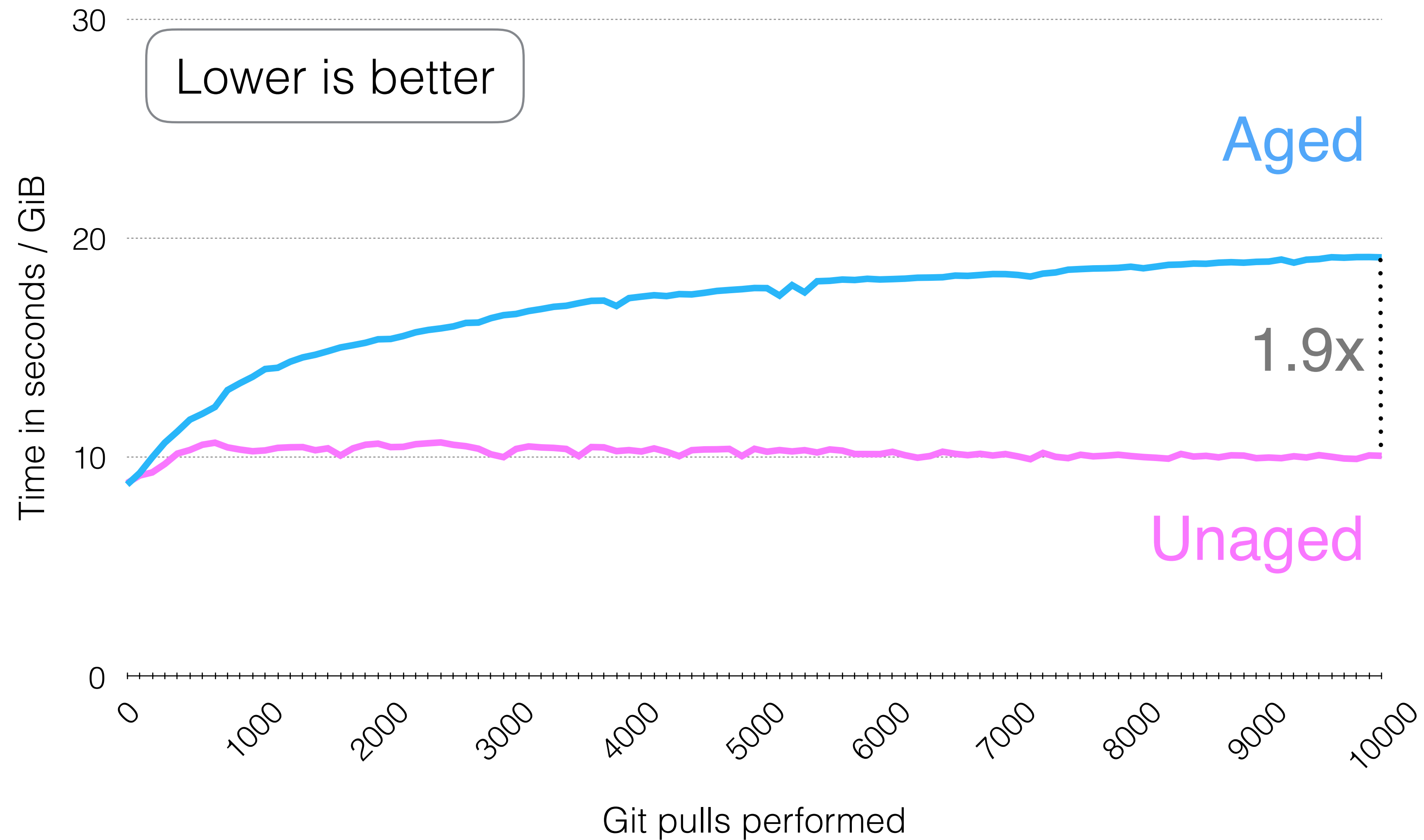
ZFS



Will SSDs save us?

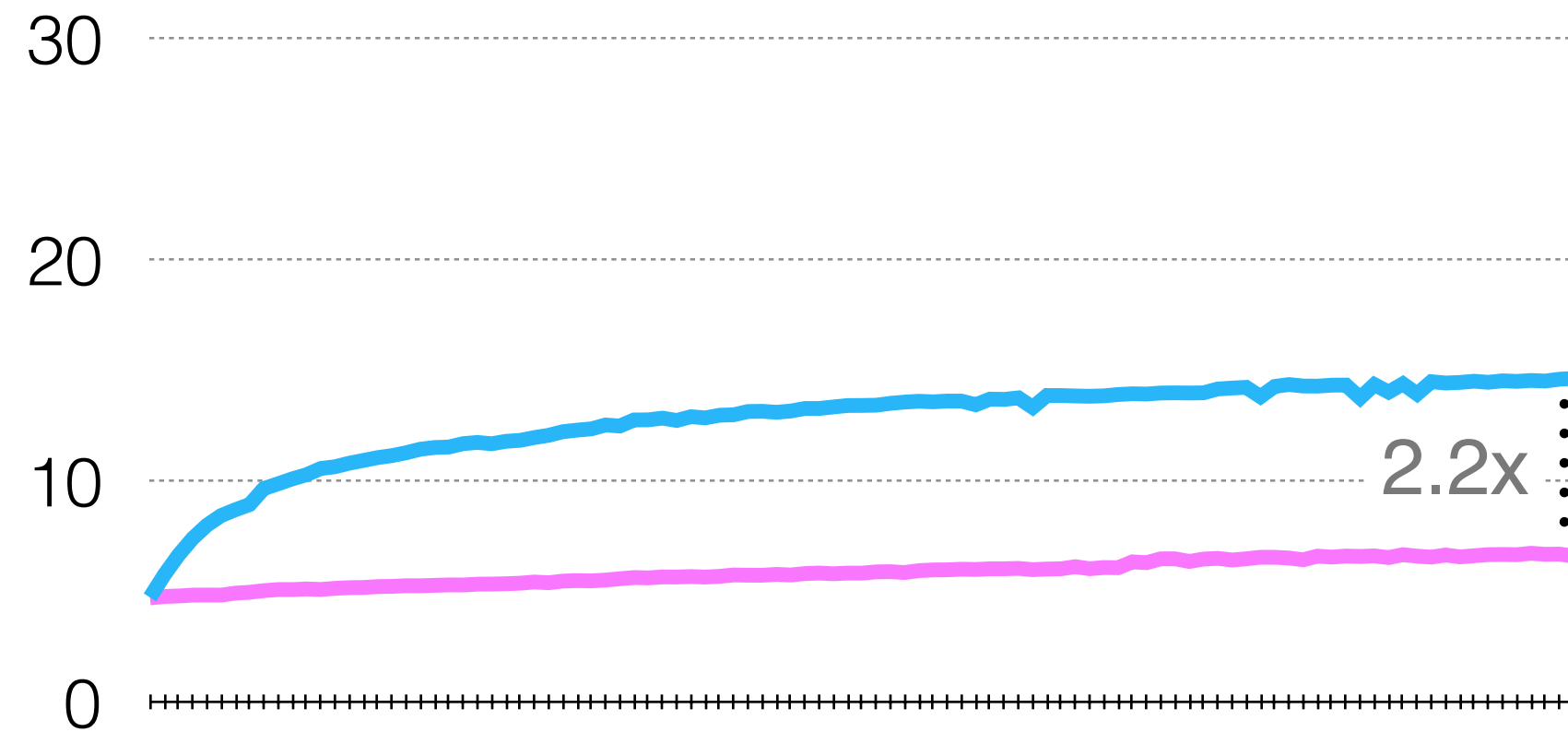


Git Workload on XFS on SSD

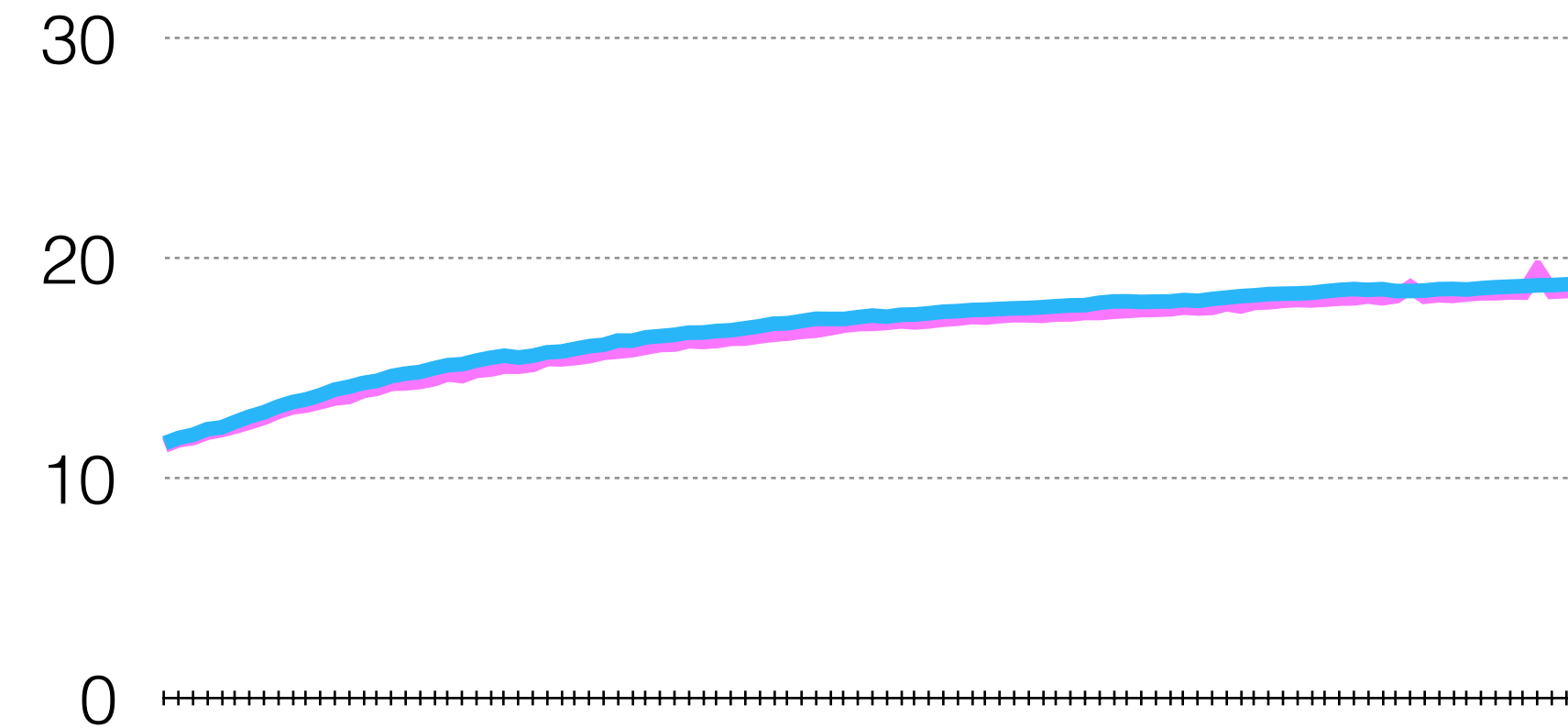


Git Workload on SSD

Btrfs

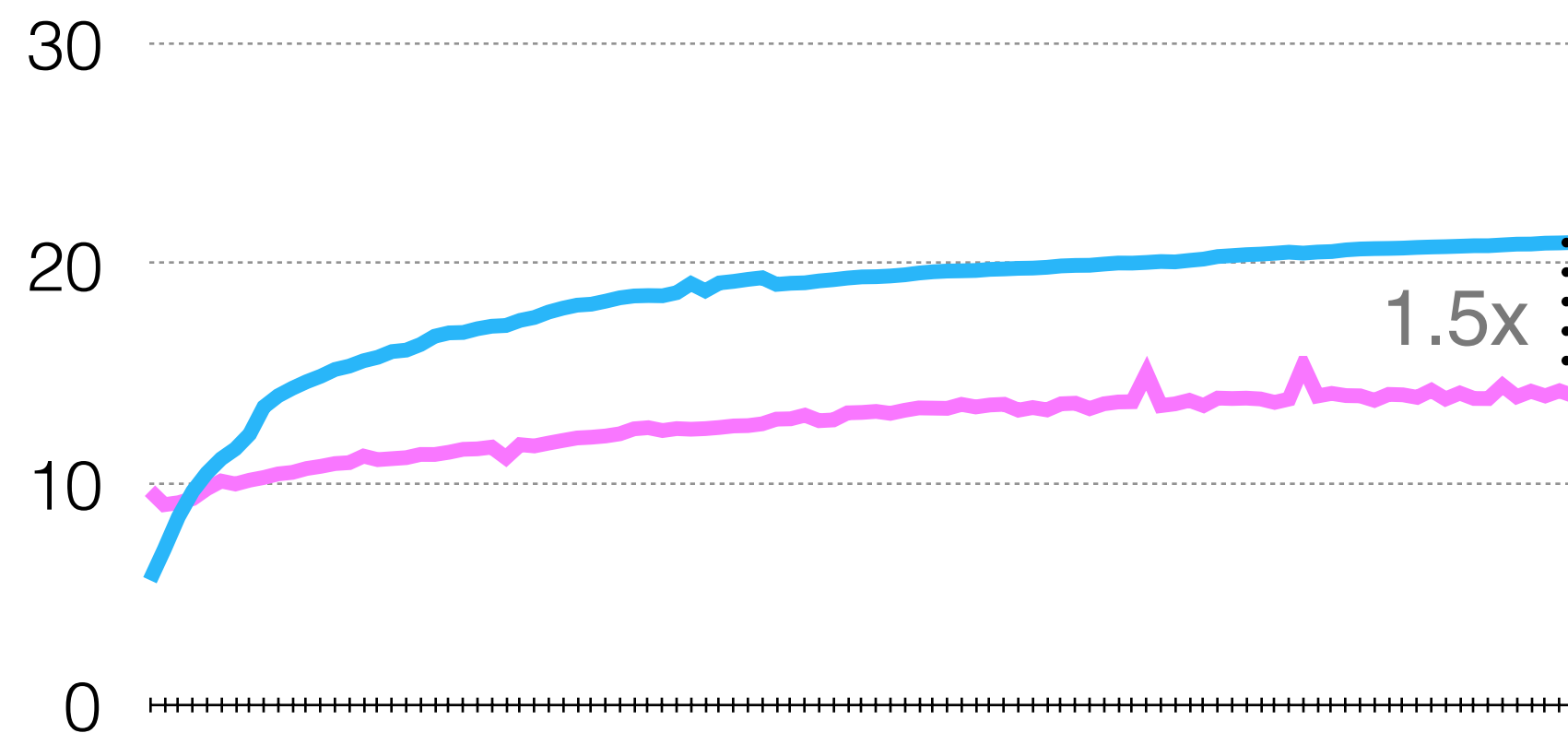


ext4

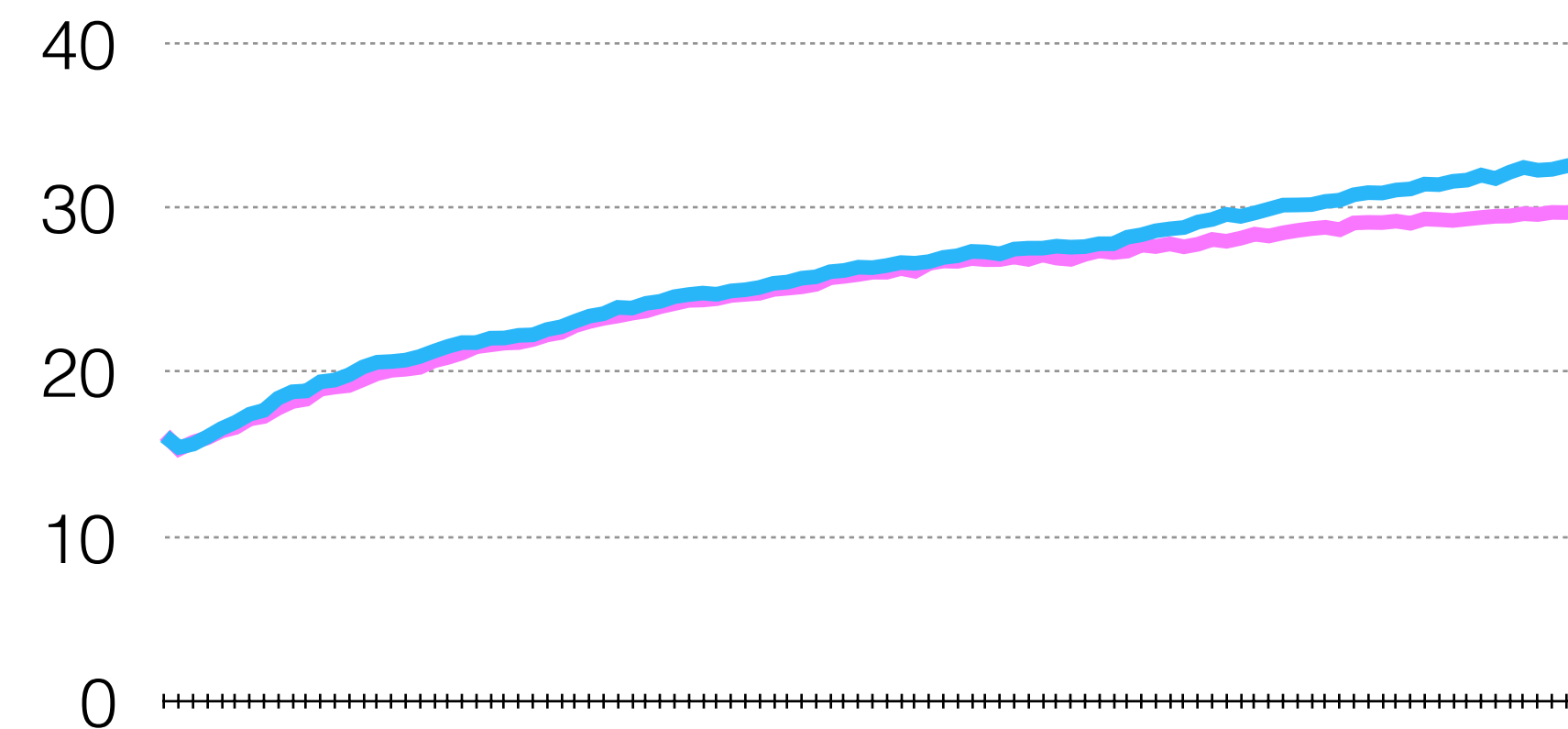


Lower is better

F2FS

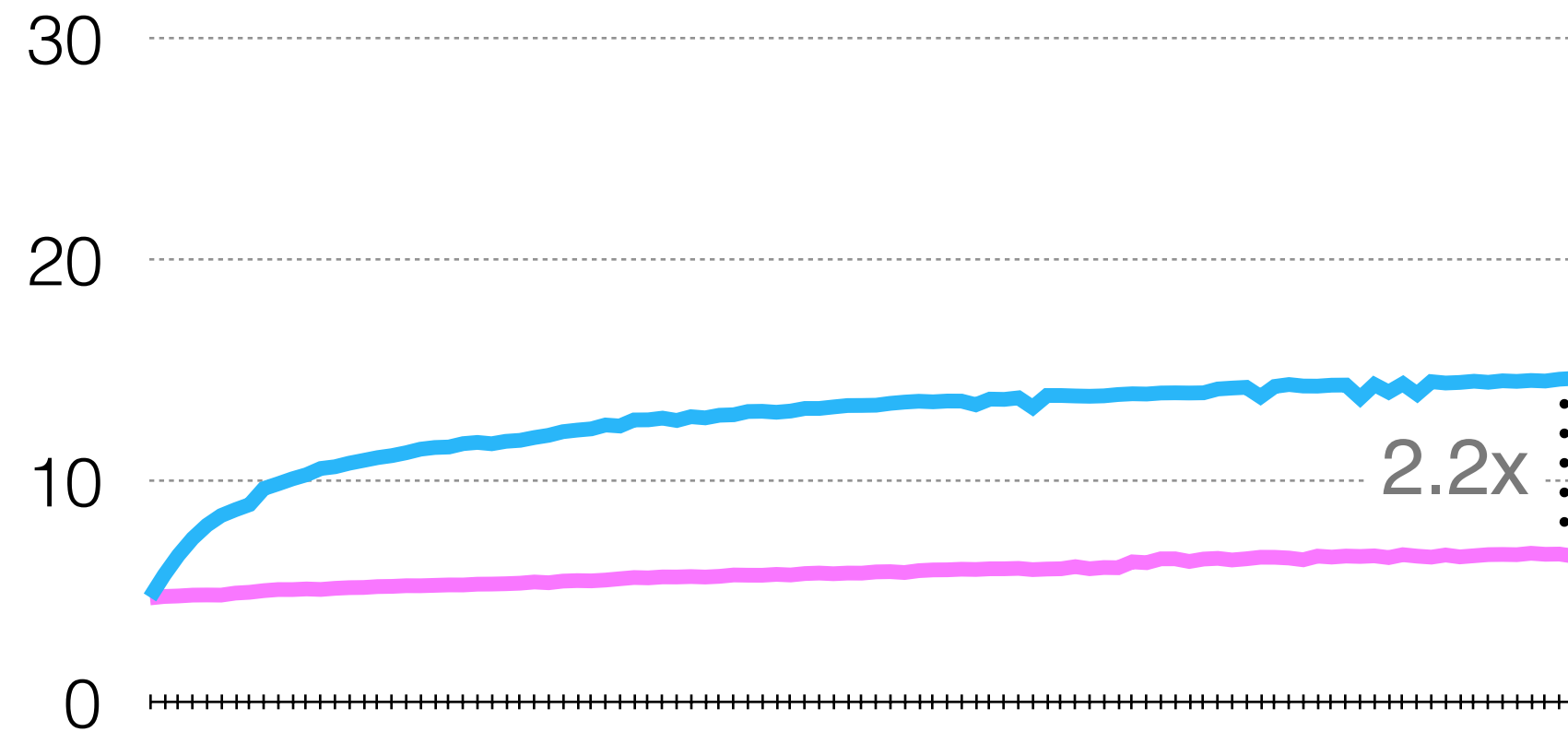


ZFS

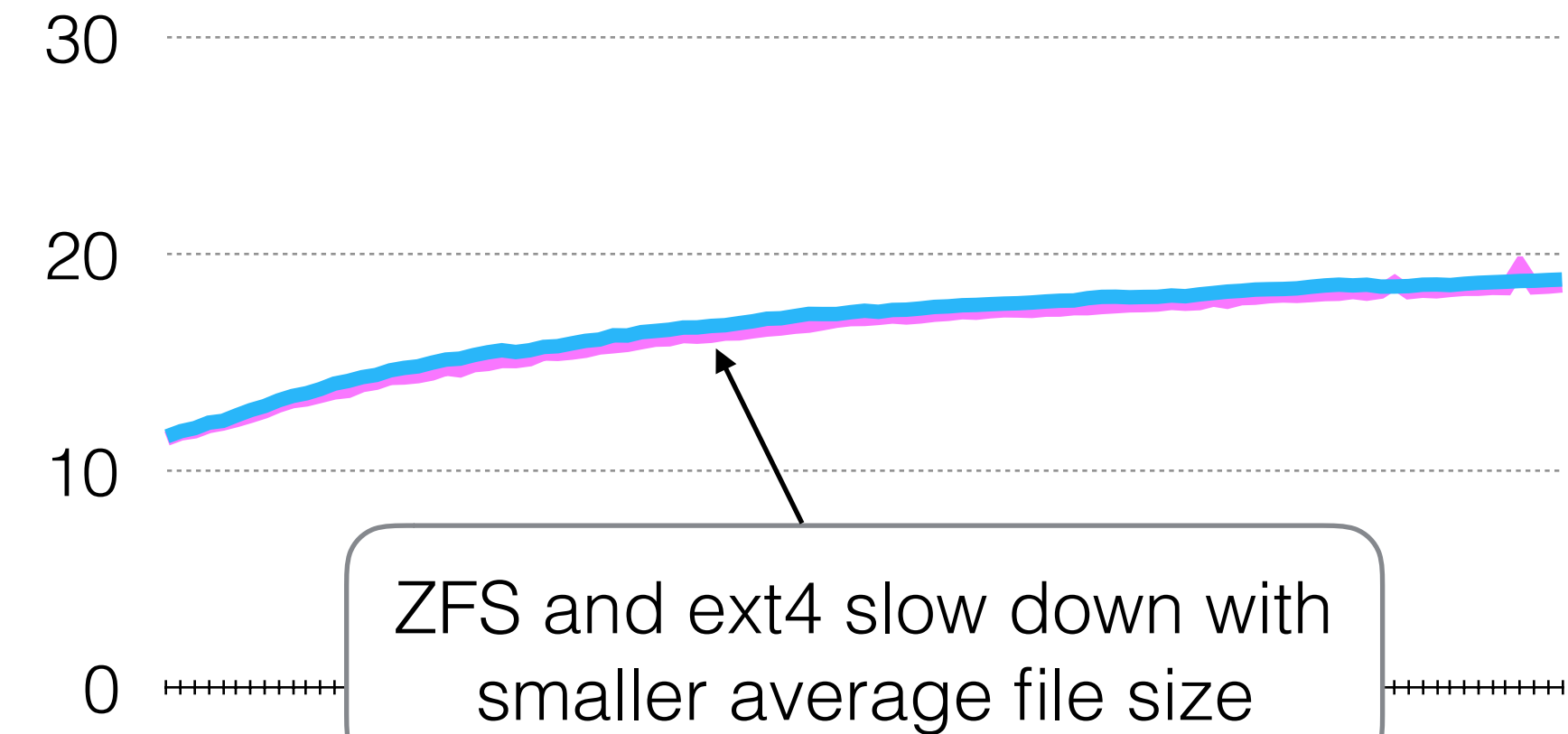


Git Workload on SSD

Btrfs

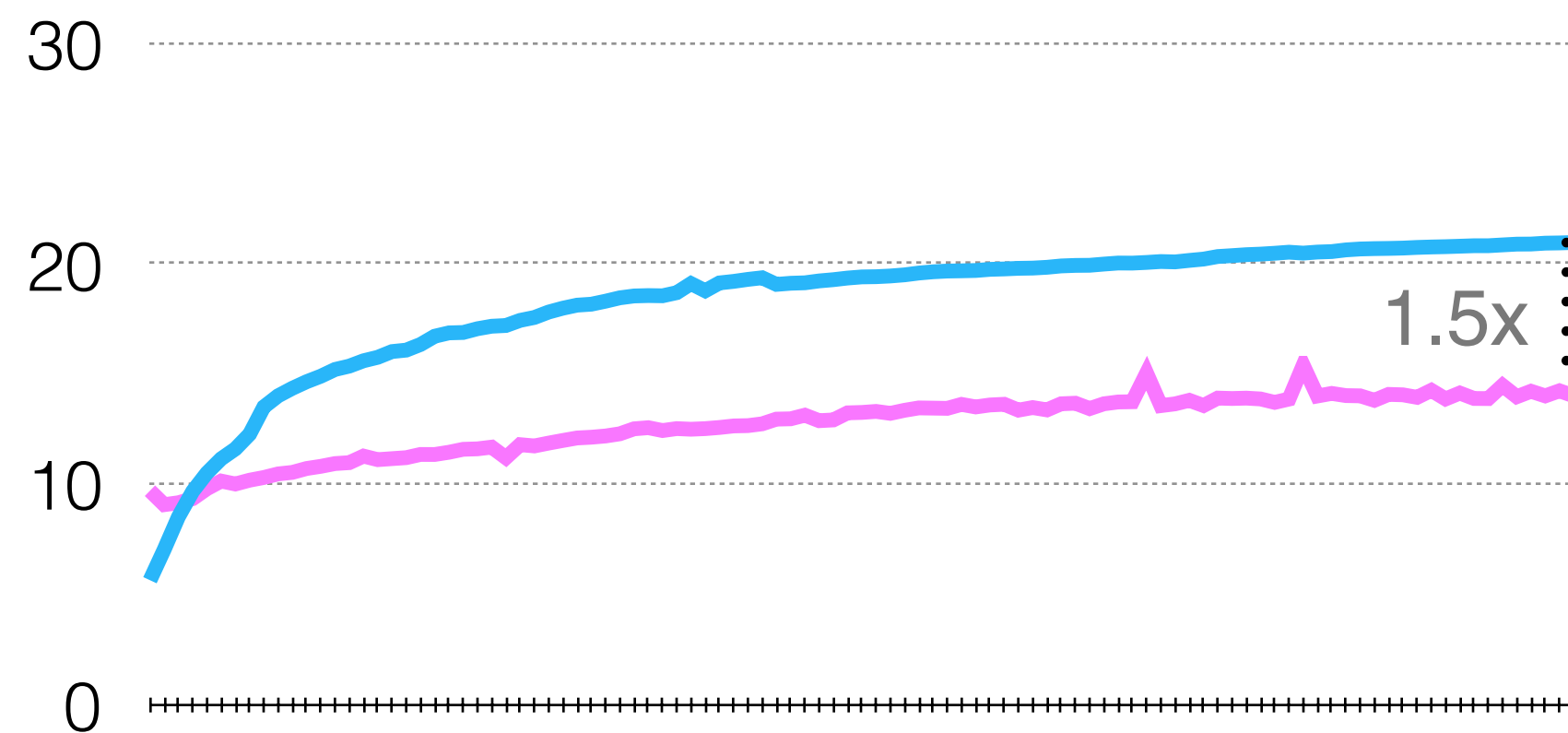


ext4

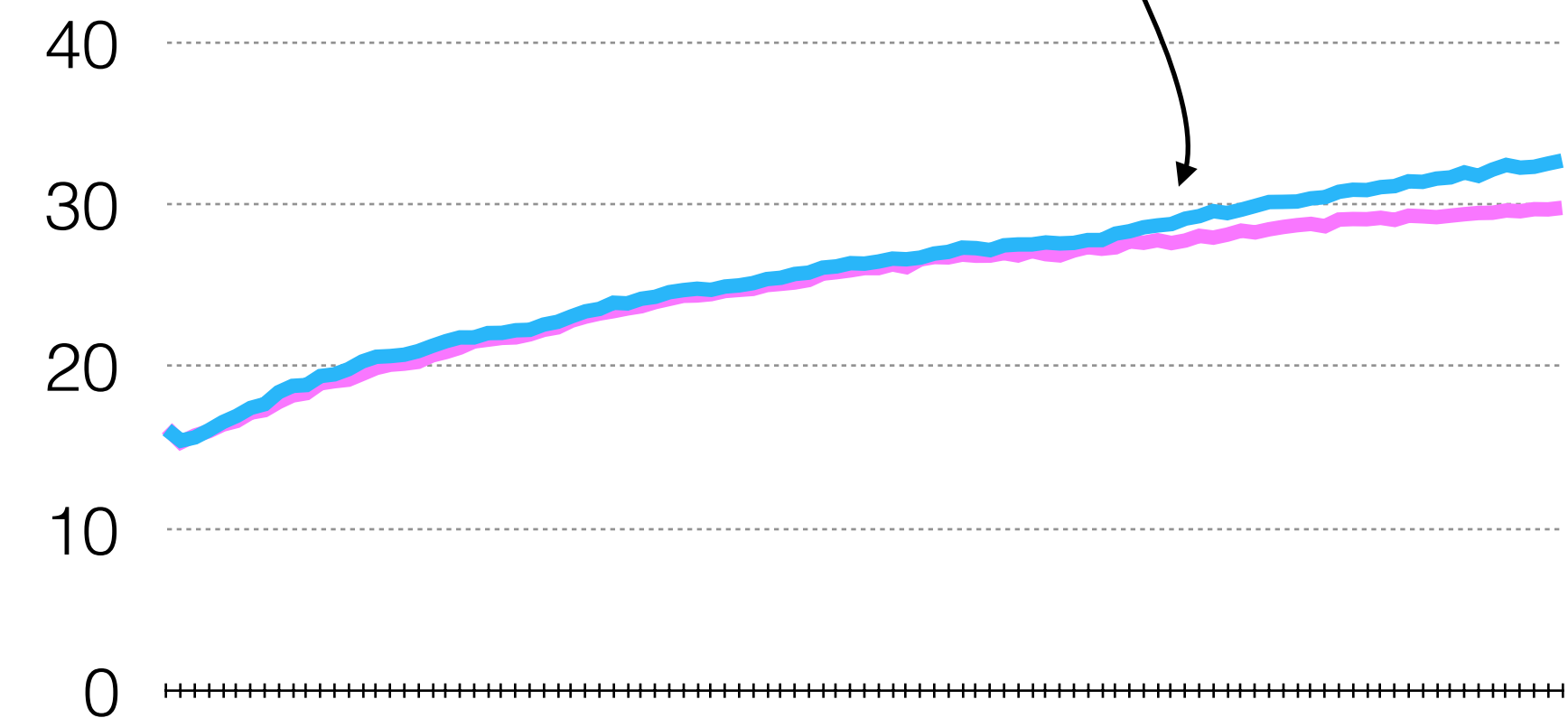


Lower is better

F2FS

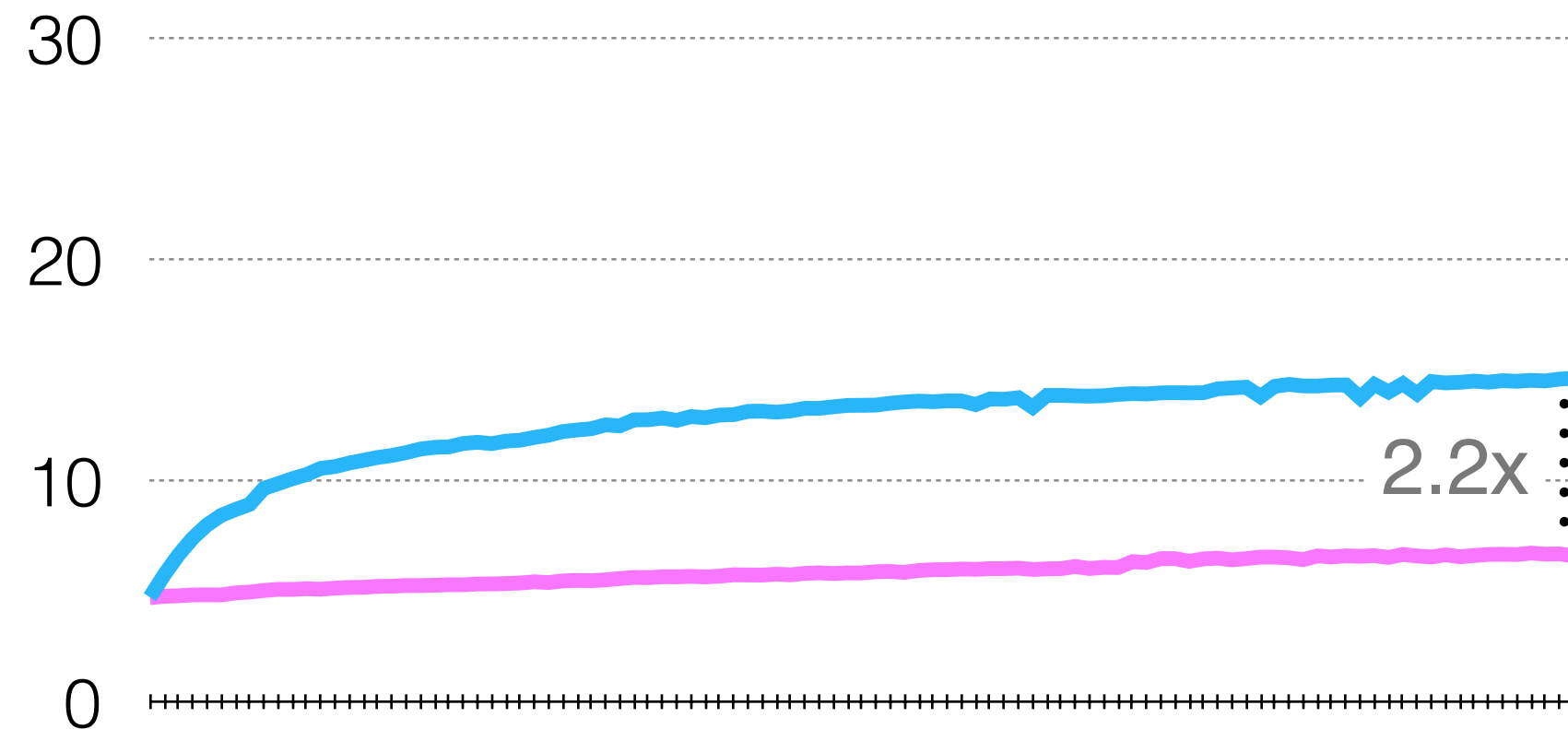


ZFS

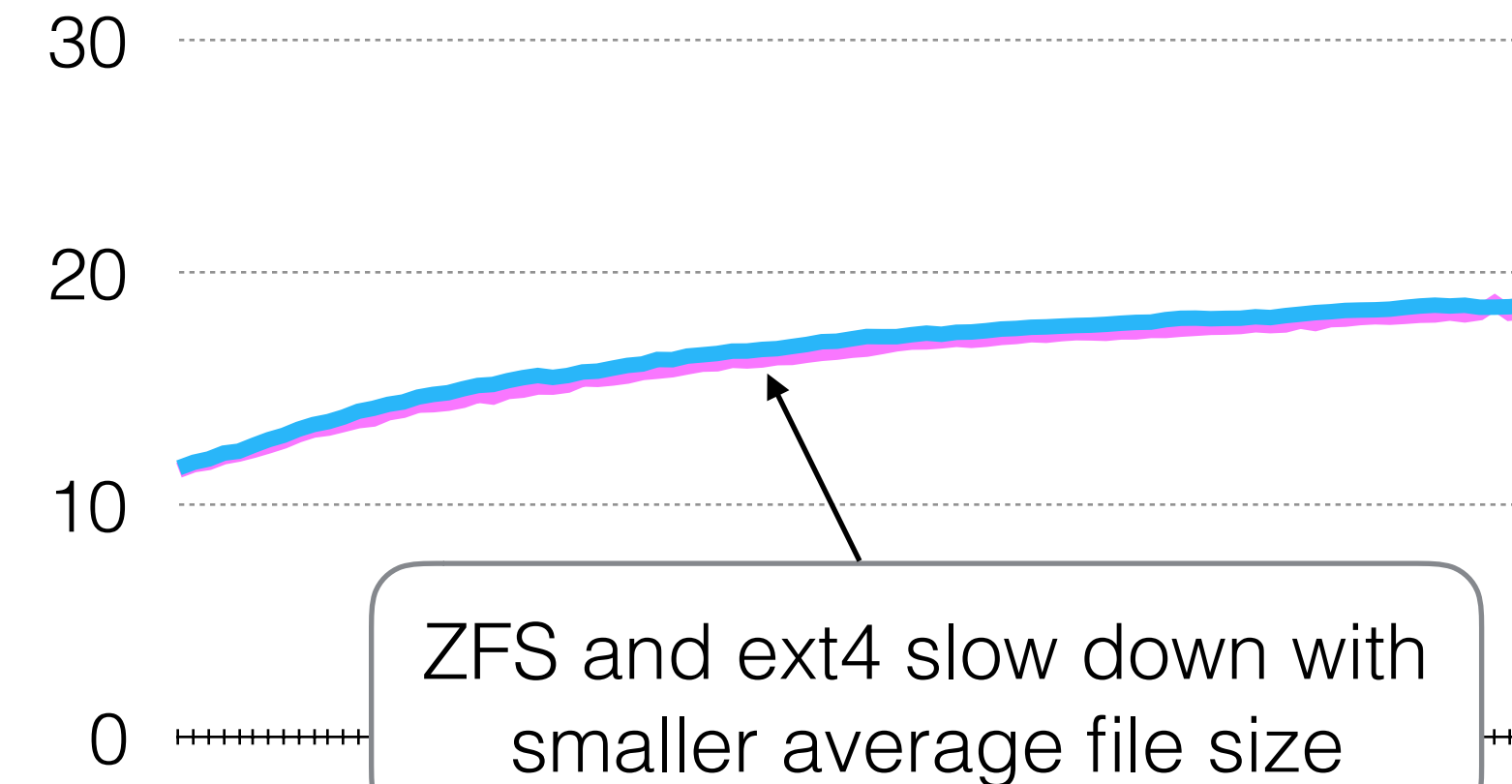


Git Workload on SSD

Btrfs

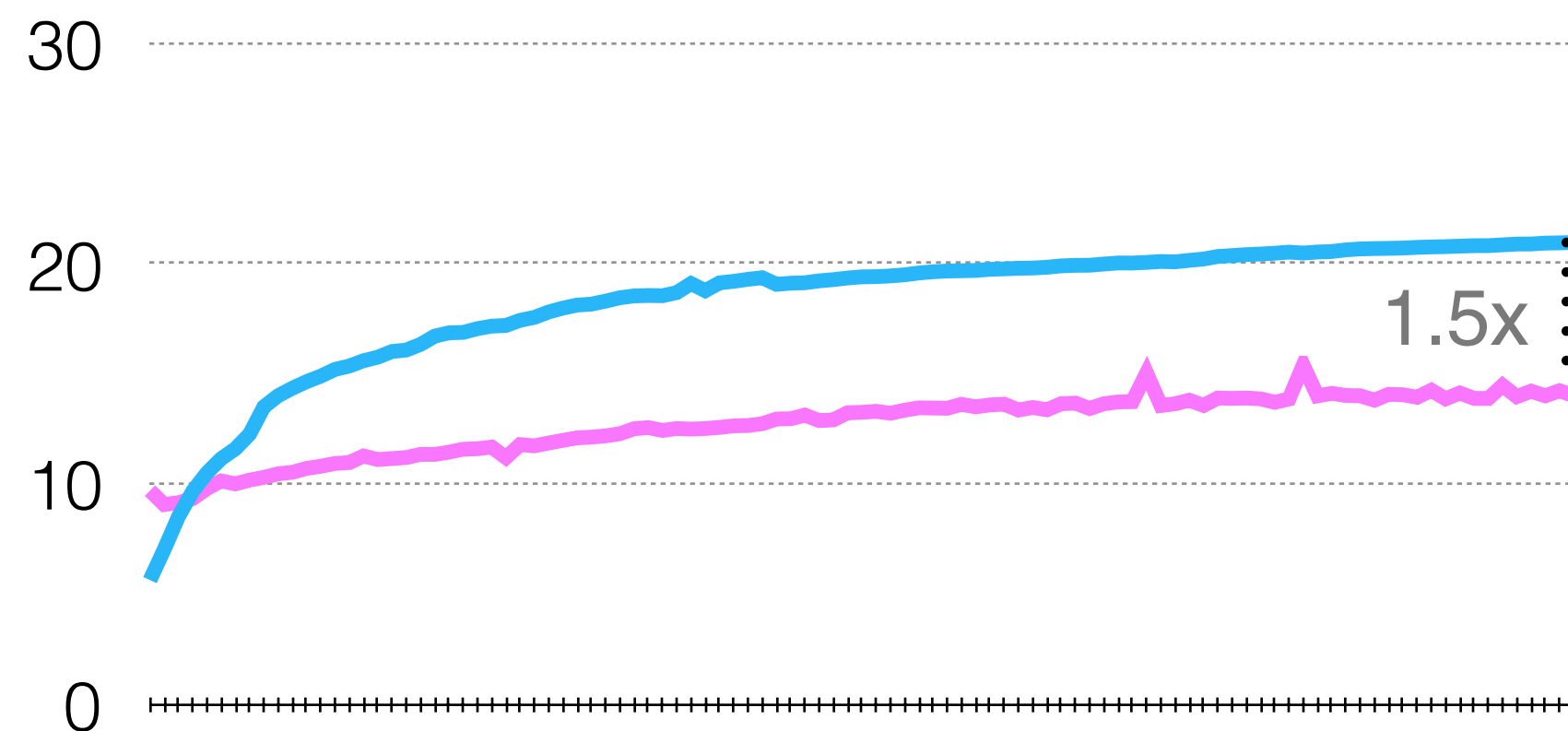


ext4

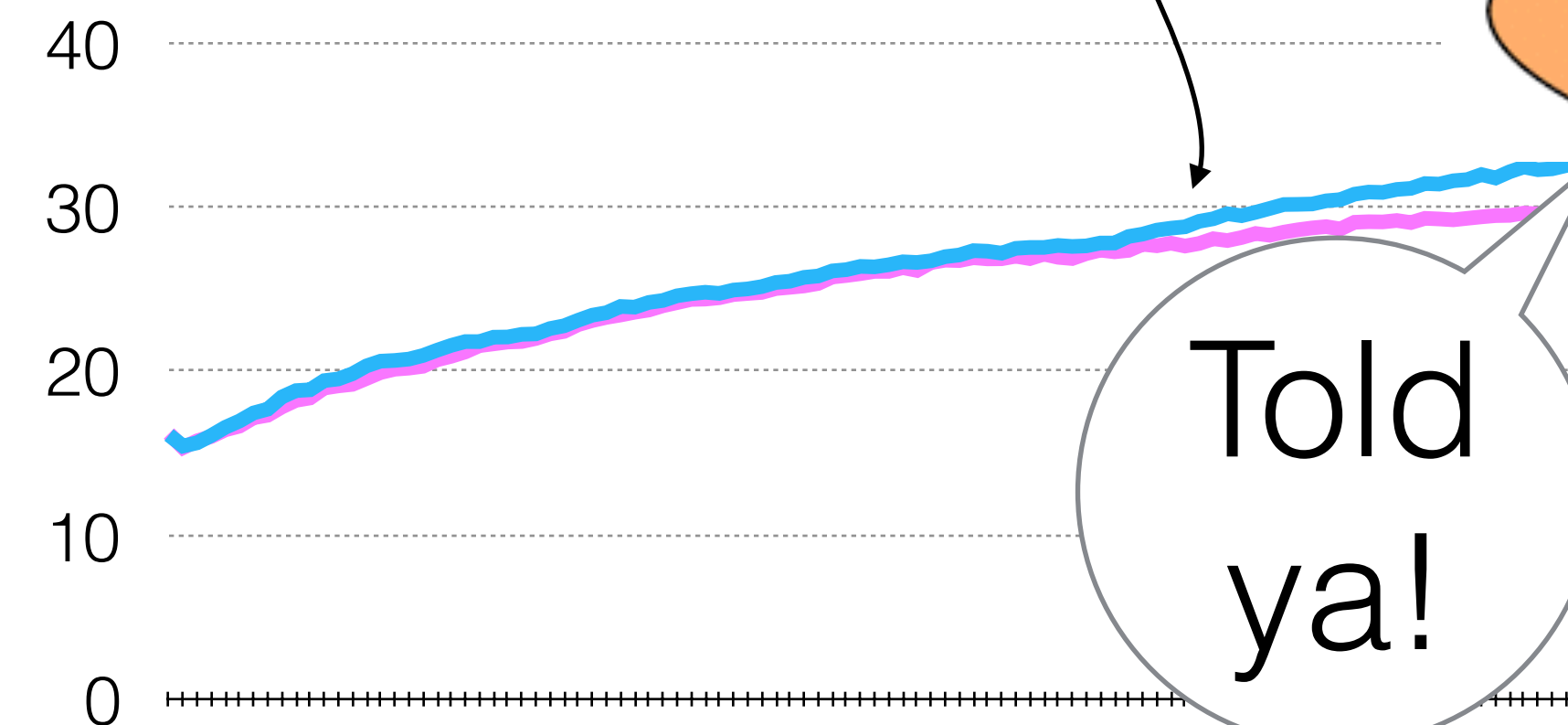


Lower is better

F2FS



ZFS



Aging is real

Btrfs, ext4, F2FS, XFS, 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
- Download these scripts from betrfs.org

How can we
prevent aging?

Design goals to address fragmentation

Intrafile Fragmentation:
Avoid breaking large files into small fragments

Design goals to address fragmentation

Intrafile Fragmentation:
Avoid breaking large files into small fragments

Interfile Fragmentation:
Cluster logically related small files

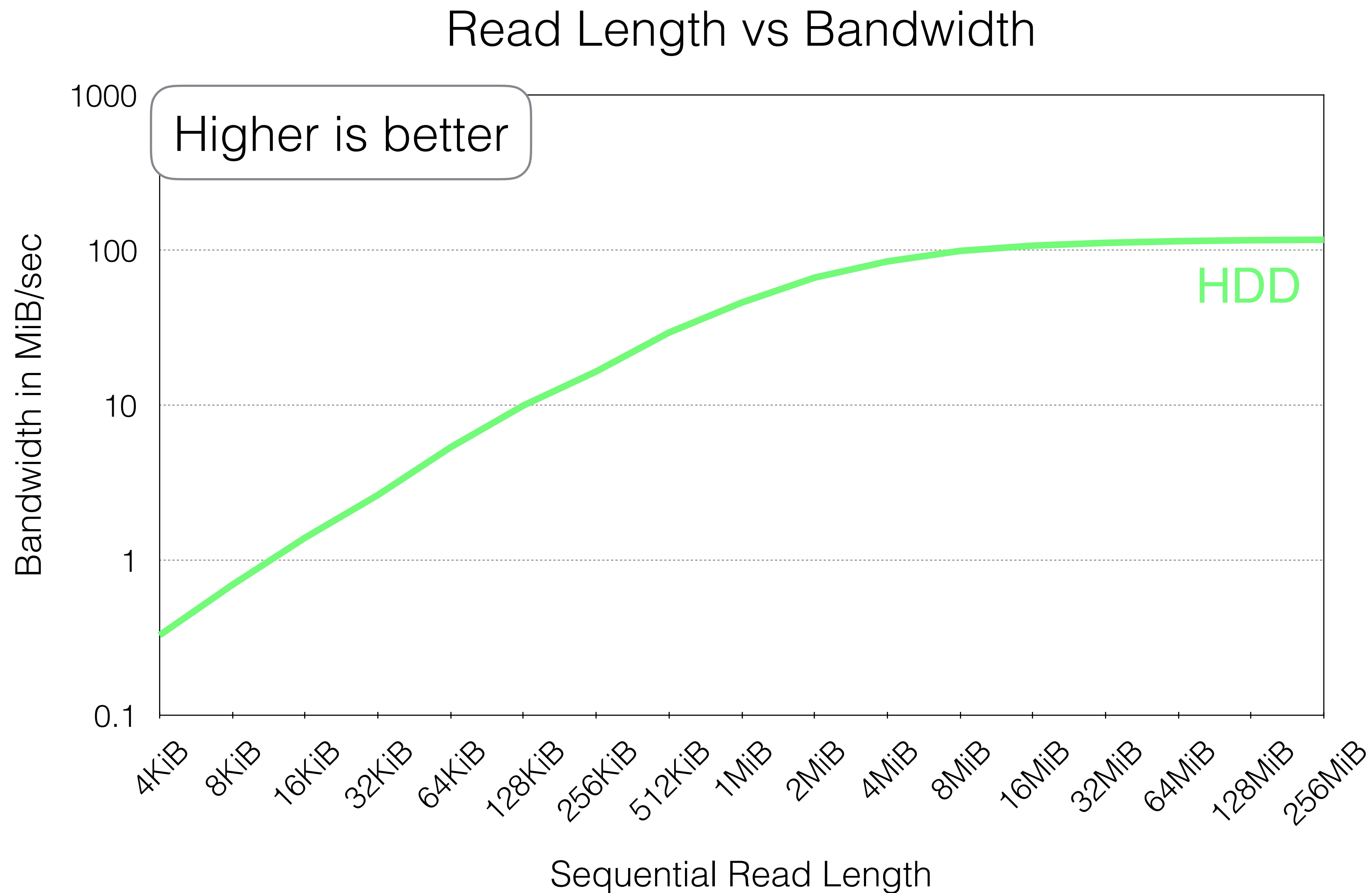
Design goals to address fragmentation

Intrafile Fragmentation:
Avoid breaking large files into small fragments

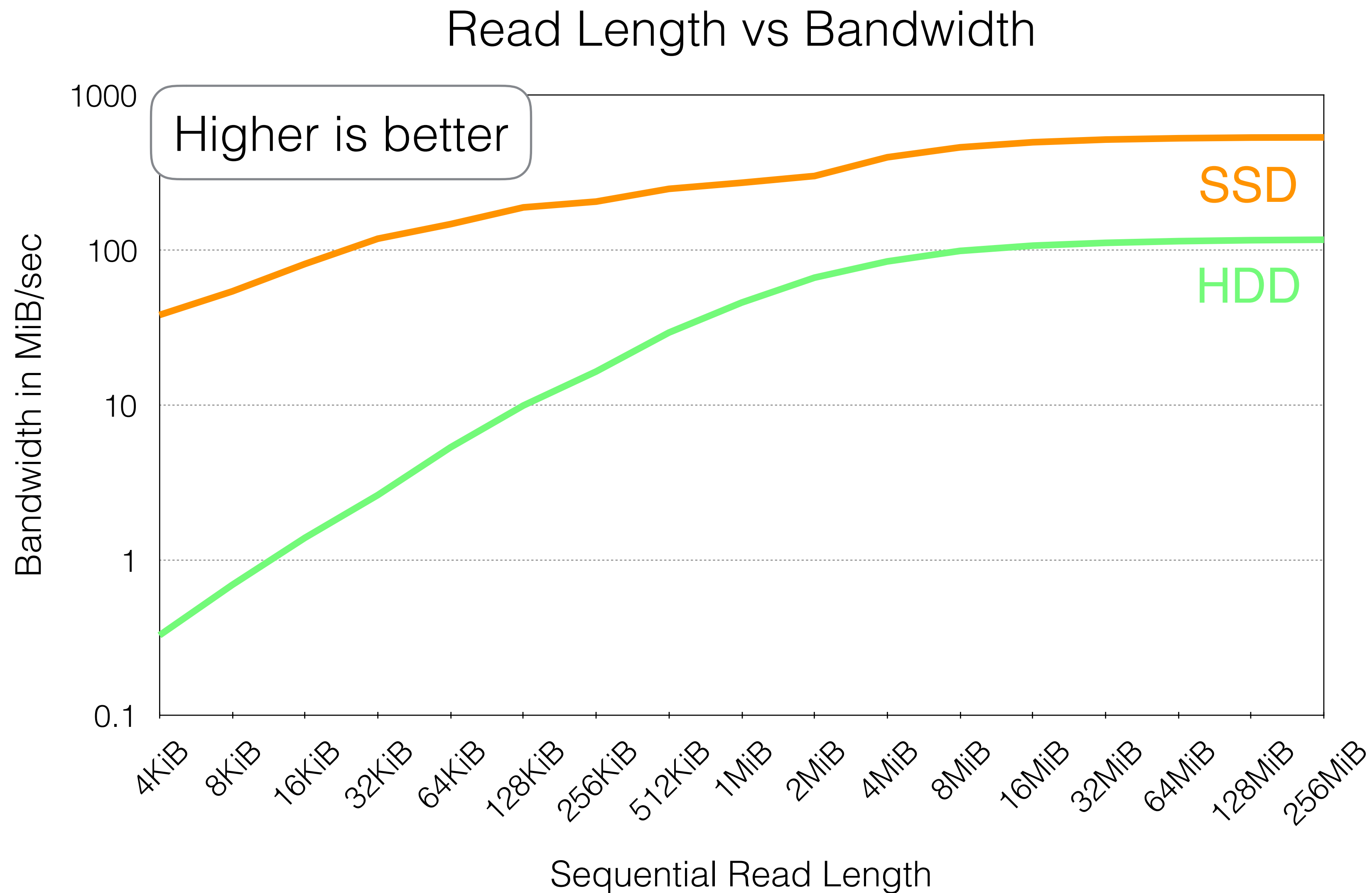
Interfile Fragmentation:
Cluster logically related small files

What do we mean by small?

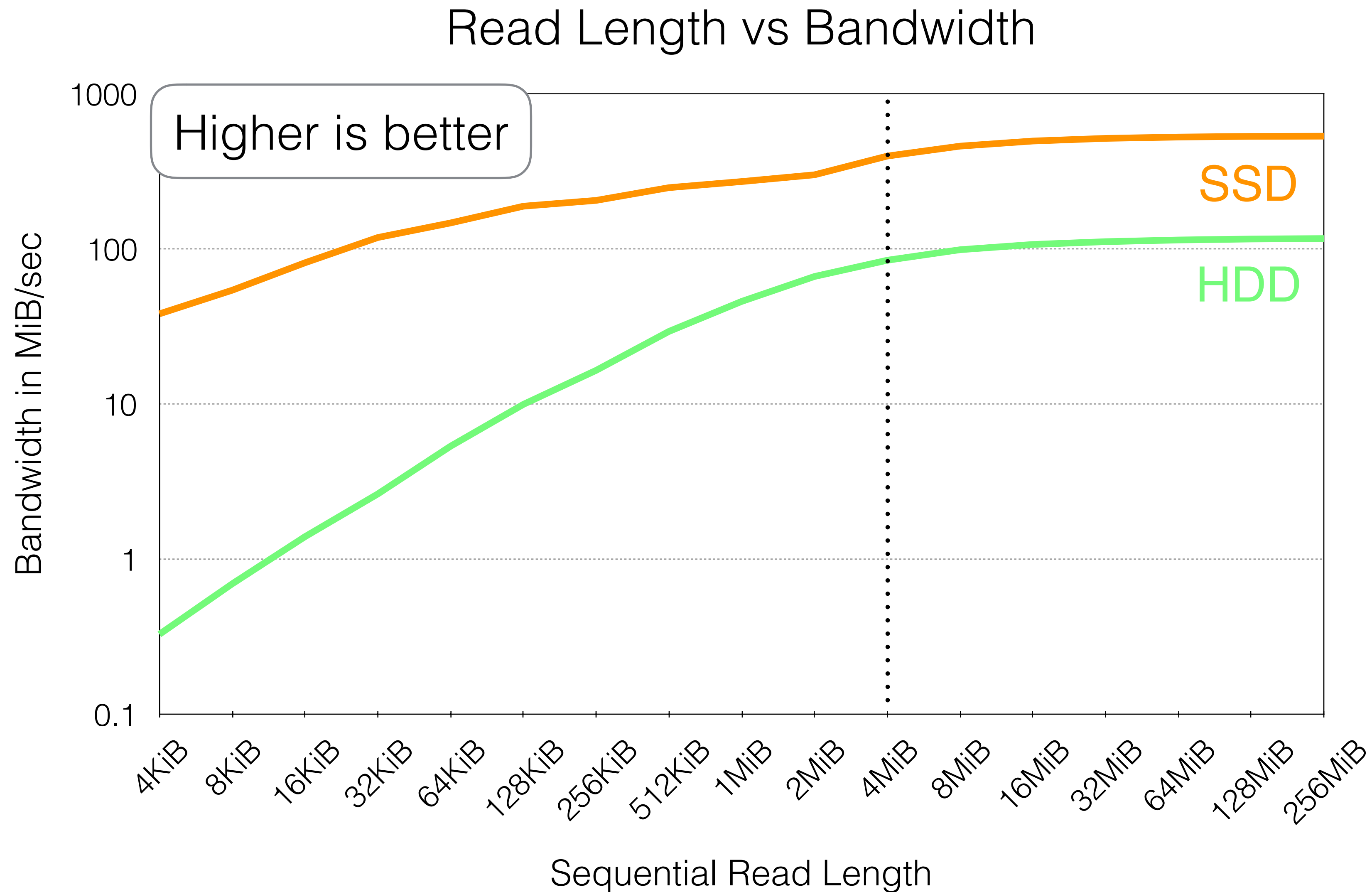
I/O Size vs Effective Bandwidth



I/O Size vs Effective Bandwidth



I/O Size vs Effective Bandwidth



Design goals to address fragmentation

Intrafile Fragmentation:
Avoid breaking large files into small fragments

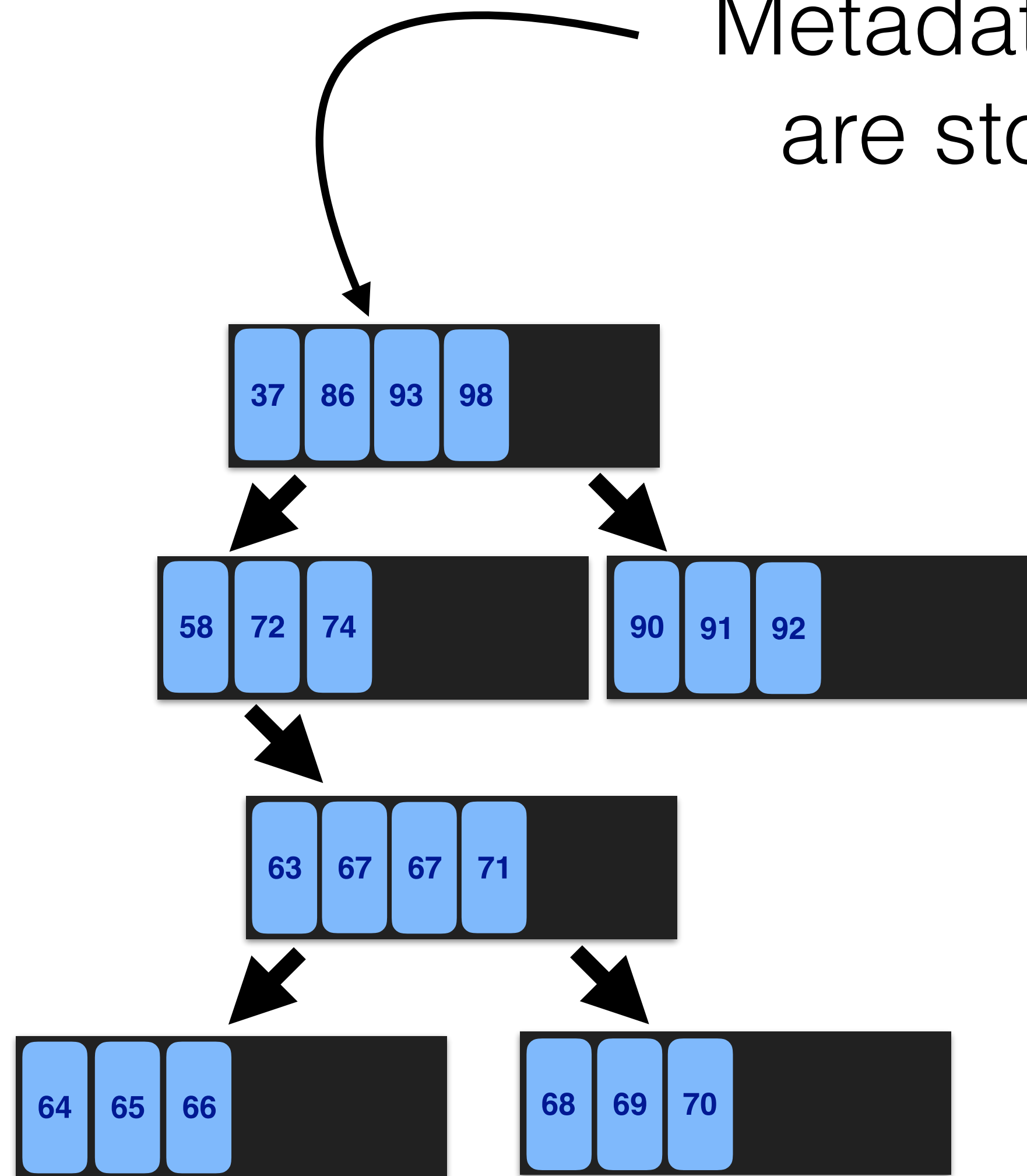
Interfile Fragmentation:
Cluster logically related small files

Prediction: 4MiB chunks will
substantially reduce aging

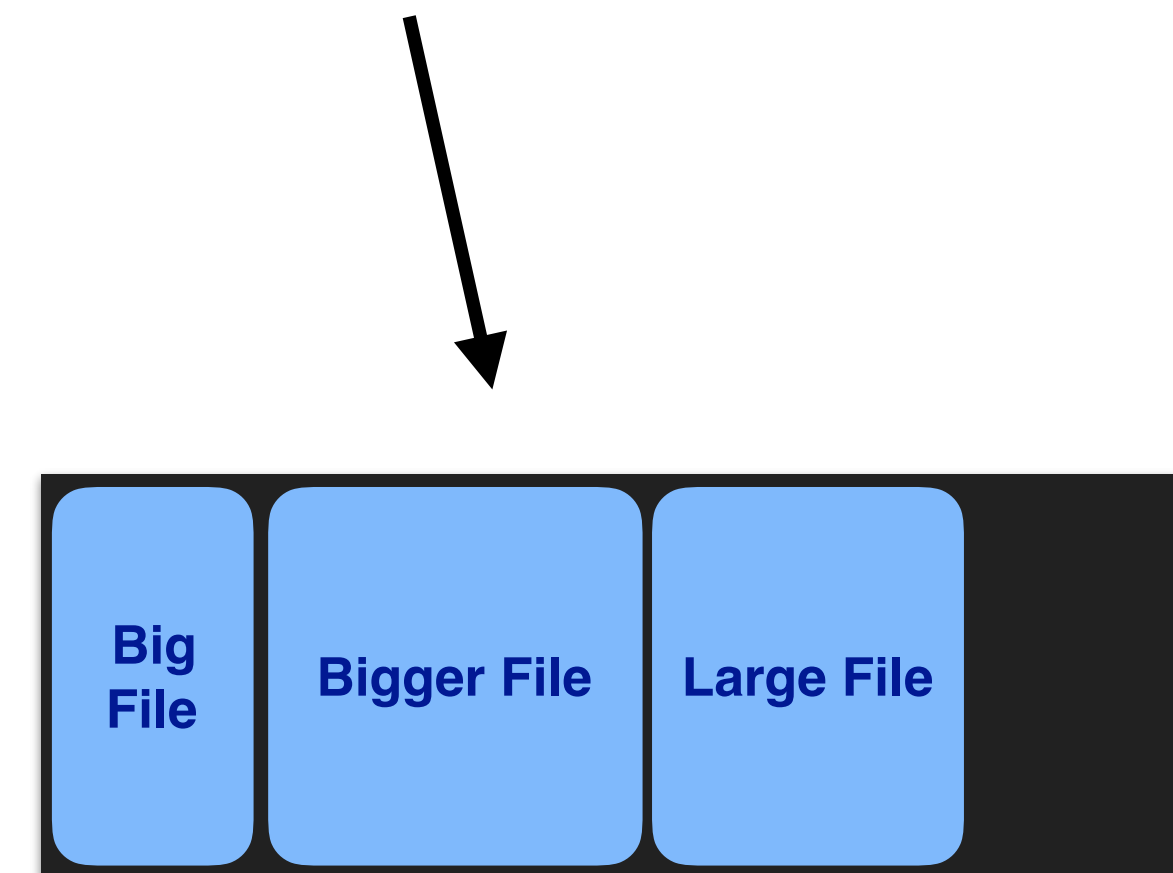
Testing this with Btrfs

Btrfs: Larger leaves = less aging?

Metadata and small files
are stored in a B-tree

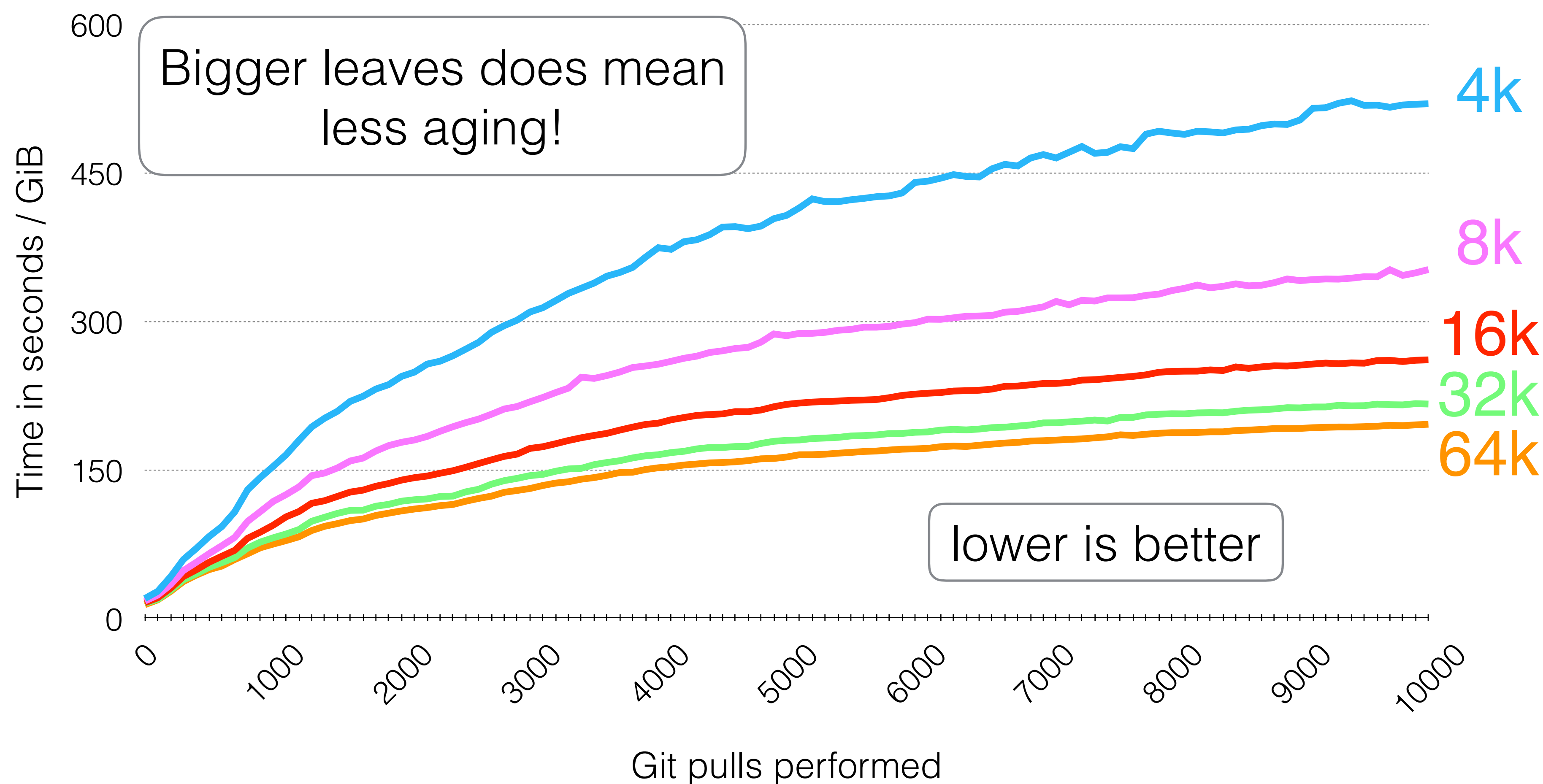


Large files get written
elsewhere



Btrfs Leaf Size Performance

Btrfs allows leaf size to be configured between 4KiB and 64KiB.



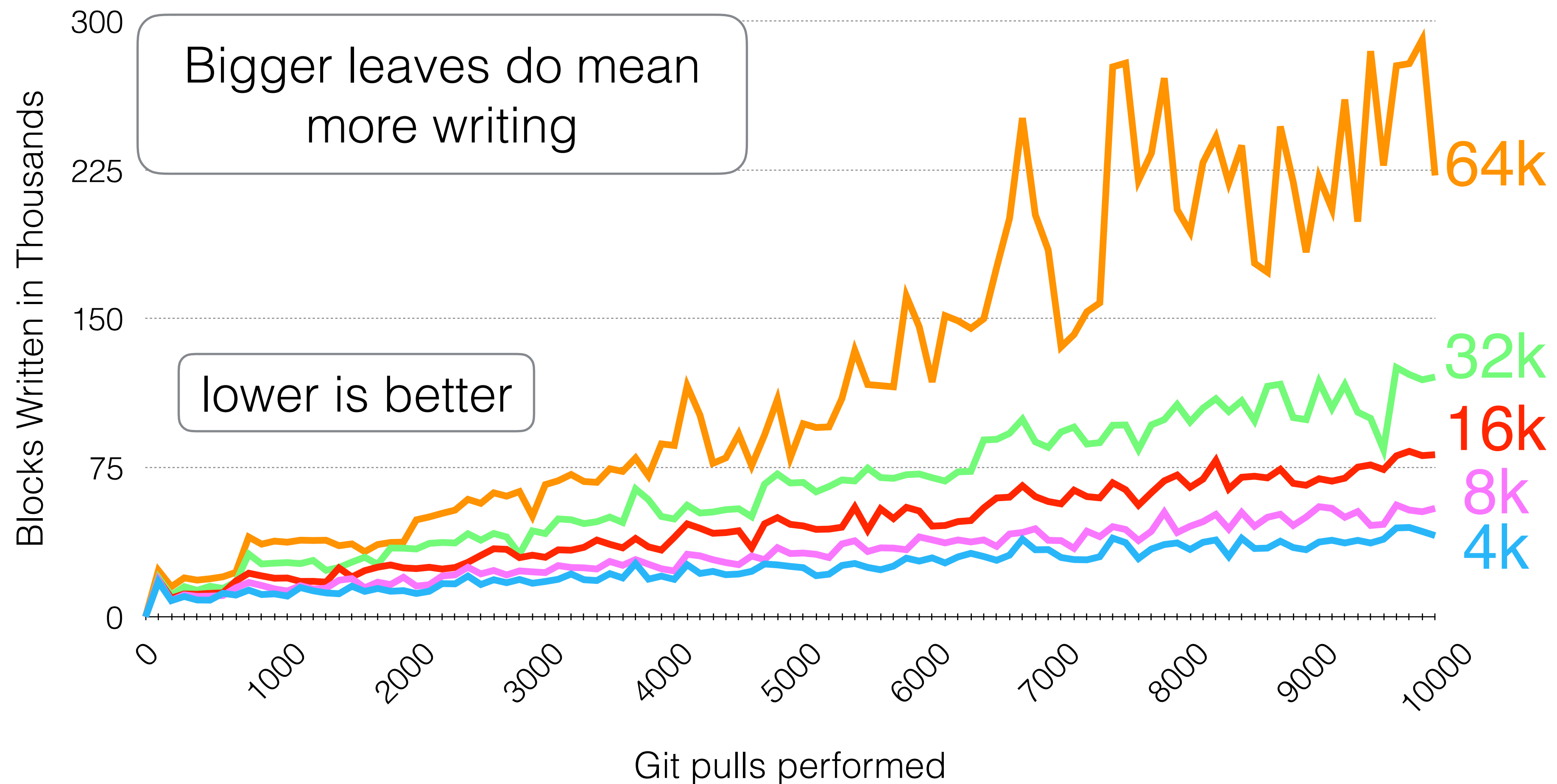
Cost of large leaves

Why don't B-tree usually
have big leaves?

Because making small changes to
big leaves causes a lot of writing

Btrfs Leaf Size Writing

Btrfs allows leaf size to be configured between 4KiB and 64KiB.



B-Tree Performance Tradeoff

Small Leaves

More Aging 😞

Less Writing 😊

Large Leaves

Less Aging 😊

More Writing 😞

B-Tree Performance Tradeoff

Small Leaves

More Aging 😞

Less Writing 😊



Large Leaves

Less Aging 😊

More Writing 😞



This tradeoff is inherent to B-trees

Other File System Types

Must other types of file systems age?

Update-in-place

Log-structured

Write-Optimized



See the paper

BetrFS

BεtrFS packs small
logically related data in a
B^ε-tree with 4MiB nodes.

BεtrFS packs small logically related data in a B ϵ -tree with 4MiB nodes.

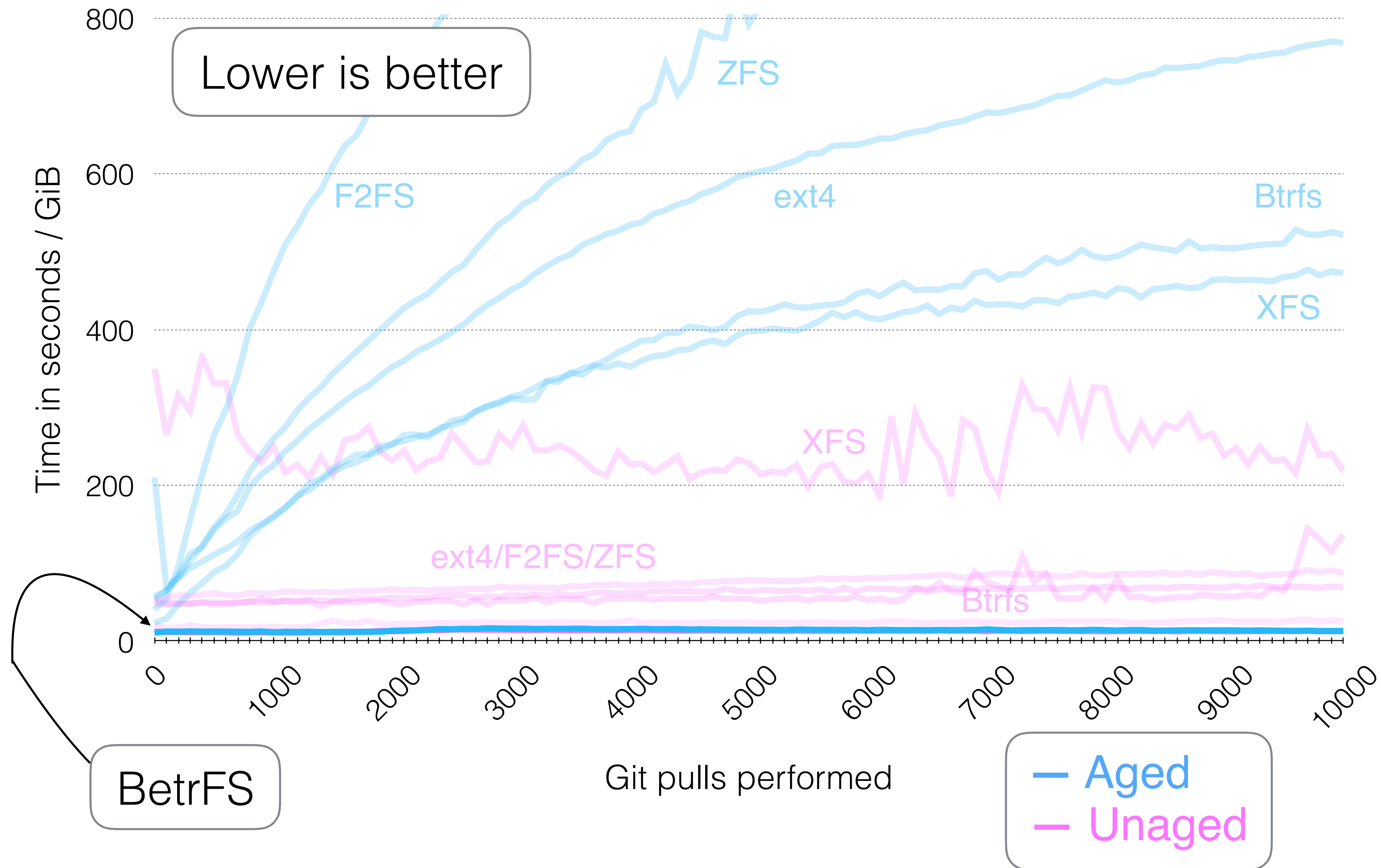


BεtrFS packs small logically related data in a B ϵ -tree with 4MiB nodes.

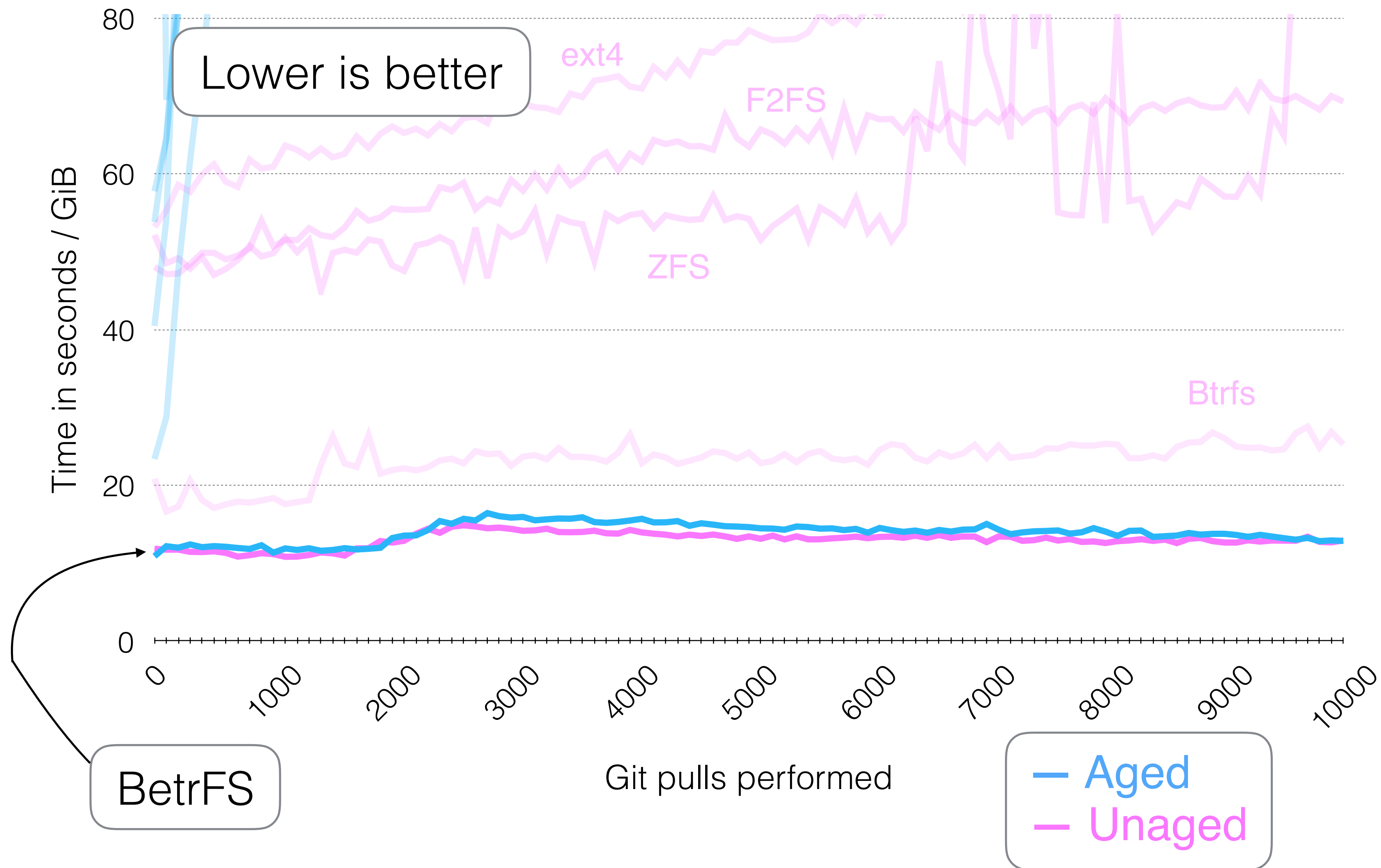
B ϵ -trees batch updates which allows leaves to be big without increasing the amount of writing



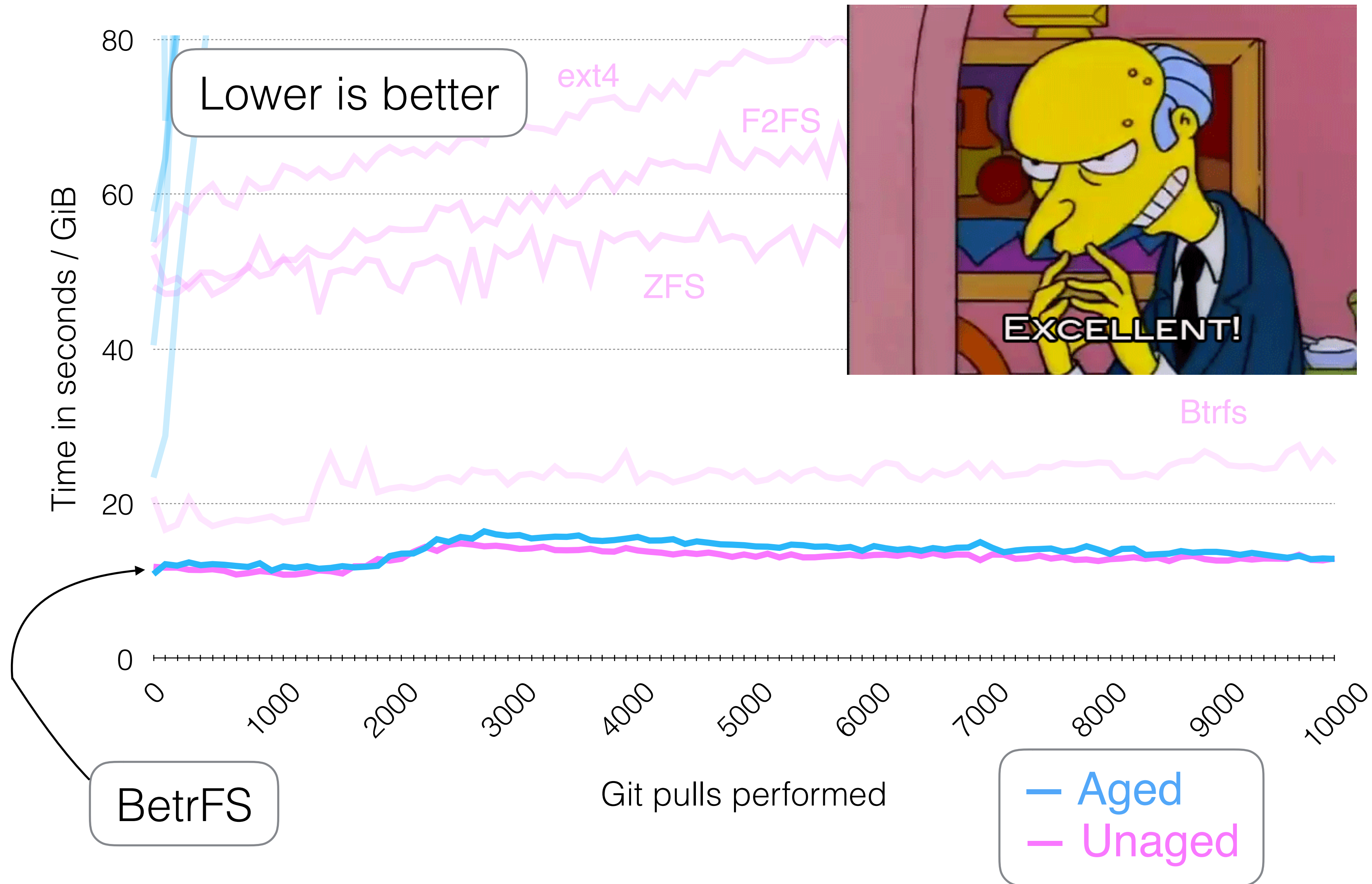
Git on BetrFS on HDD



Git on BetrFS on HDD



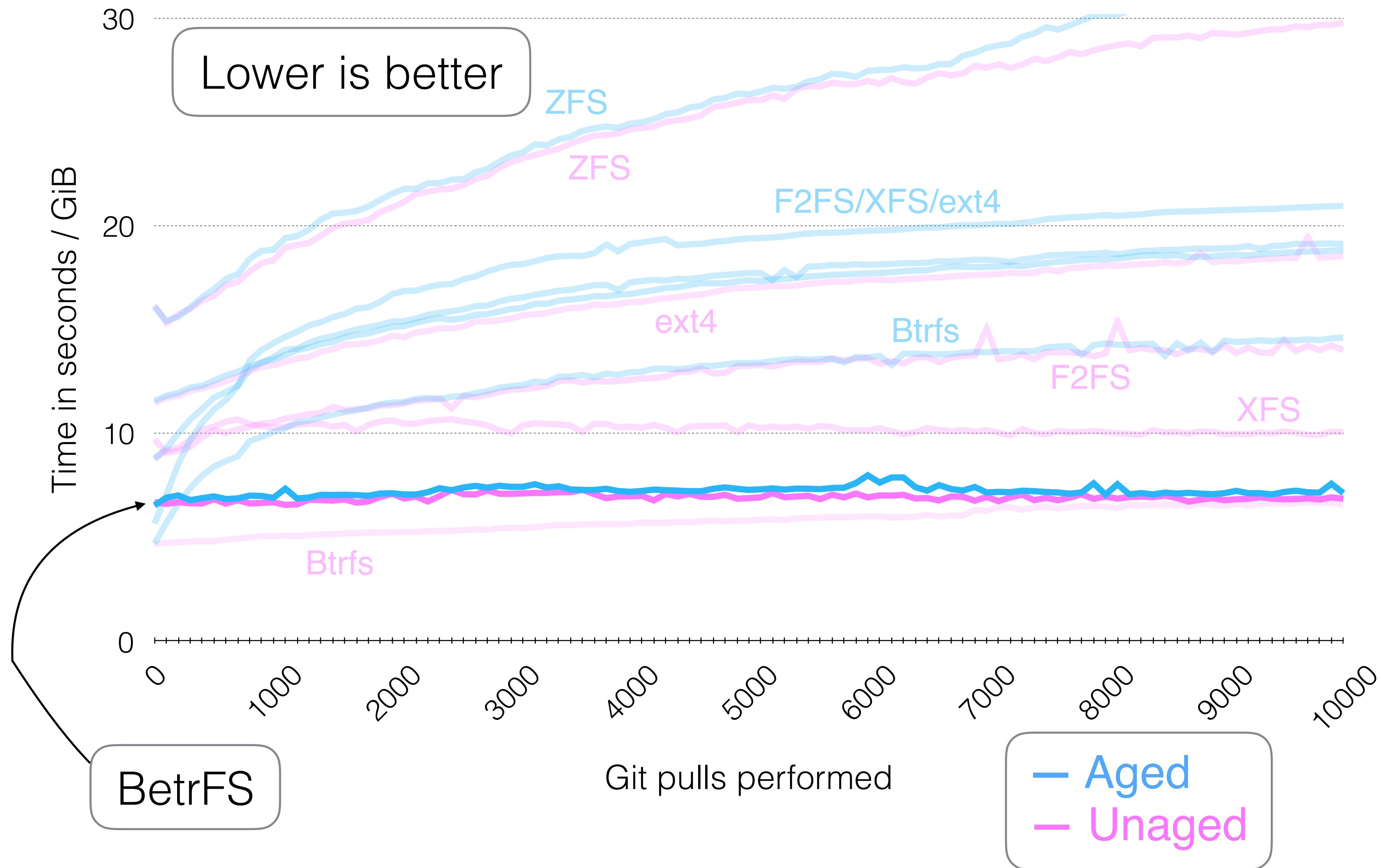
Git on BetrFS on HDD



And SSDs?



Git on BetrFS on SSD



How to prevent aging

Rewrite to keep related data in large blocks

Batch updates to avoid too much writing

Conclusion

It's easy to age file systems
quickly and substantially

Aging is avoidable

Thank you!



Alex Conway

betrfs.org

alexander.conway@rutgers.edu