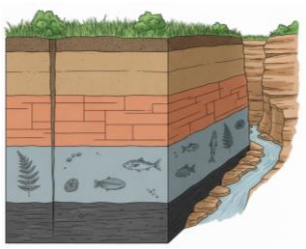
Creating a Standardized Corpus for Digital Stratigraphic Methods with fsstratify

Julian Uthoff, Lisa Marie Dreier, Martin Lambertz, Mariia Rybalka, Felix Freiling

Digital Stratigraphy

> *Stratigraphy is the scientific study of layers […] with the aim of determining the origin, composition, distribution, and time frame of each stratum.*

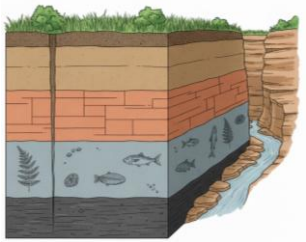


> *[…] approaches to conducting contextual analysis of file allocation traces, […] called digital stratigraphy, translate concepts from stratigraphy in geology and archeology into the digital realm.*

– Casey: „Digital Stratigraphy: Contextual Analysis of File System Traces in Forensic Science“, 2018.

Digital Stratigraphy

> *Stratigraphy is the scientific study of layers [...] with the aim of determining the origin, composition, [...] file allocation traces, [...] called digital stratigraphy, translate concepts from stratigraphy in geology and archeology into the digital realm.*
>
> – Casey: „Digital Stratigraphy: Contextual Analysis of File System Traces in Forensic Science“, 2018.



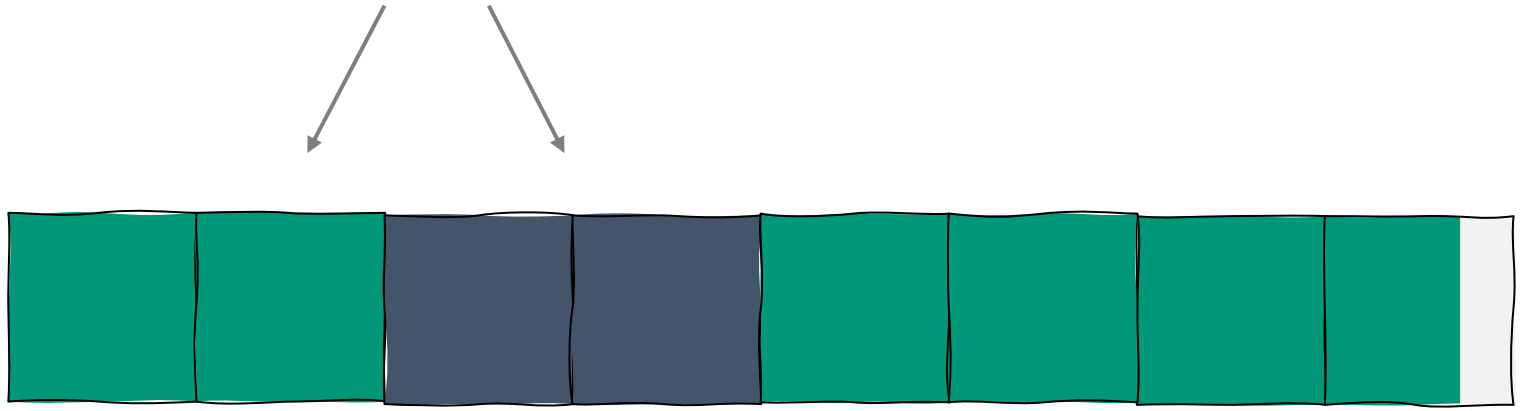
tl;dr

Extract all available metadata of a target file (system) from its context, even when the metadata is missing.

Digital Stratigraphy: Examples

Assume we have no metadata:

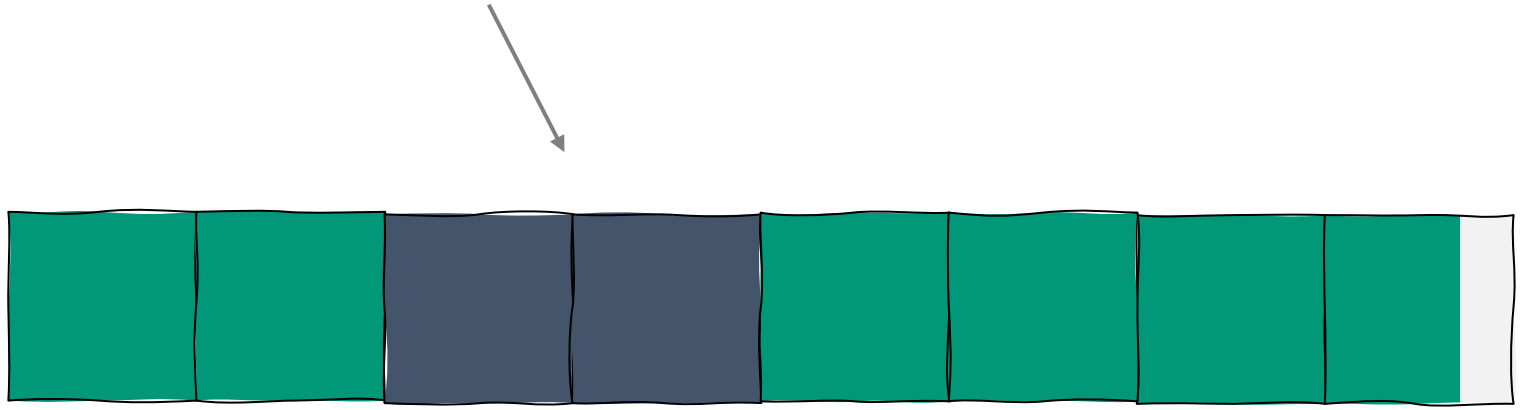
Which file is older?



Digital Stratigraphy: Examples

Assume we have no metadata:

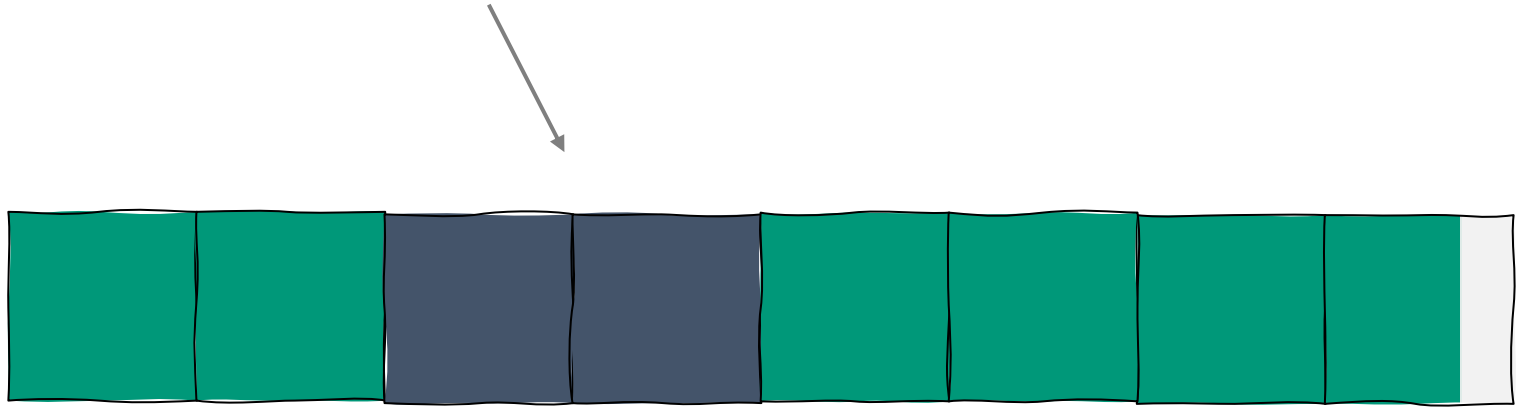
Is this file is older than 2024-10-22?



Digital Stratigraphy: Examples

Assume we have no metadata:

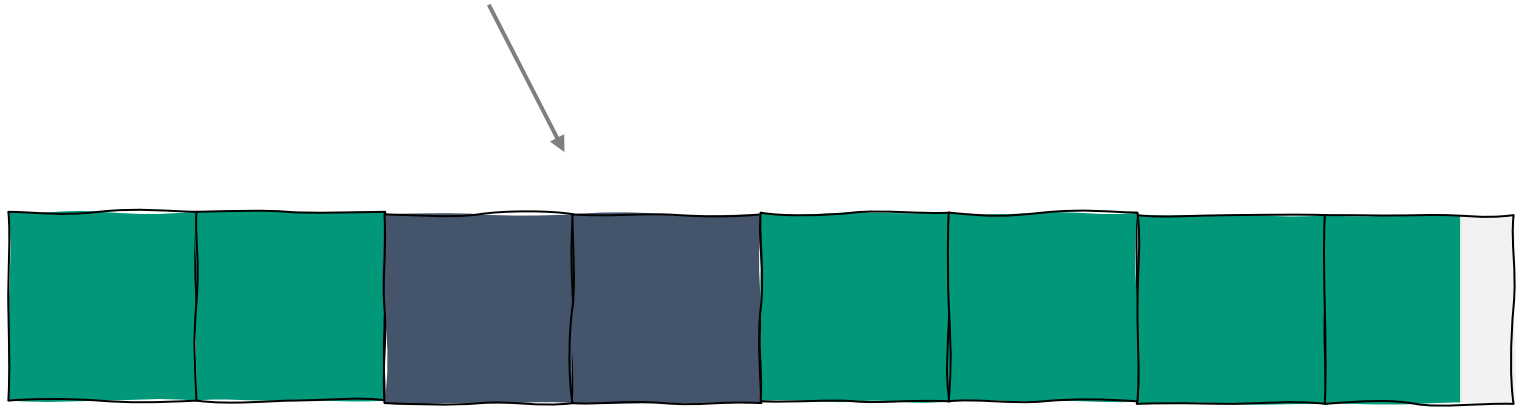
Which account created this file?



Digital Stratigraphy: Examples

If we have metadata:

Were the timestamps of this file modified?



Digital Stratigraphy: Current State

2008

Willassen: „Finding Evidence of Antedating in Digital Investigations”

2018

Casey: „Digital Stratigraphy: Contextual Analysis of File System Traces in Forensic Science”

2019

Karresand, Axelsson, Dyrkolbotn: „Disk Cluster Allocation Behavior in Windows and NTFS”

2023

Bahjat, Jones: „File Allocation Chronology and its Impact on Digital Forensics”

2025 (today)

Uthoff, Dreier, Lambertz, Rybalka, Freiling: „Creating a Standardized Corpus for Digital Stratigraphic Methods with fsstratify“

2010

Garfinkel, Parker-Wood, Huynh, Migletz: „An Automated Solution to the Multiuser Carved Data Ascription Problem”

2019

Bahjat, Jones: „Deleted file fragment dating by analysis of allocated neighbors”

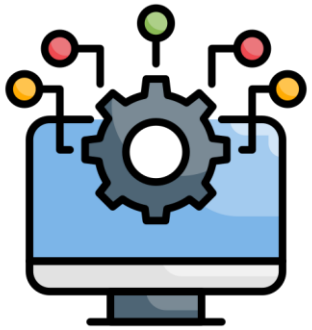
2020

Karresand, Dyrkolbotn, Axelsson: „An Empirical Study of the NTFS Cluster Allocation Behavior Over Time”

2024

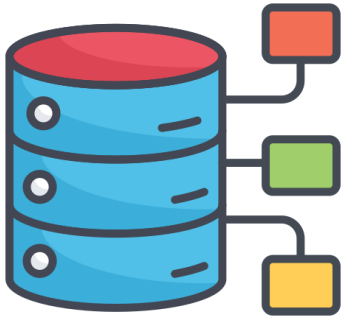
Schneider, Eichhorn, Dreier, Hargreaves: „Applying digital stratigraphy to the problem of recycled storage media”

Digital Stratigraphy: Contributions



fsstratify

We revised and extended fsstratify, a framework to generate „used“ file systems,...



Dataset

…created a corpus of NTFS file system images for chronological dating,…

Evaluation

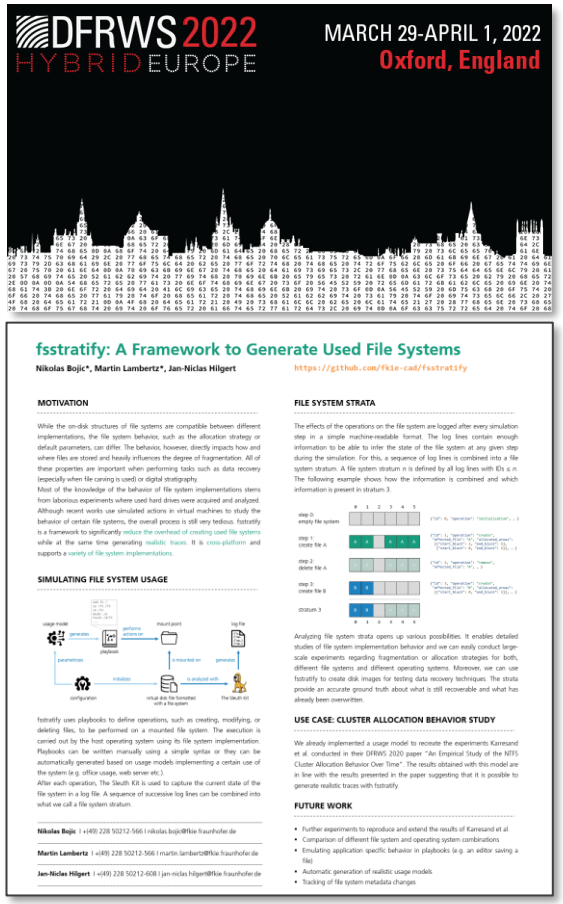
…and evaluated the method proposed by Bahjat and Jones.

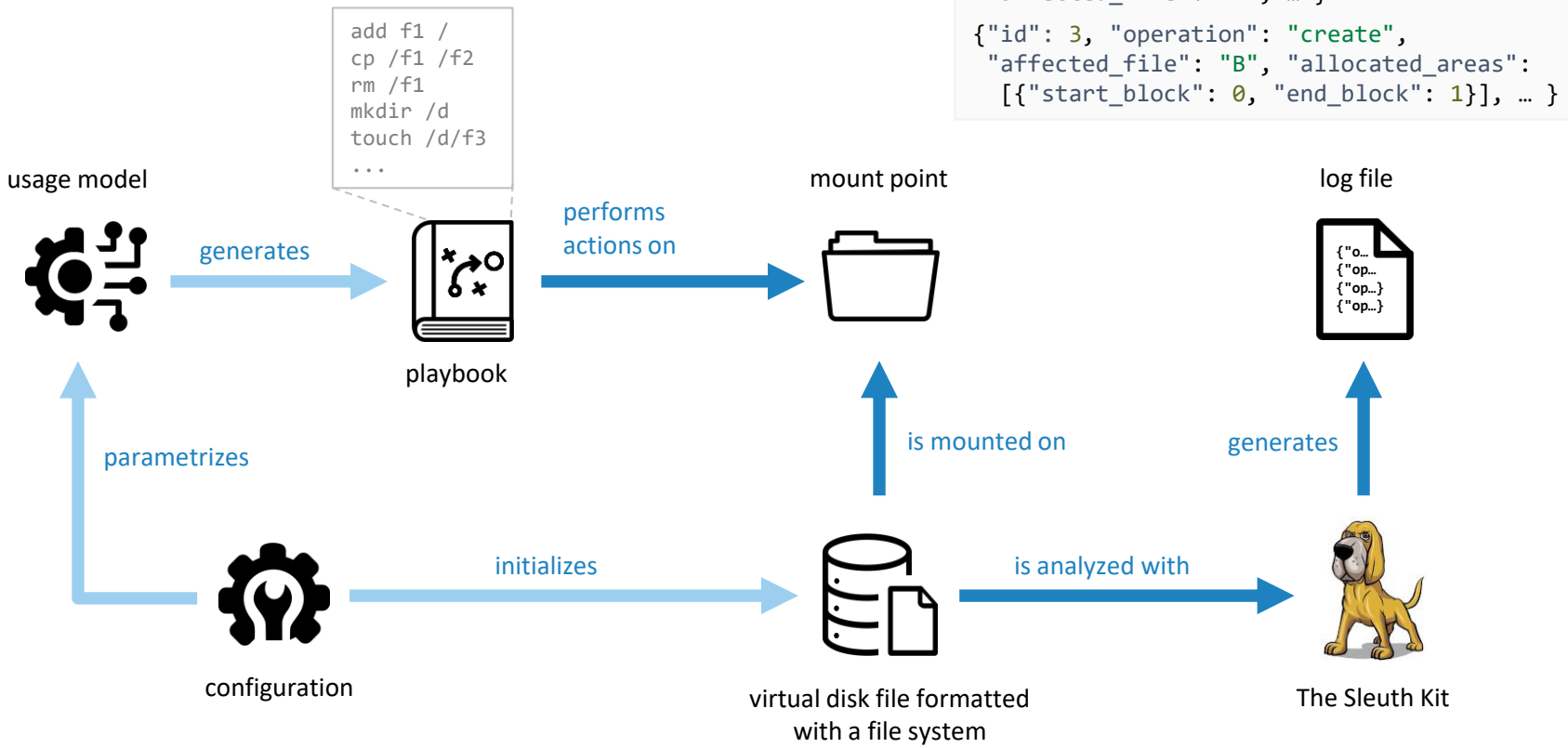
2019

Bahjat, Jones: „Deleted file fragment dating by analysis of allocated neighbors”

[Icons created by Design Circle – Flaticon]

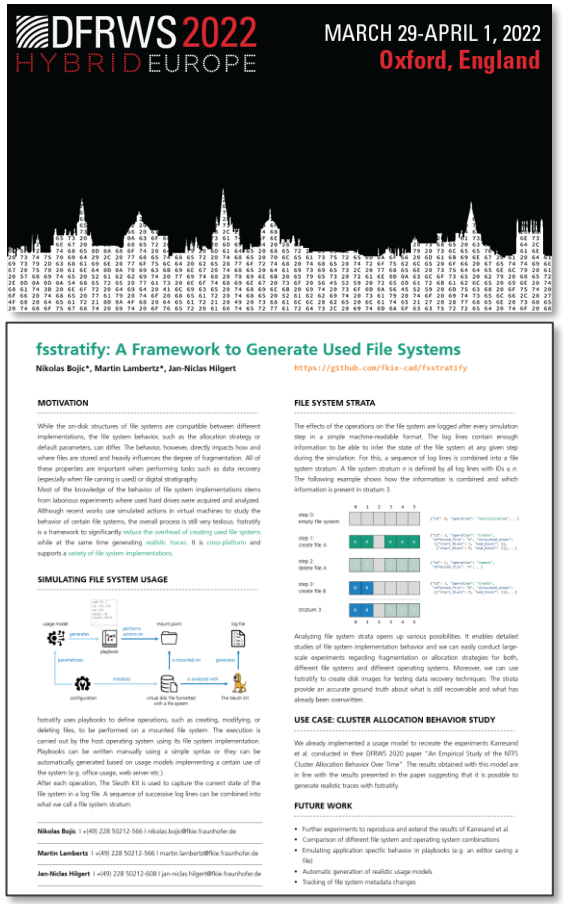
fsstratify





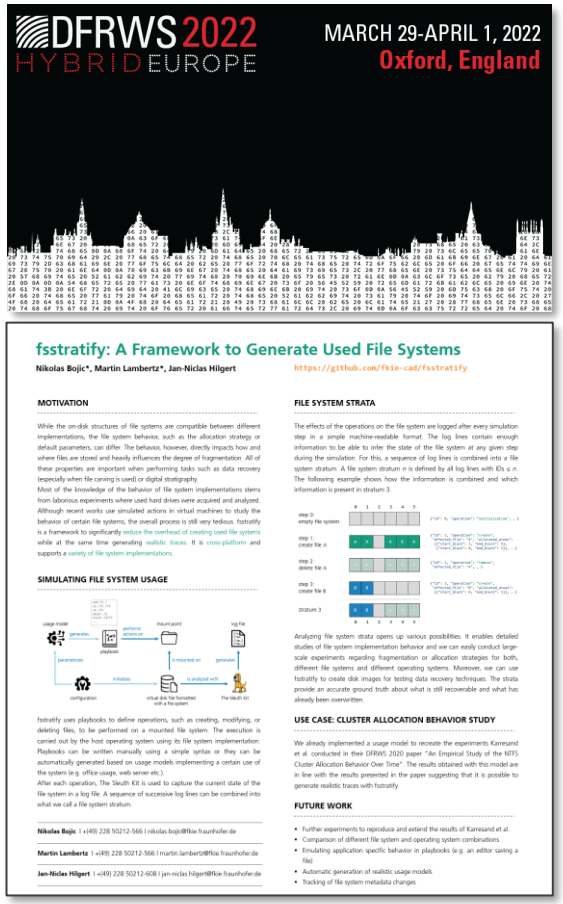

```
{"id": 0, "operation": "initialization", … }
{"id": 1, "operation": "create",
 "affected_file": "A", "allocated_areas":
  [{"start_block": 3, "end_block": 5},
   {"start_block": 0, "end_block": 1}], … }
{"id": 2, "operation": "remove",
 "affected_file": "A", … }
{"id": 3, "operation": "create",
 "affected_file": "B", "allocated_areas":
  [{"start_block": 0, "end_block": 1}], … }
```

fsstratify



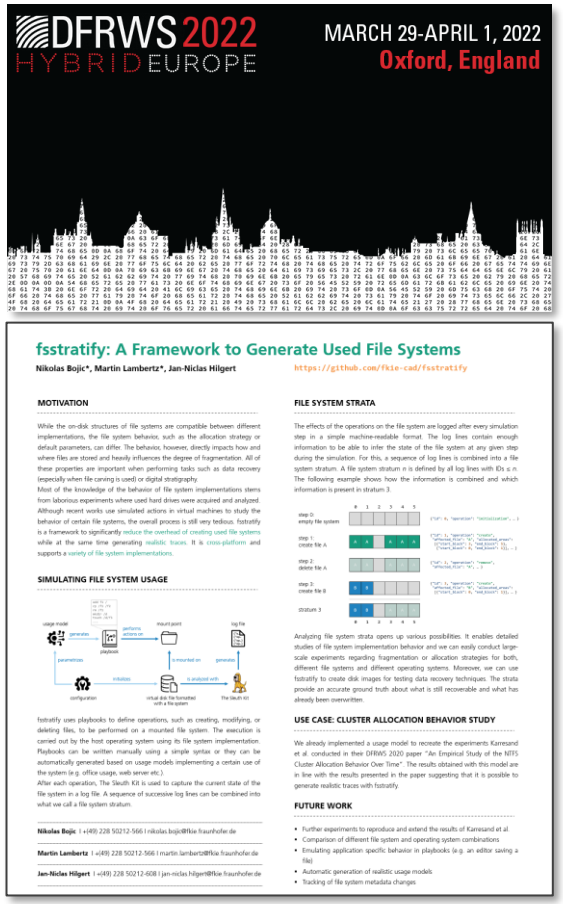
- Code not really publicly available
- Not all required features available
- Indeterminate simulation crashes
- Slow

fsstratify



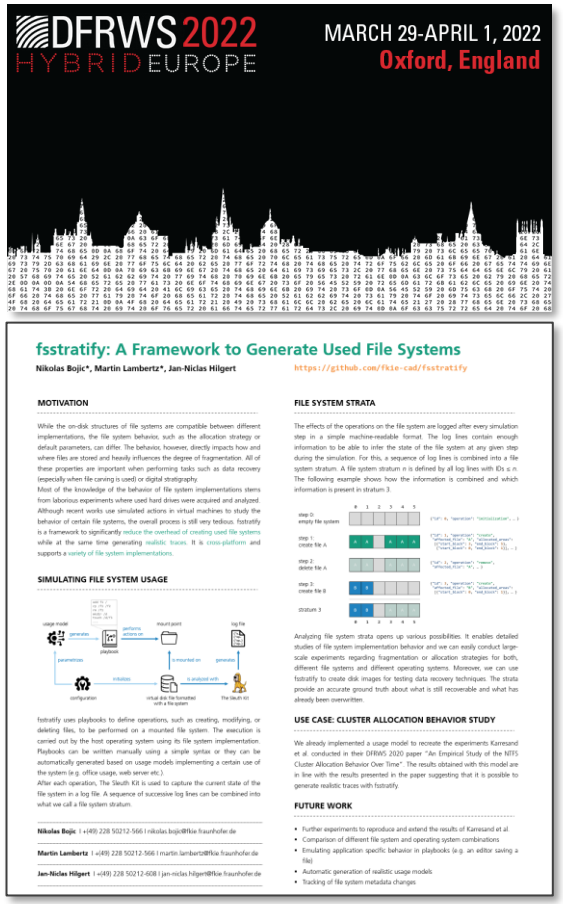
- Code not really publicly available
- Not all required features available
- Indeterminate simulation crashes
- Slow

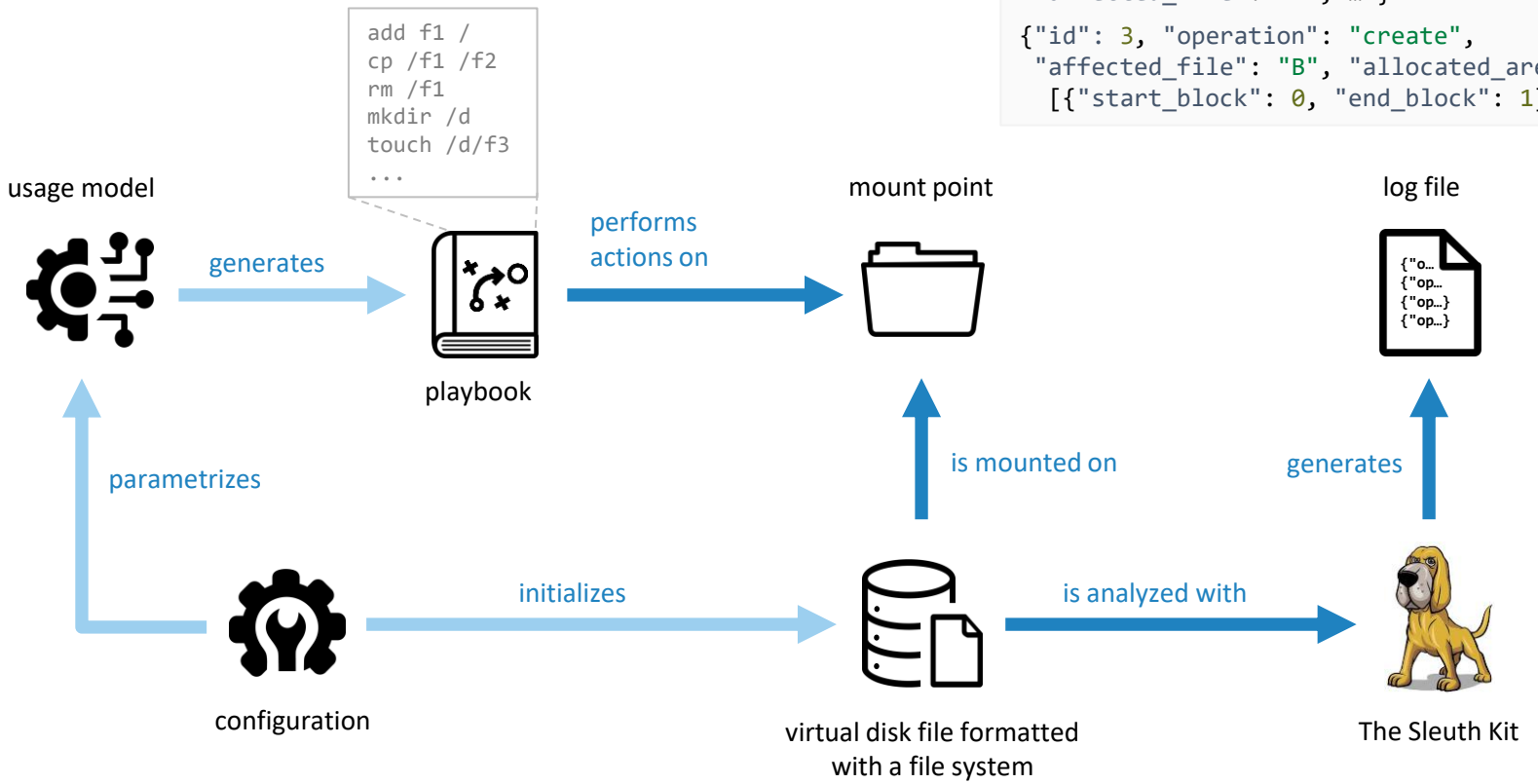
fsstratify



- Code not really publicly available
  - ✓ Code is on GitHub now
- Not all required features available
- Indeterminate simulation crashes
- Slow

fsstratify






```
{"id": 0, "operation": "initialization", … }
{"id": 1, "operation": "create",
 "affected_file": "A", "allocated_areas":
  [{"start_block": 3, "end_block": 5},
   {"start_block": 0, "end_block": 1}], … }
{"id": 2, "operation": "remove",
 "affected_file": "A", … }
{"id": 3, "operation": "create",
 "affected_file": "B", "allocated_areas":
  [{"start_block": 0, "end_block": 1}], … }
```

fsstratify

```
{"id": 0, "operation": "initialization", … }
{"id": 1, "operation": "create",
 "affected_file": "A", "allocated_areas":
  [{"start_block": 3, "end_block": 5},
   {"start_block": 0, "end_block": 1}], … }
{"id": 2, "operation": "remove",
 "affected_file": "A", … }
{"id": 3, "operation": "create",
 "affected_file": "B", "allocated_areas":
  [{"start_block": 0, "end_block": 1}], … }
```

usage model

```
add f1 /
cp /f1 /f2
rm /f1
mkdir /d
touch /d/f3
...
```

mount point

log file

```
{"o…
{"op…
{"op…}
{"op…}
```

playbook

parametrizes

performs

is mounted on

generates

configuration

virtual disk file formatted with a file system

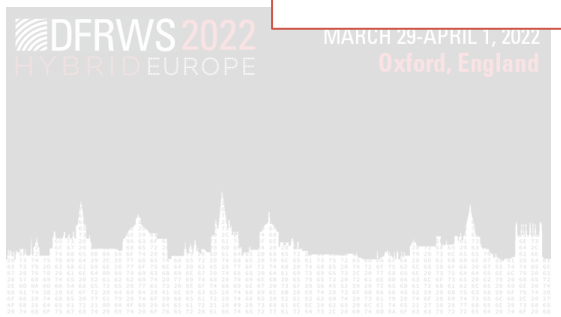
The Sleuth Kit

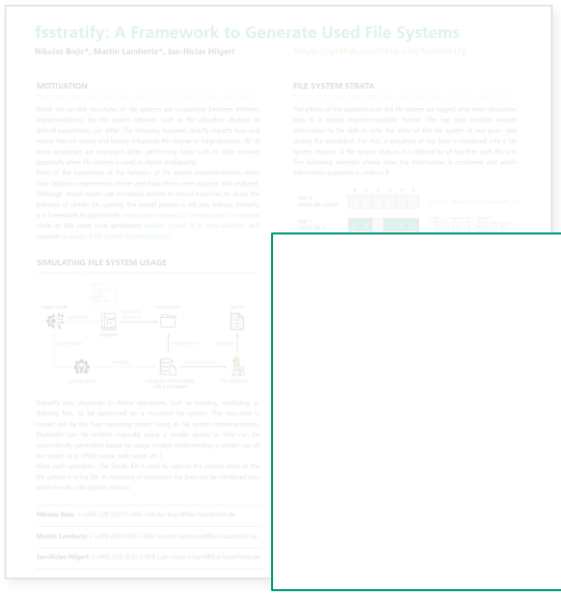
Log file was missing required temporal metadata

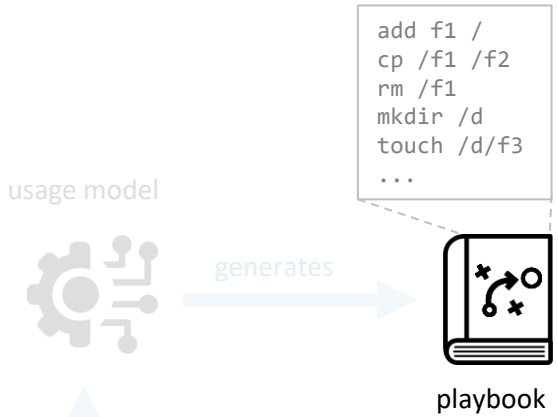
(file-system-specific) metadata is now included

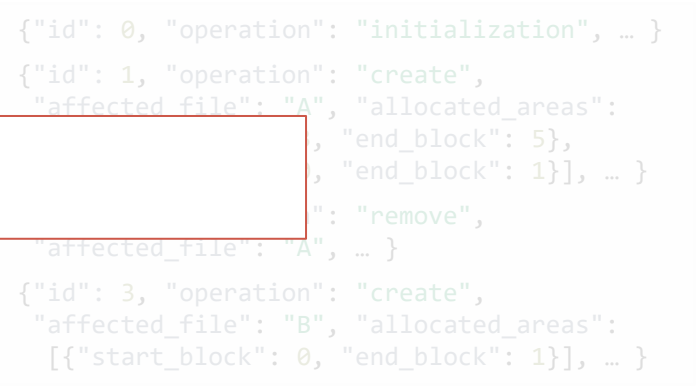
fsstratify

No means to control time









The Sleuth Kit

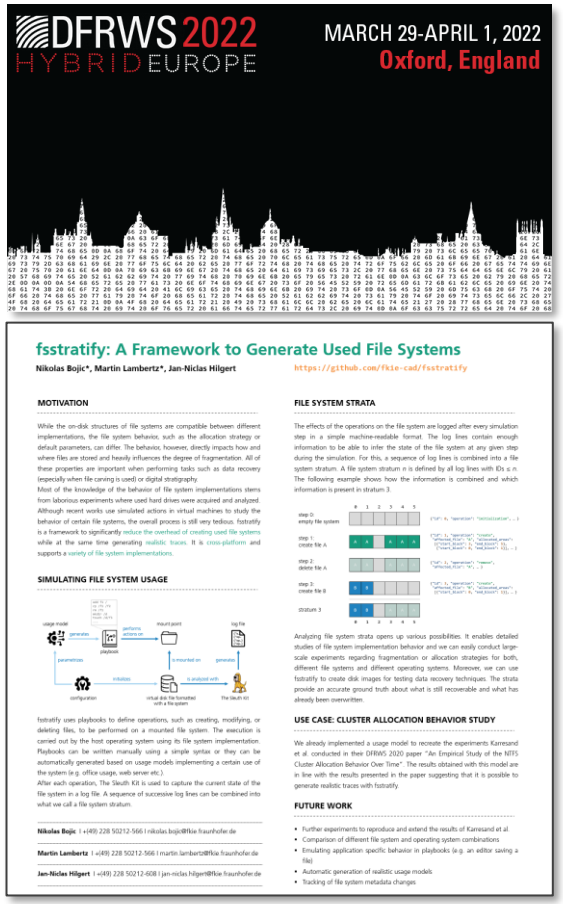
`sleep` operation

(no impact on simulation system; slow)

`time` operation

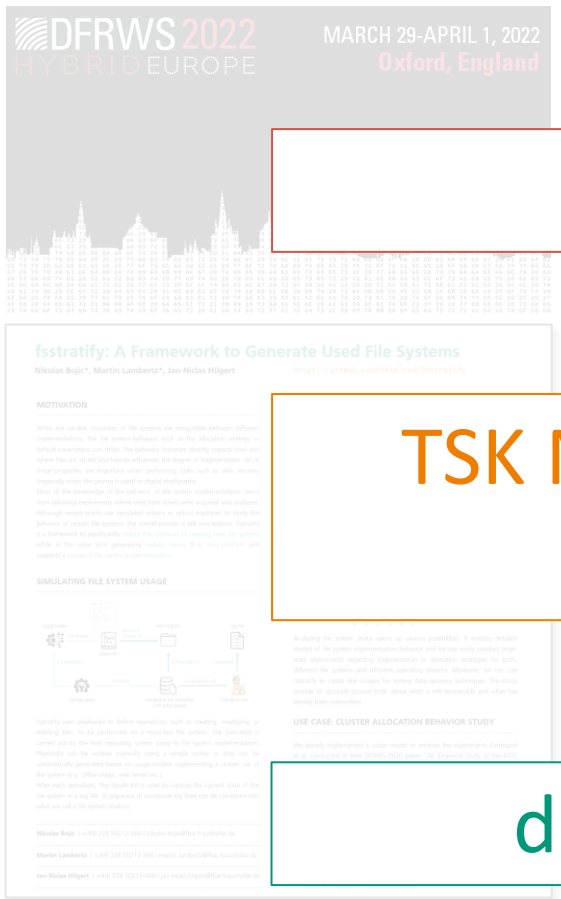
(fast; impact on simulation system)

fsstratify



- Code not really publicly available
  - ✓ Code is on GitHub now
- Not all required features available
  - ✓ Missing features implemented
- Indeterminate crashes
- Slow

fsstratify

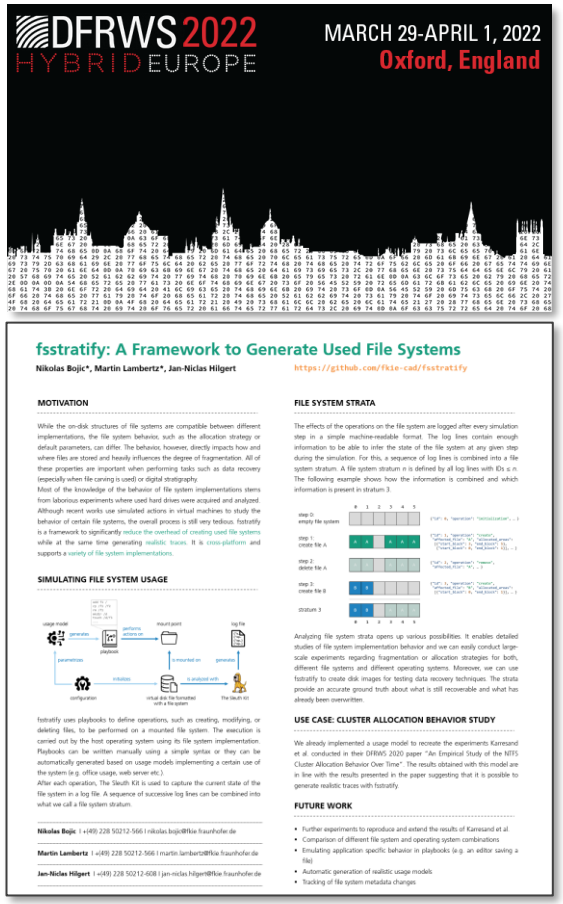


Indeterminate simulation crashes

TSK NTFS parser reported deleted files to be active in some cases

dissect as file system parsing backend

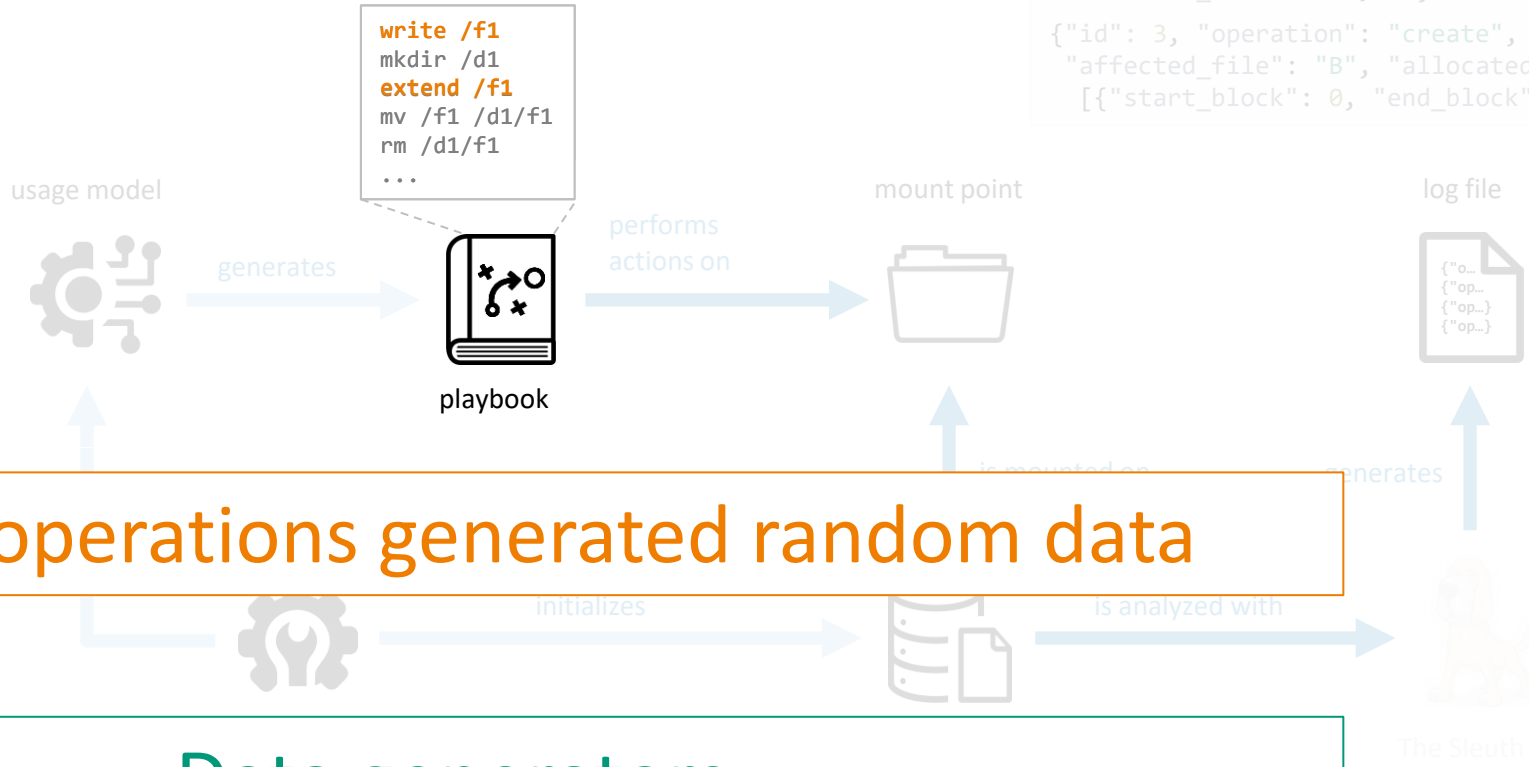
fsstratify

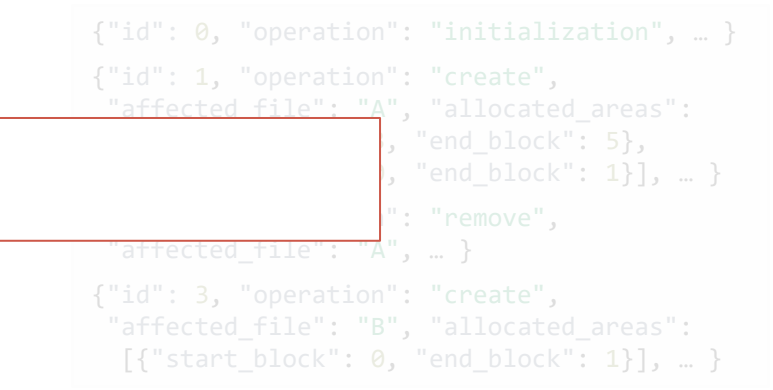


- Code not really publicly available
  - ✓ Code is on GitHub now
- Not all required features available
  - ✓ Missing features implemented
- Indeterminate crashes
  - ✓ Dissect as file system parser
- Slow

fsstratify

Simulations were rather slow



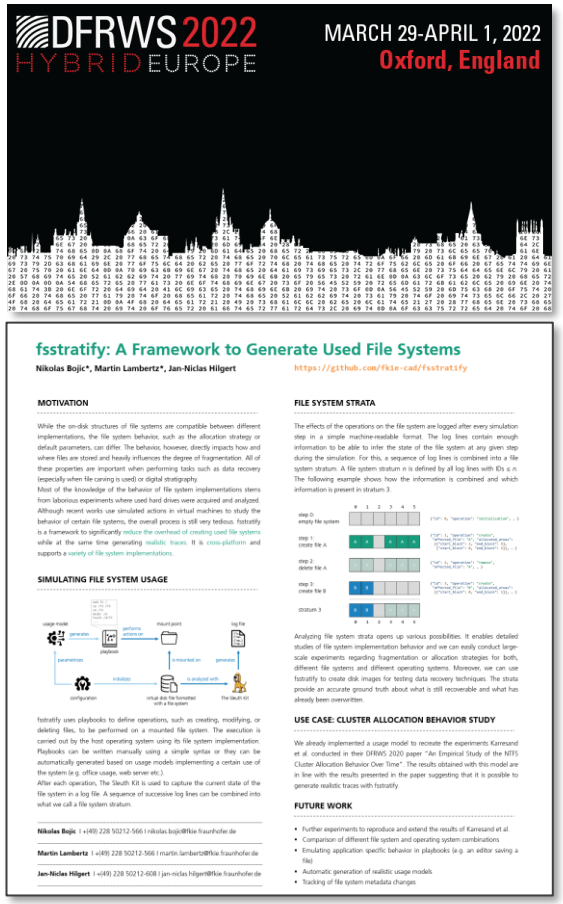


Writing operations generated random data

Data generators

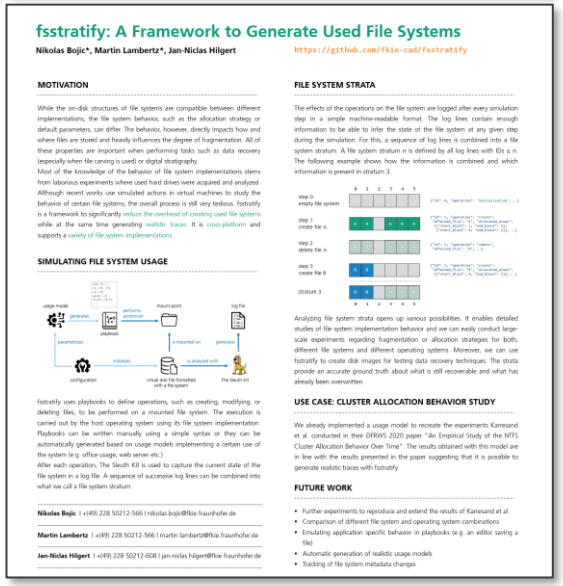
(configurable data; e. g., static, patterns, random)

fsstratify



- Code not really publicly available
  - ✓ Code is on GitHub now
- Not all required features available
  - ✓ Missing features implemented
- Indeterminate crashes
  - ✓ Dissect as file system parser
- Slow
  - ✓ Data generators

fsstratify



fsstratify: A Framework to Generate Used File Systems

Nikolas Bojic*, Martin Lambertz*, Jan-Niclas Hilgert

https://github.com/fkie-cad/fsstratify

MOTIVATION

While the on-disk structures of file systems are compatible between different implementations, the file system behavior, such as the allocation strategy or default parameters, can differ. The behavior, however, directly impacts how and where files are stored and heavily influences the degree of fragmentation. All of these properties are important when performing tasks such as data recovery (especially when file carving is used) or digital stratigraphy.

Most of the knowledge of the behavior of file system implementations stems from laborious experiments where used hard drives were acquired and analyzed. Although recent works use simulated actions in virtual machines to study the behavior of certain file systems, the overall process is still very tedious. fsstratify is a framework to significantly reduce the overhead of creating used file systems while at the same time generating realistic traces. It is cross-platform and supports a variety of file system implementations.

SIMULATING FILE SYSTEM USAGE

fsstratify uses playbooks to define operations, such as creating, modifying, or deleting files, to be performed on a mounted file system. The execution is carried out by the host operating system using its file system implementation. Playbooks can be written manually using a simple syntax or they can be automatically generated based on usage models implementing a certain use of the system (e.g. office usage, web server etc.).

After each operation, The Sleuth Kit is used to capture the current state of the file system in a log file. A sequence of successive log lines can be combined into what we call a file system stratum.

FILE SYSTEM STRATA

The effects of the operations on the file system are logged after every simulation step in a simple machine-readable format. The log lines contain enough information to be able to infer the state of the file system at any given step during the simulation. For this, a sequence of log lines is combined into a file system stratum. A file system stratum n is defined by all log lines with IDs $\leq n$. The following example shows how the information is combined and which information is present in stratum 3.

Analyzing file system strata opens up various possibilities. It enables detailed studies of file system implementation behavior and we can easily conduct large-scale experiments regarding fragmentation or allocation strategies for both, different file systems and different operating systems. Moreover, we can use fsstratify to create disk images for testing data recovery techniques. The strata provide an accurate ground truth about what is still recoverable and what has already been overwritten.

USE CASE: CLUSTER ALLOCATION BEHAVIOR STUDY

We already implemented a usage model to recreate the experiments Karresand et al. conducted in their DFRWS 2020 paper "An Empirical Study of the NTFS Cluster Allocation Behavior Over Time". The results obtained with this model are in line with the results presented in the paper suggesting that it is possible to generate realistic traces with fsstratify.

FUTURE WORK

- Further experiments to reproduce and extend the results of Karresand et al.
- Comparison of different file system and operating system combinations
- Emulating application specific behavior in playbooks (e.g. an editor saving a file)
- Automatic generation of realistic usage models
- Tracking of file system metadata changes

Nikolas Bojic | +(49) 228 50212-566 | nikolas.bojic@fkie.fraunhofer.de
Martin Lambertz | +(49) 228 50212-566 | martin.lambertz@fkie.fraunhofer.de
Jan-Niclas Hilgert | +(49) 228 50212-608 | jan-niclas.hilgert@fkie.fraunhofer.de

- Code not really publicly available
  - ✓ Code is on GitHub now
- Not all required features available
  - ✓ Missing features implemented
- Indeterminate crashes
  - ✓ Dissect as file system parser
- Slow
  - ✓ Data generators

Corpus for temporal analysis

Multiple disk image sizes

74 image files in 12 categories

Test data without restrictions on use or distribution

Detailed documentation

NTFS filesystem

Not representative of real-world usage

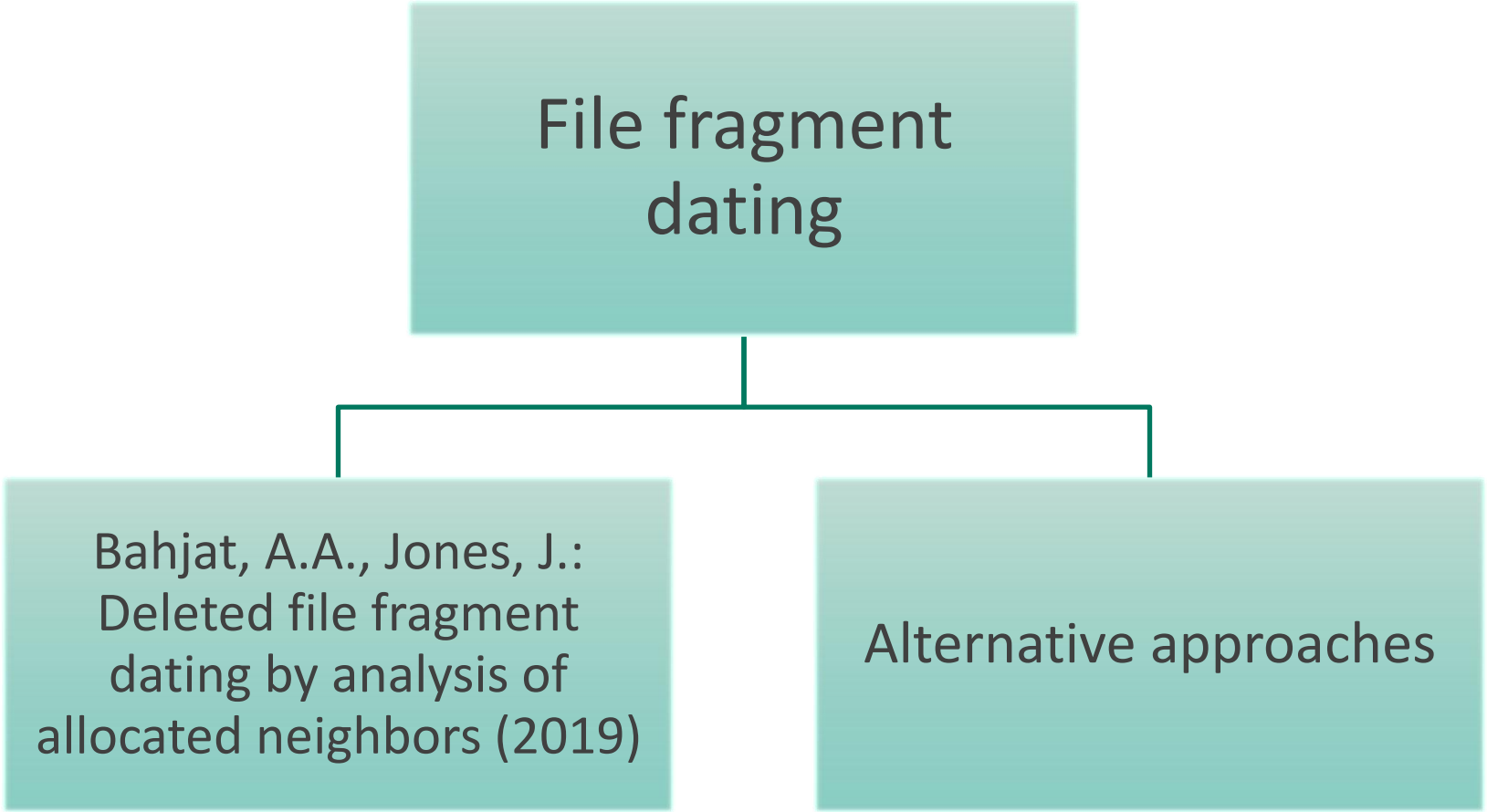
Corpus categories

- Basic Images
  - Write files until capacity
  - Deleted files
- Operation-specific Images
  - Write, copy, move, extend
  - Deleted fragments in file slacks

Corpus categories

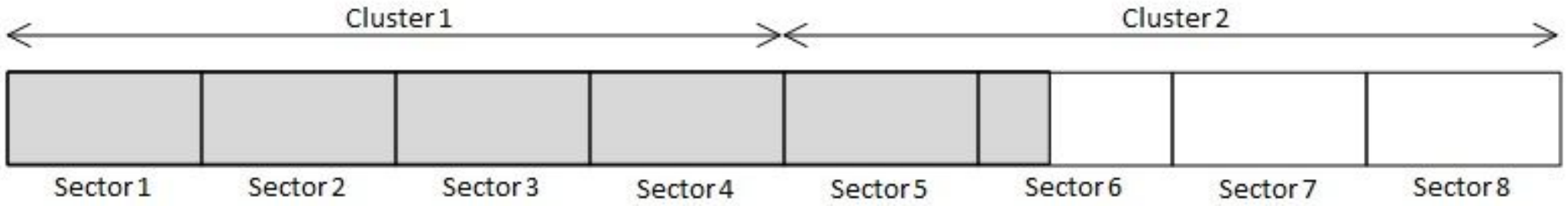
- Overwrite-specific Images
  - Overwrite existing files by write, move, copy
  - Deleted fragments in file slacks
- Images with high degree of fragmentation
  - Deleted fragments in file slacks
- Images with manipulated timestamps

Corpus evaluation



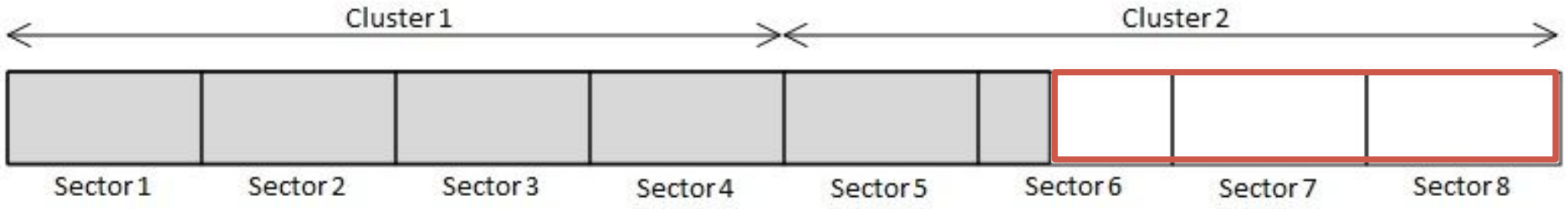
File fragment dating

- Fragments in slack space of another file



File fragment dating

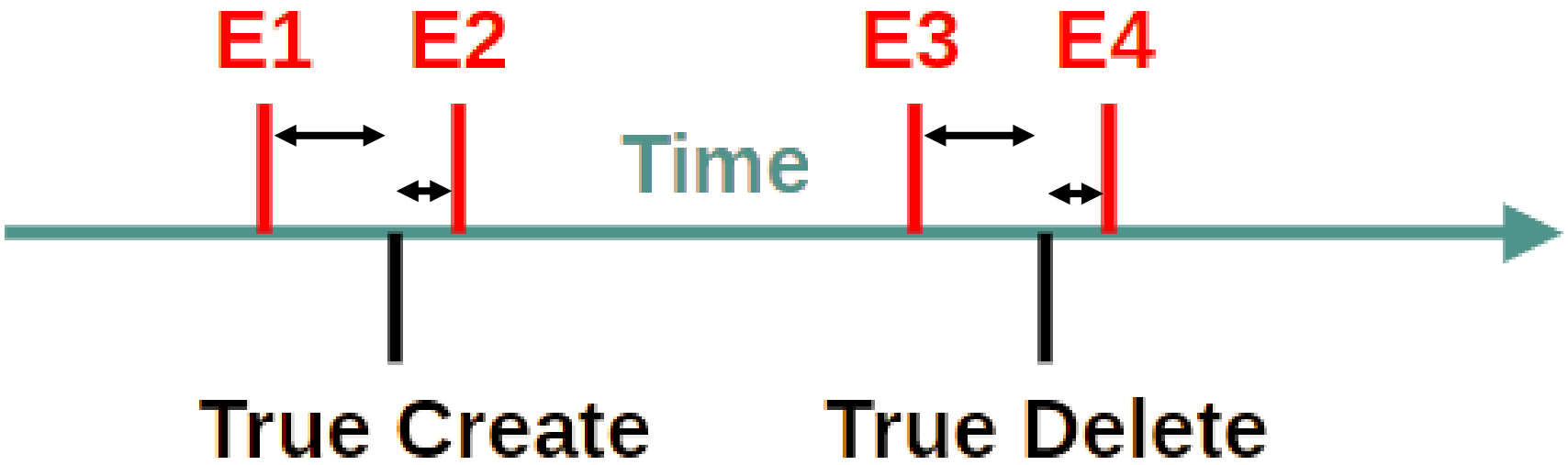
- Fragments in slack space of another file



File Slack

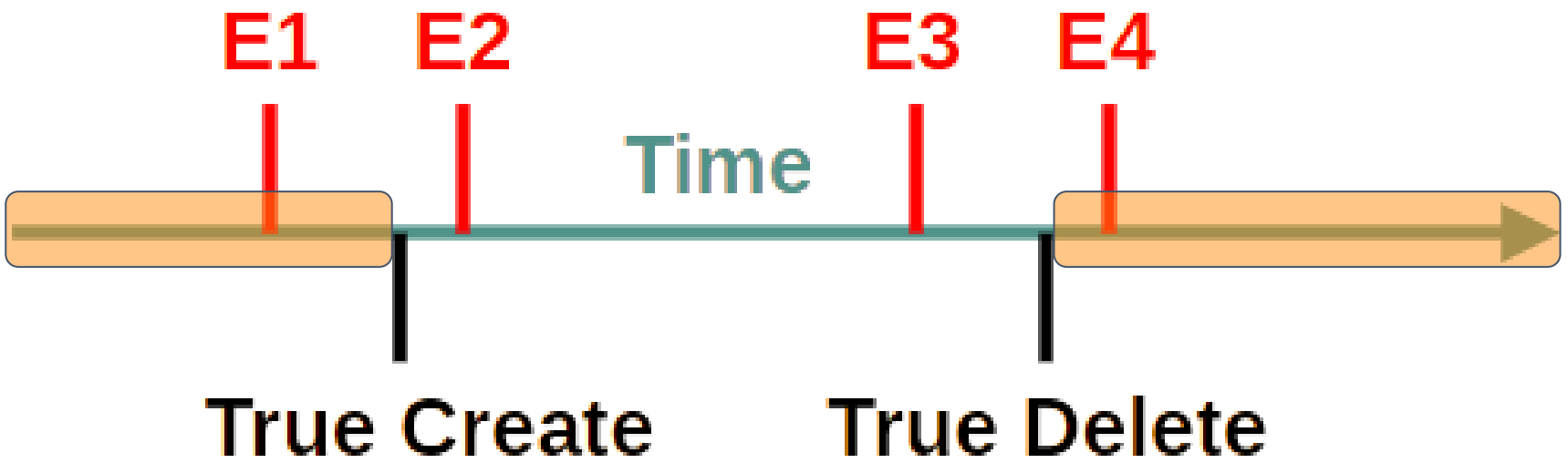
File fragment dating

- Lifetime estimation
- Evaluation metric: **Accuracy**
  - Absolute difference from the true value



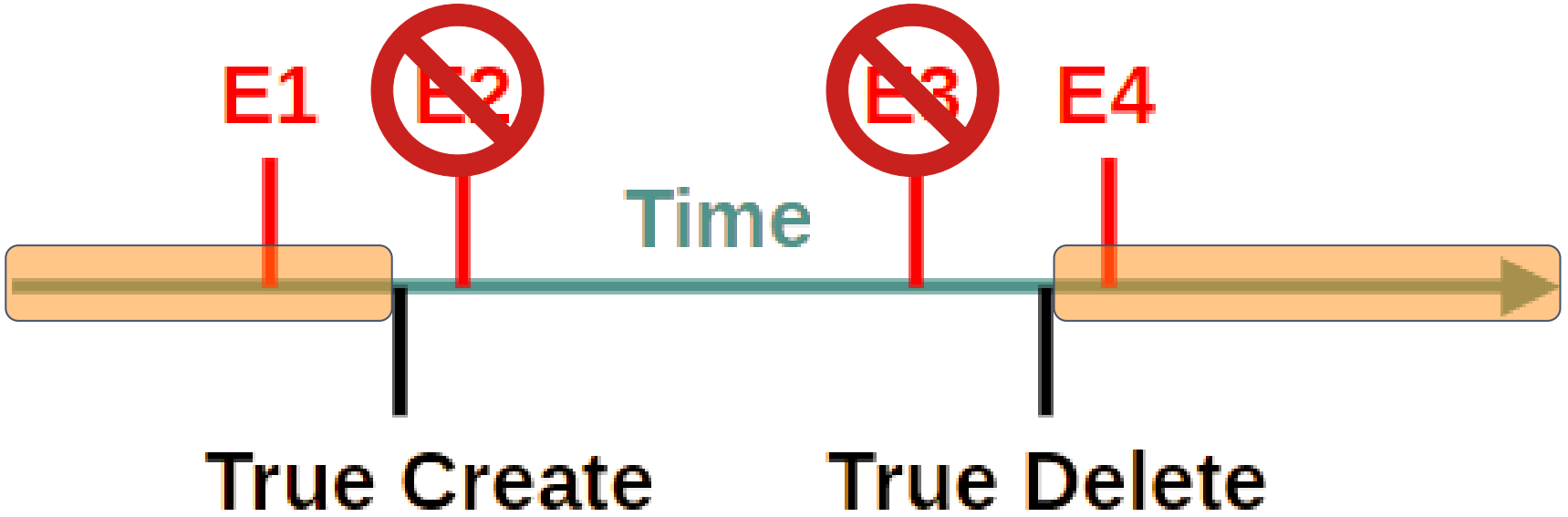
File fragment dating

- Lifetime bounding
- Evaluation metric: **Precision**
  - Estimate precedes True Create / occurs after True Delete



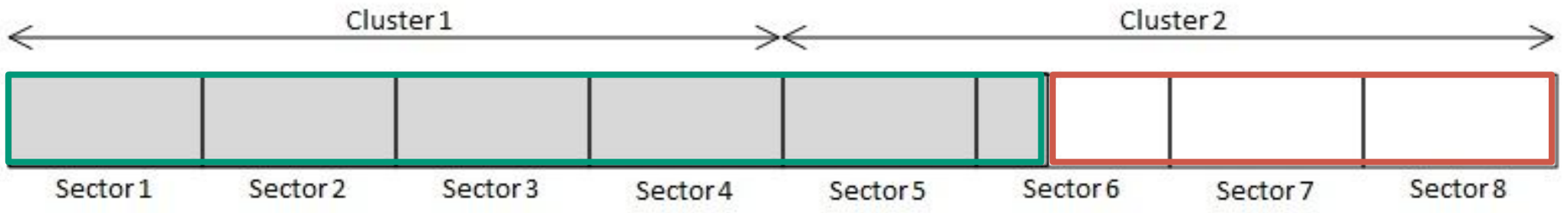
File fragment dating

- Lifetime bounding
- Evaluation metric: **Precision**
  - Estimate precedes True Create / occurs after True Delete



Bahjat and Jones (2019)

- Estimation of lower and upper bounds for a deleted file
- Upper bound calculation: Maximum out of eight dates of slack owner file



Slack Owner

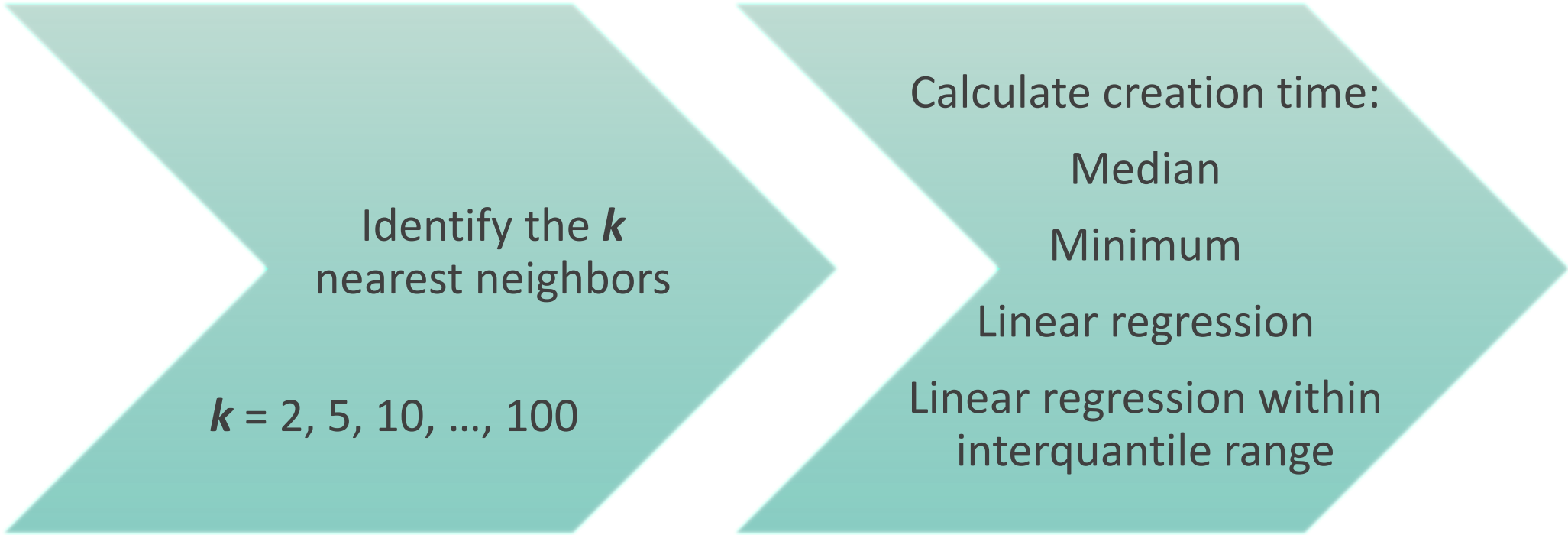
File Slack

Bahjat and Jones (2019)

- Estimation of lower and upper bounds for a deleted file
- Lower bound calculation:

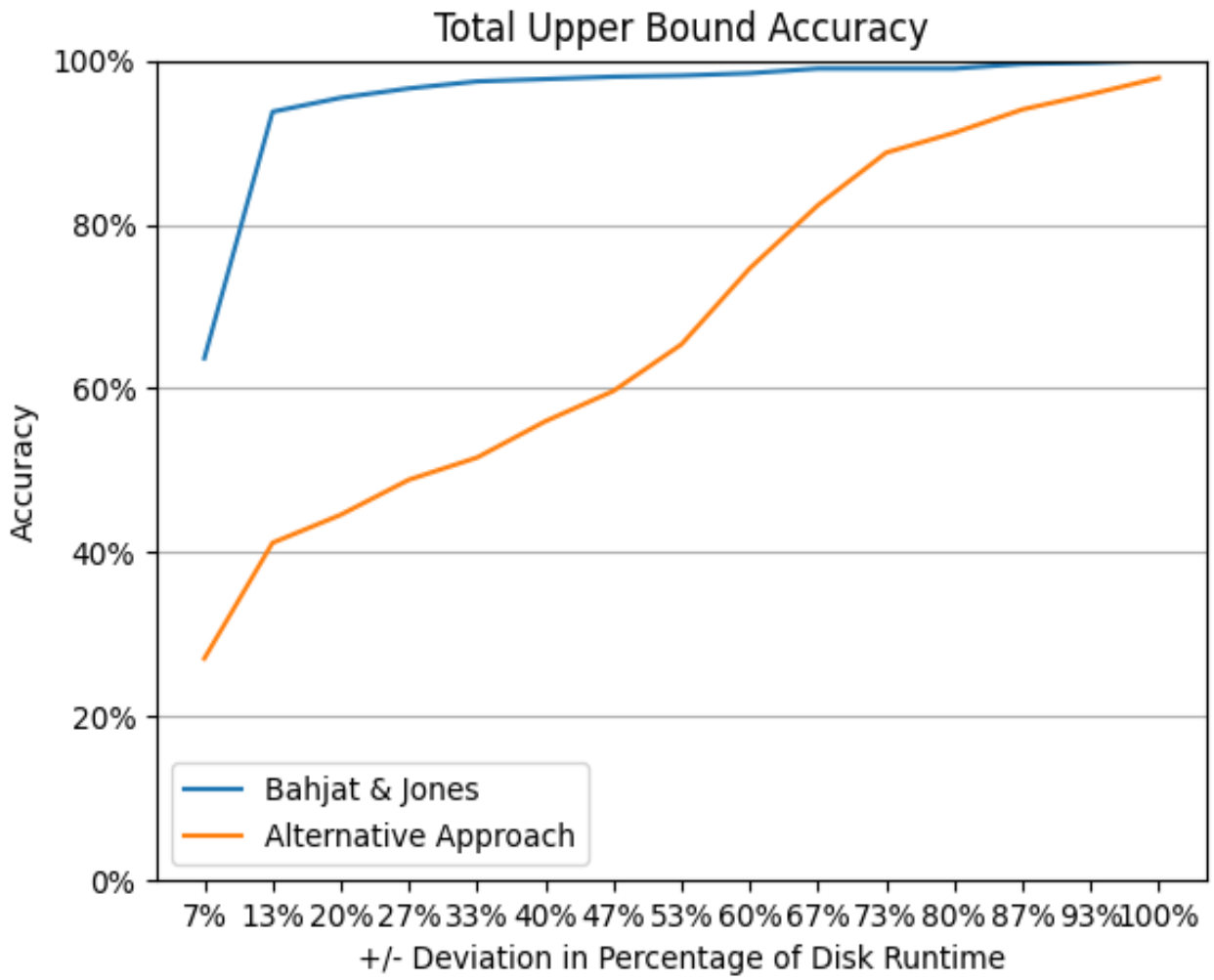
Alternative approaches

- Naive upper bound: Maximum of creation timestamps
- Lower bound calculation:



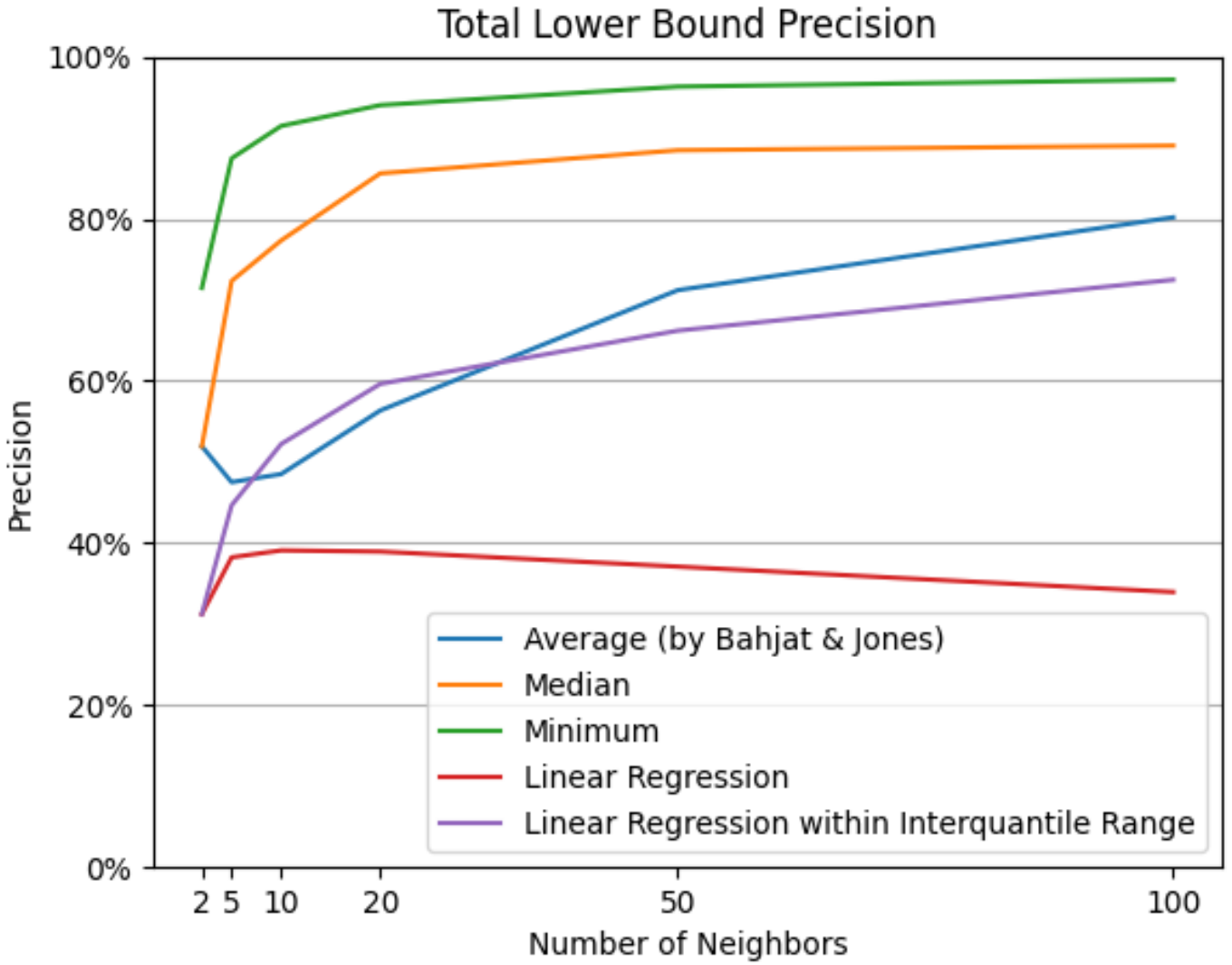
Results

- Upper bound precision:
  - Bahjat and Jones: 100 %
  - Naive approach: 9.5 % to 80 %
- Upper bound accuracy



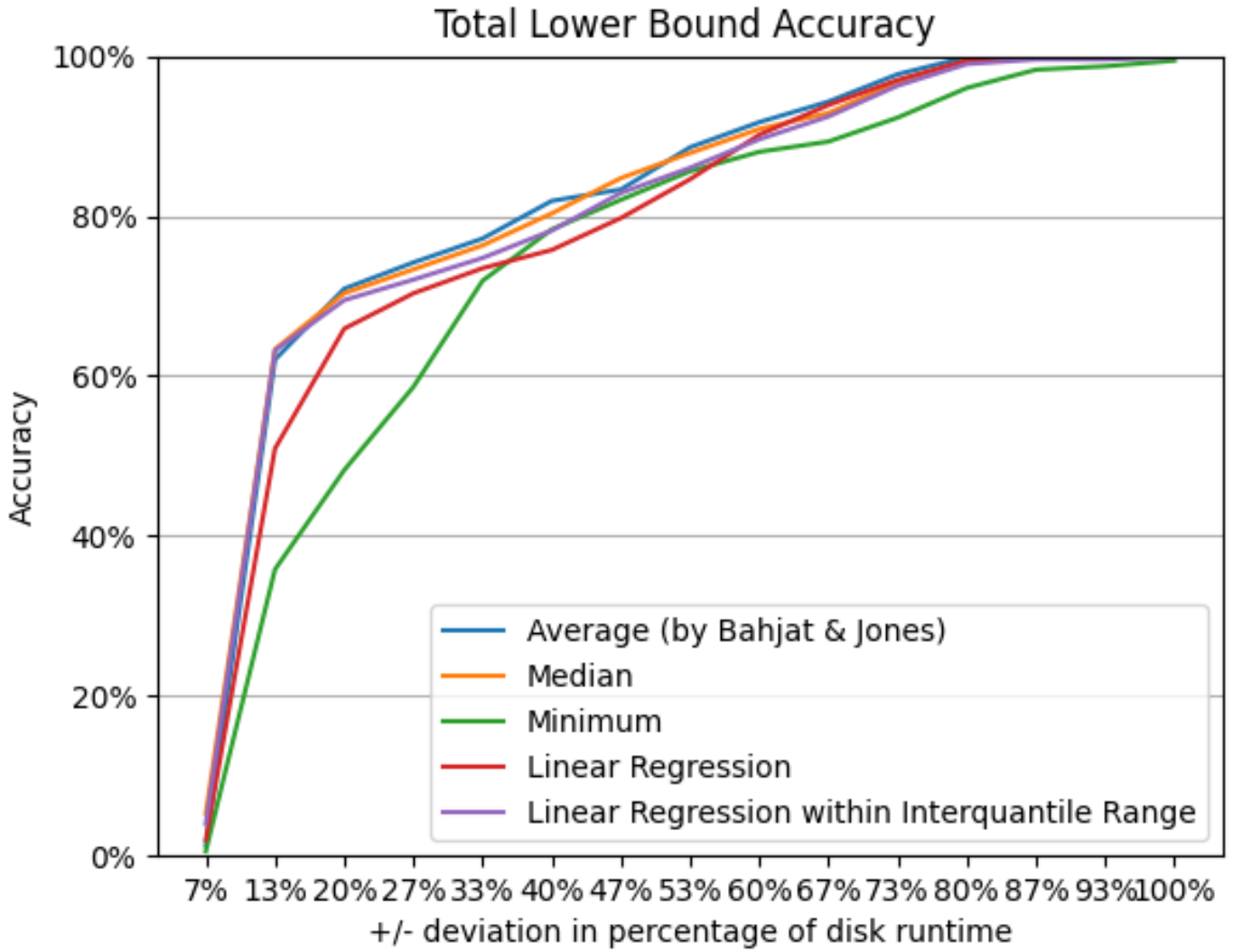
Results

- Lower bound precisior

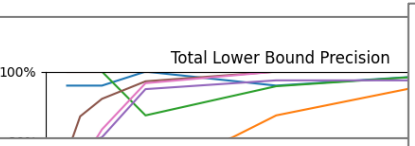


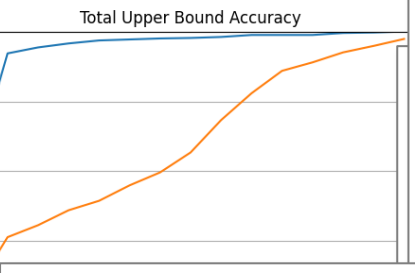
Results

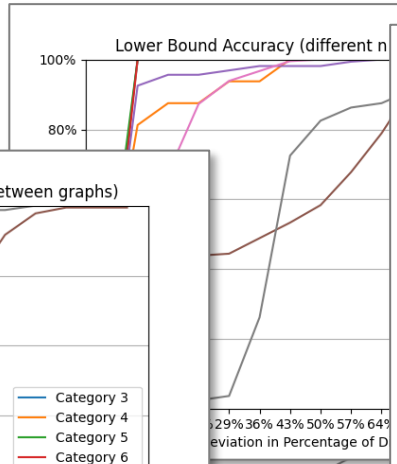
- Lower bound accuracy

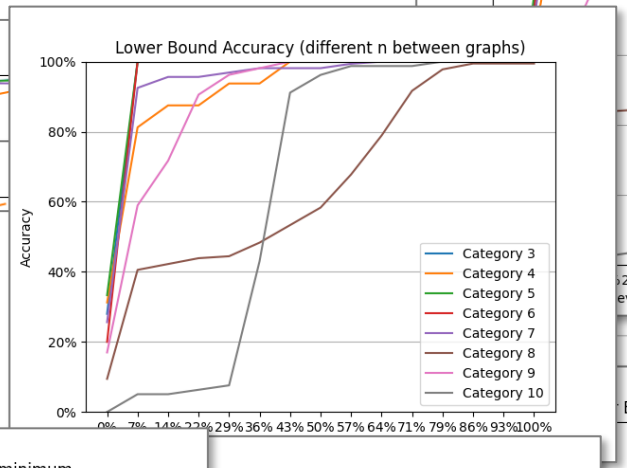


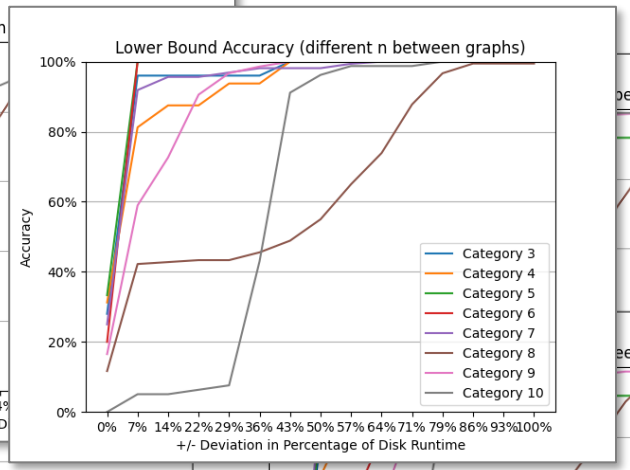
More Results :-)

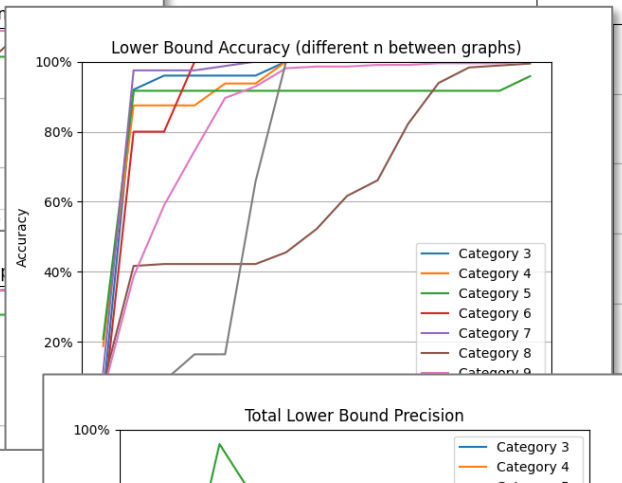


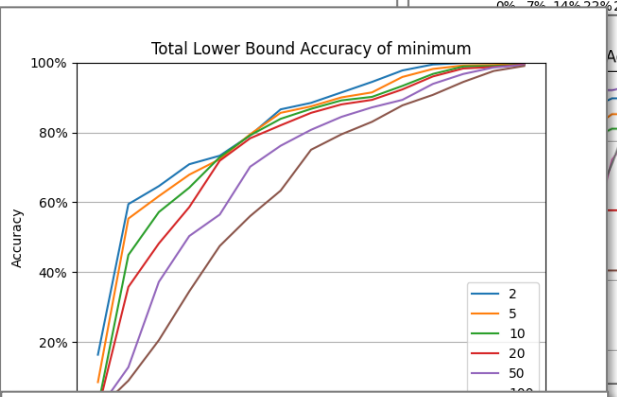


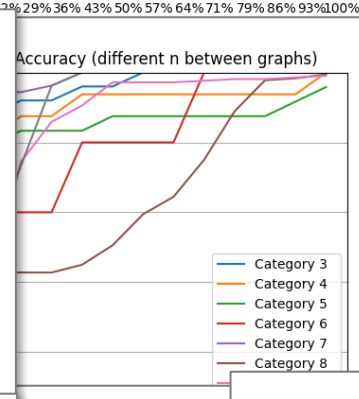


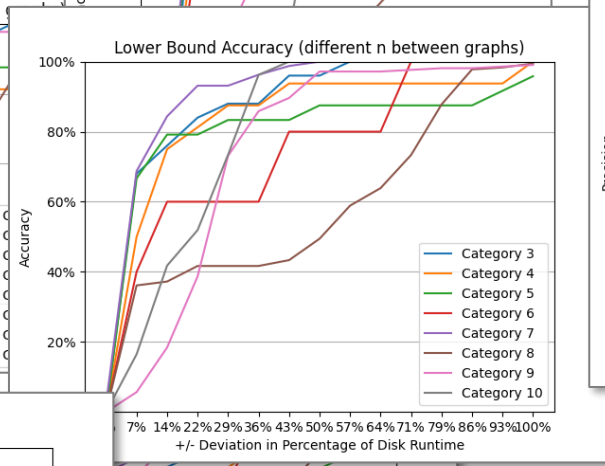


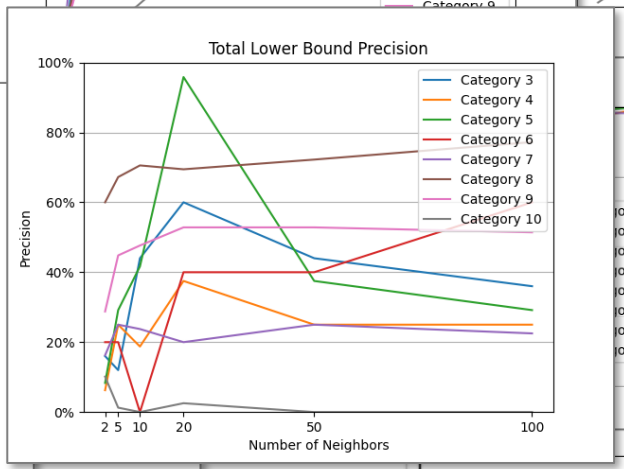


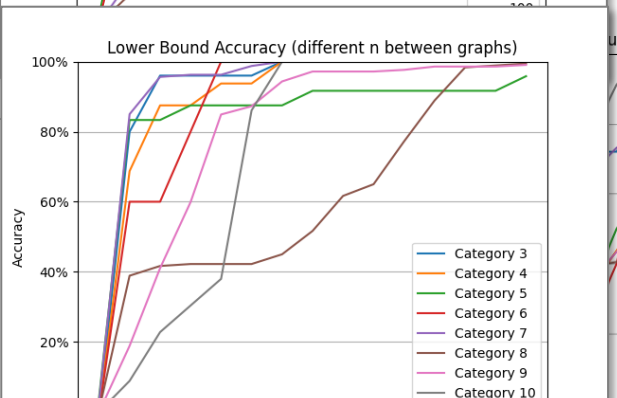


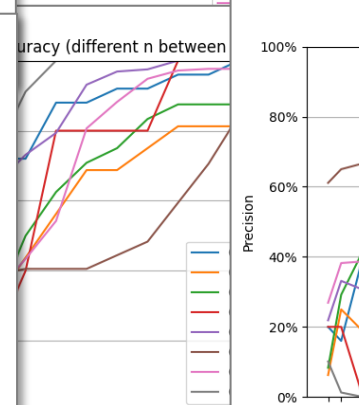


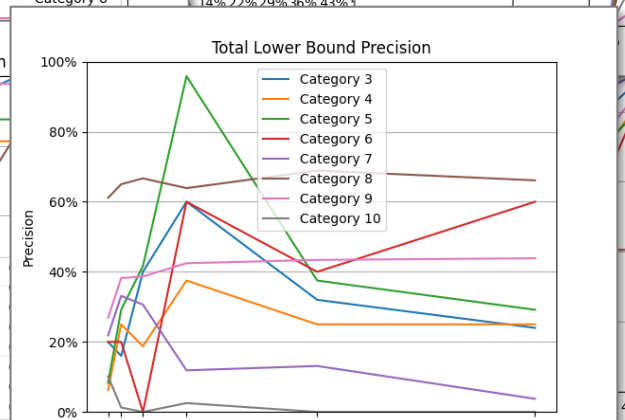


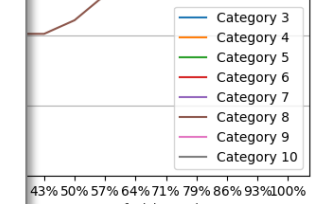


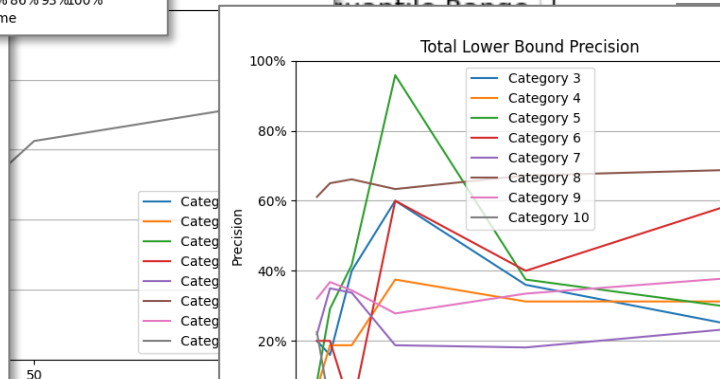




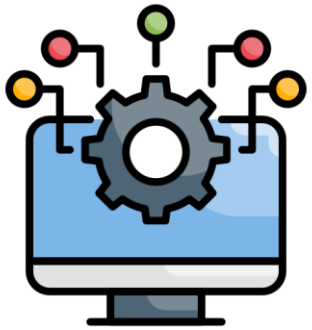




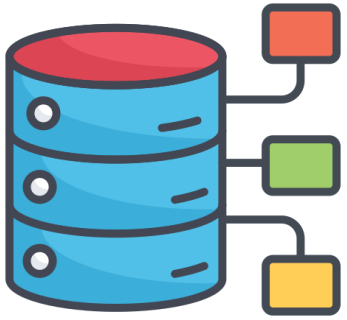




Digital Stratigraphy: Contributions



fsstratify



Dataset

Evaluation

We revised and extended fsstratify, a framework to generate „used“ file systems,...

…created a corpus of NTFS file system images for chronological dating,…

…and evaluated the method proposed by Bahjat and Jones.

2019

Bahjat, Jones: „Deleted file fragment dating by analysis of allocated neighbors”

[Icons created by Design Circle – Flaticon]

Thanks for listening! Any questions?

Julian Uthoff
<julian.uthoff@mailbox.org>

Lisa Marie Dreier
<lisa.dreier@fau.de>

Martin Lambertz
<martin.lambertz@fkie.fraunhofer.de>

Mariia Rybalka
<mariia.rybalka@fkie.fraunhofer.de>

Felix Freiling
<felix.freiling@fau.de>

doi.org/10.5281/zenodo.16637418

github.com/fkie-cad/fsstratify

FAU

Fraunhofer FKIE

[Icons created by Design Circle – Flaticon]