InDeF: An Advanced Defragmenter Supporting Migration Offloading on ZNS SSD

Wenjie Qi, Zhipeng Tan, Jicheng Shao, Lihua Yang, Yang Xiao

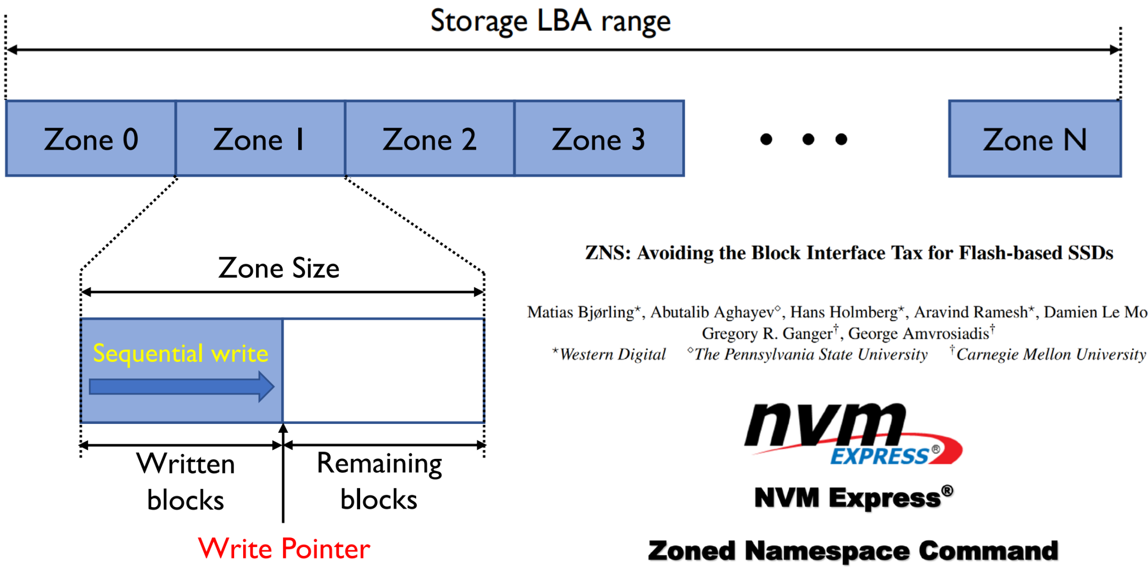
Wuhan National Laboratory for Optoelectronics,
Huazhong University of Science and Technology (HUST), China

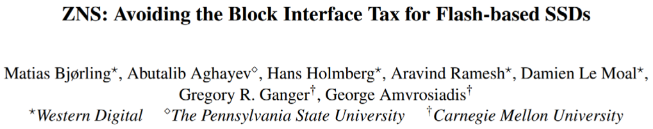
Outline

- **Background and motivation**
- **Our Work: InDeF**
- **Performance Evaluation**
- **Conclusion**

Background : What is ZNS SSD?

- ➢The logical address space is divided into fixed-sized zones
- ➢Each zone must be written sequentially and reset explicitly for reuse

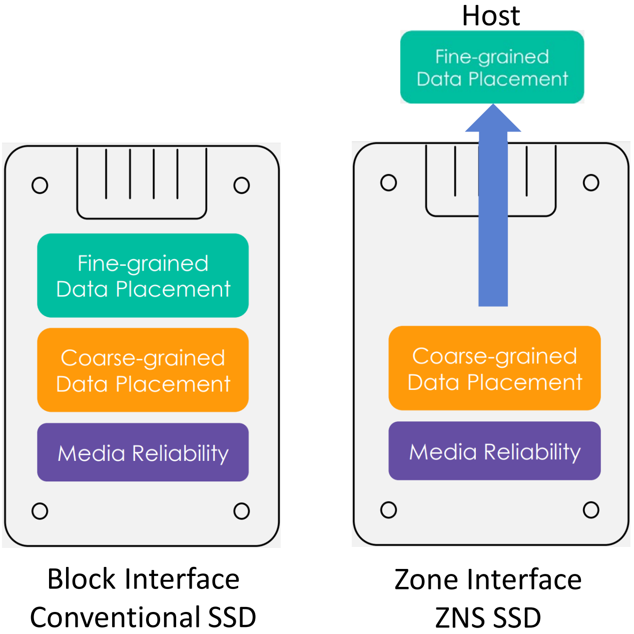



ZNS: Avoiding the Block Interface Tax for Flash-based SSDs

Matias Bjørling*, Abutalib Aghayev◇, Hans Holmberg*, Aravind Ramesh*, Damien Le Moal*,
Gregory R. Ganger†, George Amvrosiadis†
*Western Digital ◇The Pennsylvania State University †Carnegie Mellon University

NVM Express®

Zoned Namespace Command Set Specification


Host
Fine-grained Data Placement
Fine-grained Data Placement
Coarse-grained Data Placement
Media Reliability
Coarse-grained Data Placement
Media Reliability
Block Interface Conventional SSD
Zone Interface ZNS SSD

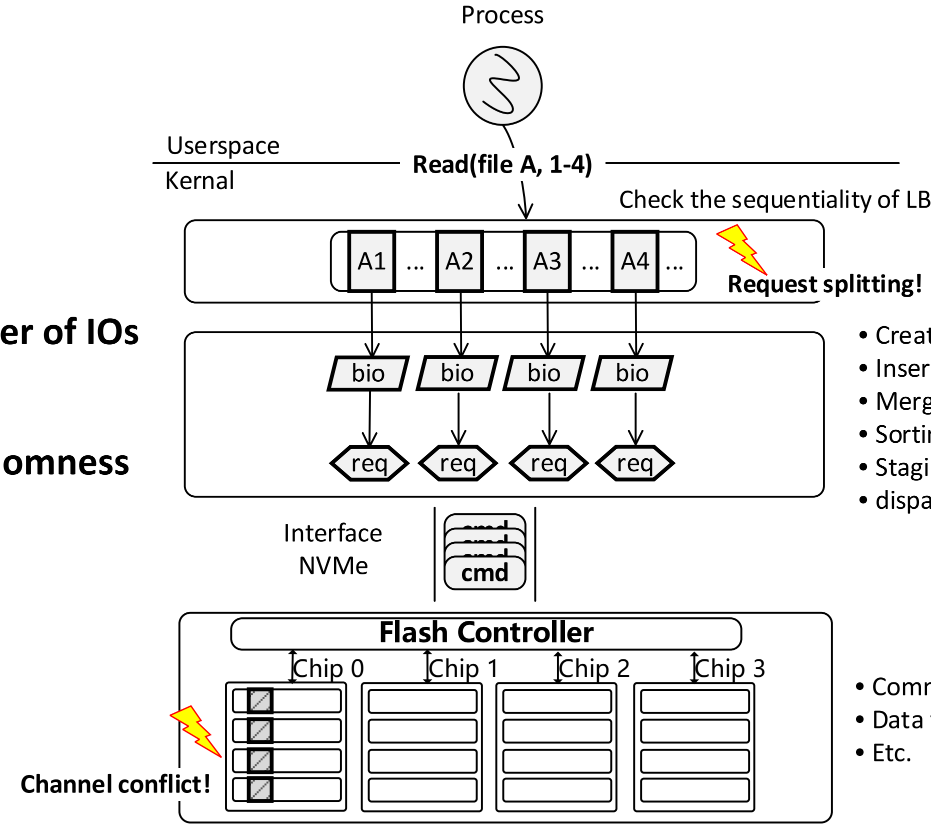
Background : What is fragmentation?

Request splitting

1)Increases the number of IOs

2)Makes I/Os smaller

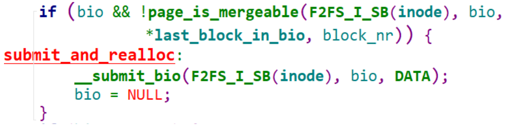
3)Increases their randomness



Check the sequentiality of LBA

Request splitting!

Channel conflict!



```
if (bio && !page_is_mergeable(F2FS_I_SB(inode), bio,
                        *last_block_in_bio, block_nr)) {
submit_and_realloc:
        __submit_bio(F2FS_I_SB(inode), bio, DATA);
        bio = NULL;
}
```

- Creation
- Insertion
- Merging
- Sorting
- Staging
- dispatching

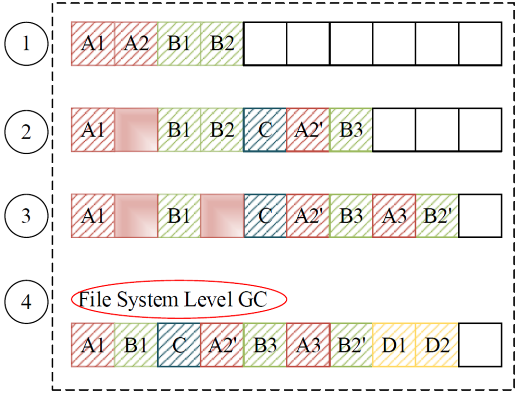
Logical Fragmentation

Physical Fragmentation

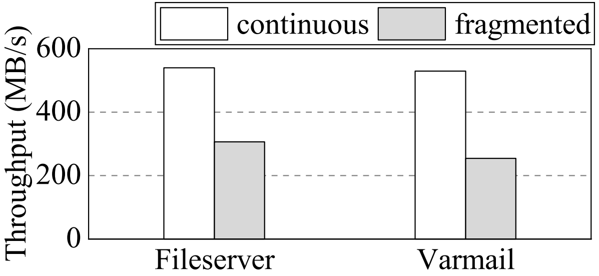
- Command processing
- Data fetching(DMA)
- Etc.

Motivation : Fragmentation accumulation

□ **Free Page** ▨ **Valid Page** 🟥 **Invalid Page**



- **Incoming Data Stream：**
  - **A1A2, B1B2 → C, A2', B3 → A3, B2' → D1D2**



- **Fragmentation on ZNS SSD**

  Sequential read after running Fileserver/Varmail

  Filesystem: F2FS

Motivation : Definition of the fragmentation

- **Definition of the fragmentation of an I/O request**
  - **The degree of logical fragmentation (DoLF) is the number of logical fragments in an I/O range**

  - **The degree of physical fragmentation (DoPF)**

    We measure the degree of physical fragmentation of an I/O request by how evenly the data in the I/O range are distributed among the flash parallel units.

$$DoPF = \frac{\sum_{i=1}^{L} (N_i - \frac{M}{L})^2}{L}$$

Motivation : The impact of fragmentation

➢Evaluation Setup

- ➢32KB 64KB 128KB 256KB O_DIRECT sequential read on F2FS
- ➢Varying DoLF/DoPF

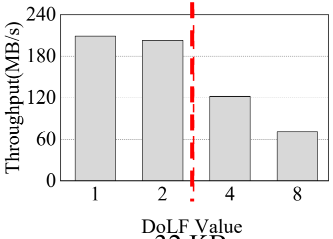
➢Observations

- ➢**A low DoLF value has a small impact on I/O performance**

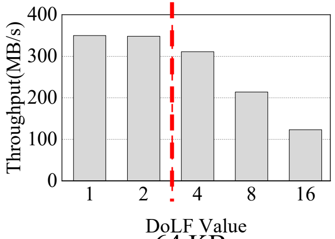
  Low kernel overhead

- ➢**When the DoPF value is small, e.g., less than 40, physical fragmentation has a negligible impact on I/O performance**

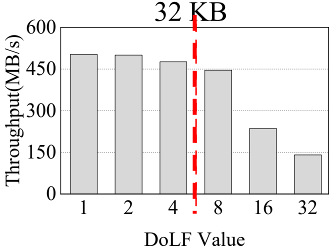
  Software overhead of I/O dominates the total I/O latency



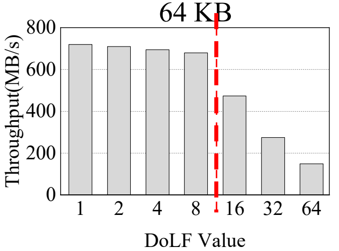
32 KB



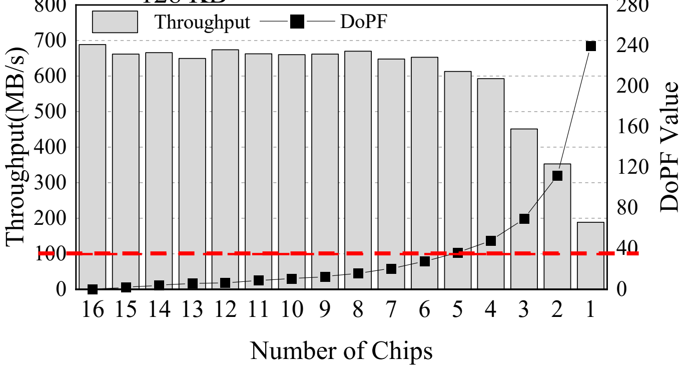
64 KB



128 KB



256 KB



No need to defragment all the fragments!

Motivation : How to Select the Appropriate Data

- **Modern storage systems typically perform data access in non-uniform distribution [1],[2]**
- **Multiple reads to fragmented data accumulate the access latency caused by fragmentation**
- **Our idea:**
  - Defragmenting data with a low degree of fragmentation or cold data that is rarely accessed provides little performance gain
  - **We define the I/O data defragmentation priority (IODP)**

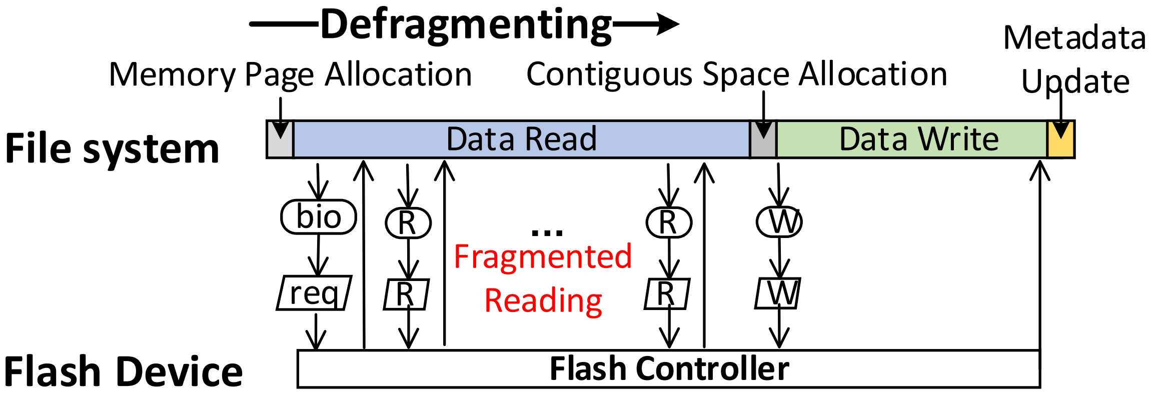
$$IODP = (\alpha \cdot DoLF + \beta \cdot DoPF) \times readcount$$

[1] Q. Wang, J. Li, P. P. Lee, T. Ouyang, C. Shi, and L. Huang, "Separating data via block invalidation time inference for write amplification reduction in log-structured storage," in *Proc. of USENIX FAST*, 2022.

[2] Y. Lv, L. Shi et al., "Access characteristic guided partition for read performance improvement on solid state drives," in 2020 57th ACM/IEEE Design Automation Conference (DAC). IEEE, 2020, pp. 1–6.

Motivation : The Conventional Defragmenter

- Cause a significant increase in the **host memory usage** and invoke page frame reclamation
- Result in a large **chip idle interval** in the SSD
- Migrate **the entire contents of files** even when there are few fragments
- The additional writes **reduce the lifespan** of modern storage devices
- **Degrades the performance** of co-running applications
- Time-consuming

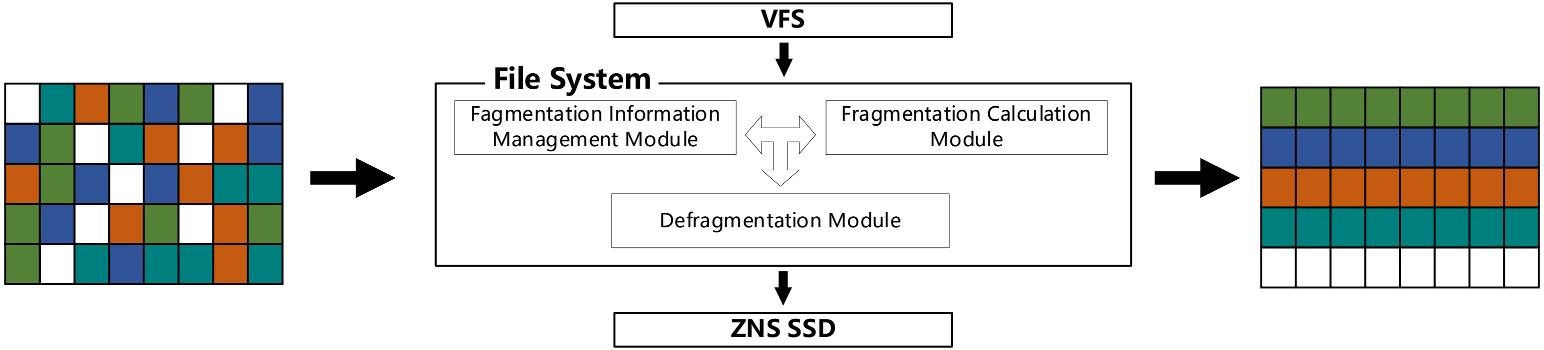


Our Scheme: InDeF for Zoned Namespace SSD

➢Main goals

- ✓ Minimizes the amount of migration data for defragmentation to reduce the write traffic of the underlying device
- ✓ Decreases elapsed time of defragmentation to reduce the impact on co-running application performance

InDeF for Zoned Namespace SSD


VFS
File System
Fagmentation Information Management Module
Fragmentation Calculation Module
Defragmentation Module
ZNS SSD

➢**The Fragmentation Information Management Module**

Collects I/O information from the filesystem

Manages I/O fragmentation information

➢**The Fragmentation Calculation Module**

Calculates the DoLF value and DoLP value based on the collected I/O information

➢**The Defragmentation Module**

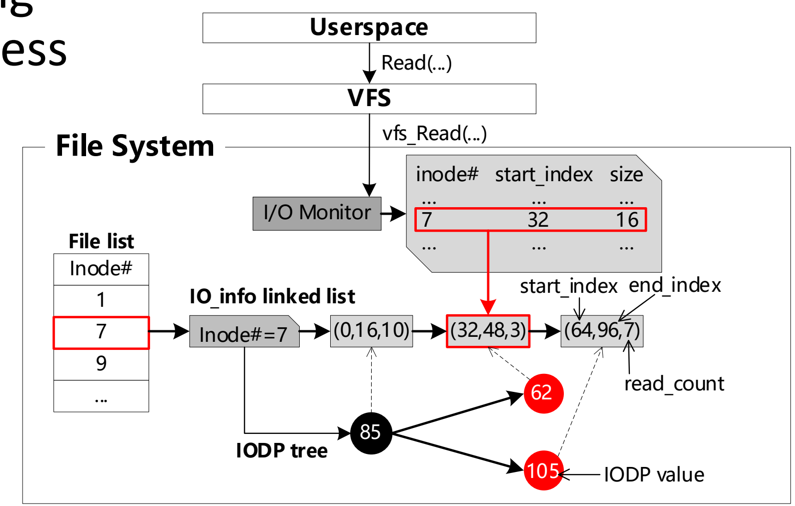
Filters the fragments based on the I/O fragmentation information

Offloads the data migration from the host to the SSD

InDeF for Zoned Namespace SSD

The Fragmentation Information Management Module

- **Monitors** I/O activity at the file system layer
- **Creates** file list and I/O information linked list
- **Merges** I/O requests that have overlapping addresses and preserves the largest address range
- **Inserts** the IODP of each I/O information into a red-black tree (called IODP tree) in order



inode#	start_index	size
...	...	...
7	32	16
...	...	...

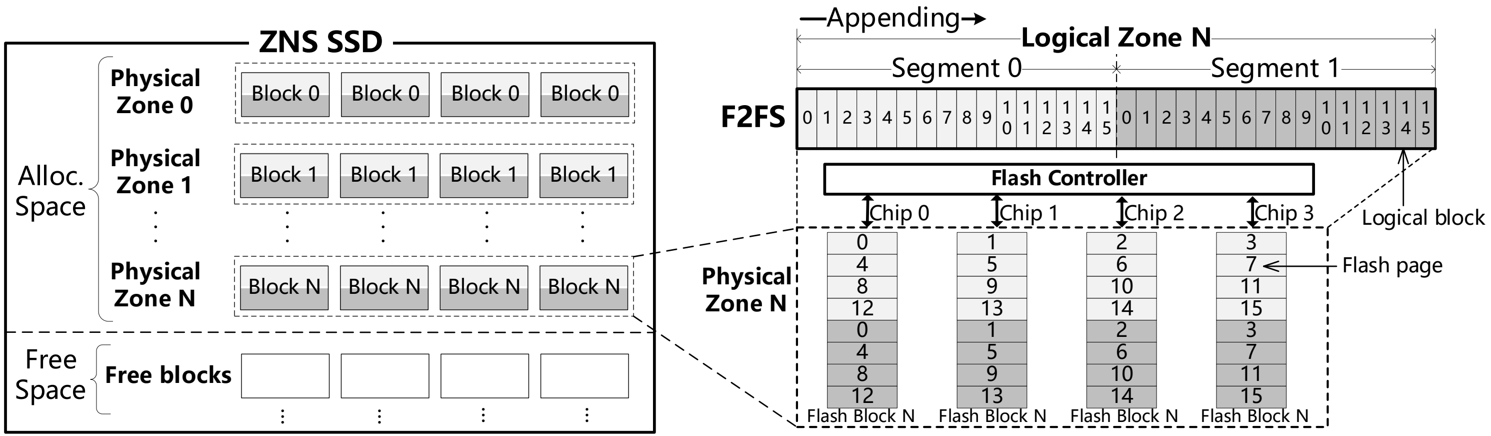
Inode#
1
7
9
...

InDeF for Zoned Namespace SSD

➢The Fragmentation Calculation Module

- ➢Gets the physical location of the logical block inside the device by the segment number and in-segment offset of the logical block

- ➢The i-th logical block of a segment is stored on the j-th flash chip that
j = i % (the number of parallel flash chips)



InDeF for Zoned Namespace SSD

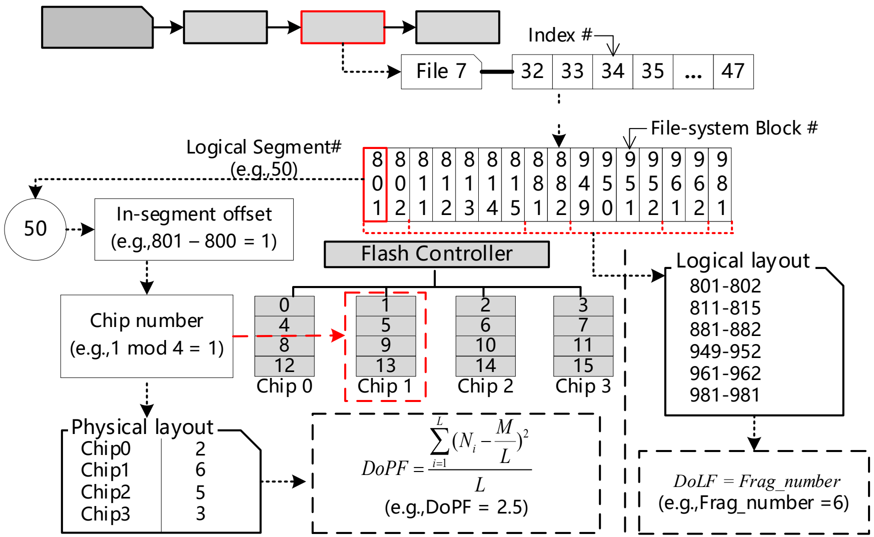
➢The Fragmentation Calculation Module

➢An example of calculating the DoLF and the DoPF

The DoLF can be calculated by the number of logical fragments in the I/O range

The DoPF can be calculated by

$$DoPF = \frac{\sum_{i=1}^{L}(N_i - \frac{M}{L})^2}{L}$$



8	8	8	8	8	8	8	8	8	9	9	9	9	9	9	9
0	0	1	1	1	1	1	8	8	4	5	5	5	6	6	8
1	2	1	2	3	4	5	1	2	9	0	1	2	1	2	1

Chip 0	Chip 1	Chip 2	Chip 3
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Logical layout
801-802
811-815
881-882
949-952
961-962
981-981

Physical layout

Chip0	2
Chip1	6
Chip2	5
Chip3	3

$$DoPF = \frac{\sum_{i=1}^{L}(N_i - \frac{M}{L})^2}{L}$$

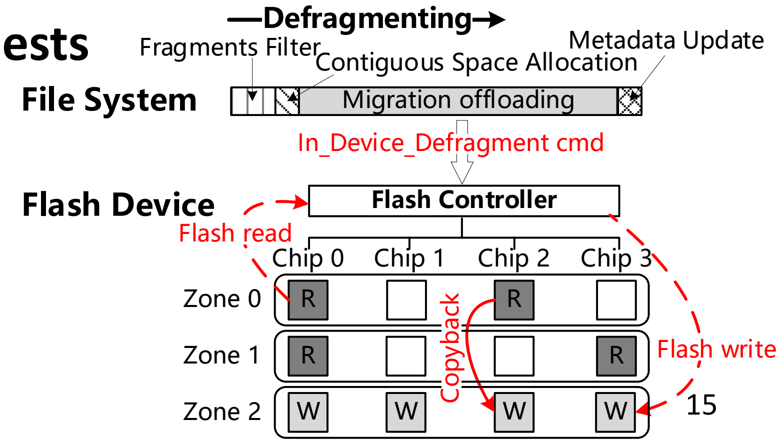
(e.g.,DoPF = 2.5)

$DoLF = Frag\_number$

(e.g.,Frag_number =6)

InDeF for Zoned Namespace SSD

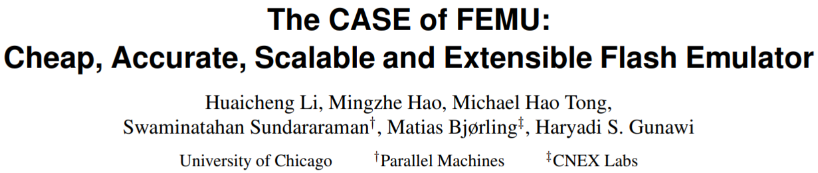
- **The Defragmentation Module**
  - **Truncates I/O information entries with a low IODP value based on the ordered IODP tree**
  - **Groups data based on whether the data page is dirty or not**
  - **Sends the In_Device_Defragment command**
    - Contains a set of source LBAs and a set of destination LBAs
  - **Handles migration offloading and host requests**
    - Flash read and write
    - Copyback



Performance Evaluation

➢Experimental Setup

- ➢InDeF emulator based on FEMU



The CASE of FEMU:
Cheap, Accurate, Scalable and Extensible Flash Emulator

Huaicheng Li, Mingzhe Hao, Michael Hao Tong,
Swaminatahan Sundararaman†, Matias Bjørling‡, Haryadi S. Gunawi

University of Chicago †Parallel Machines ‡CNEX Labs

- ➢A QEMU-based and DRAM-backed NVMe SSD Emulator
- ➢https://github.com/ucare-uchicago/femu

➢Comparison

- ➢defrag.f2fs[3] vs. FragPicker[4] vs. InDeF

➢Workloads

- ➢Synthetic and Macro Benchmarks

➢Objectives

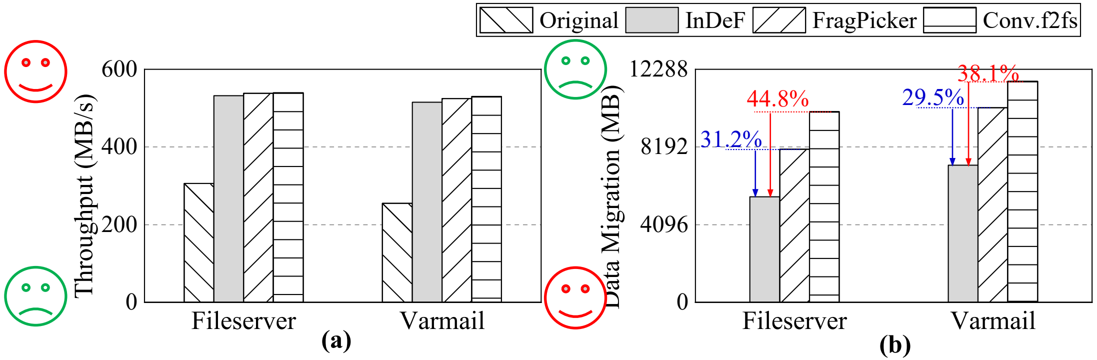
- ➢Does InDeF reduce the amount of writes for defragmentation?
- ➢Does InDeF achieve a similar level of performance gain, compared with conventional tools?
- ➢Does InDeF decrease the elapsed time of defragmentation?

[3] "defrag.f2fs," 2022, https://manpages.debian.org/testing/f2fs-tools/defrag.f2fs.8.en.html.

[4] J. Park and Y. I. Eom, "Fragpicker: A new defragmentation tool for modern storage devices," in Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles, 2021, pp. 280–294.

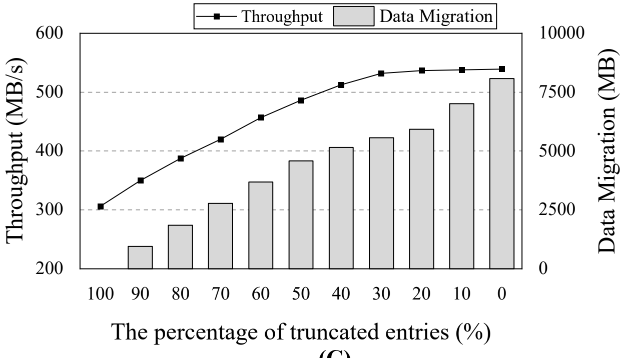
Experiment – Performance and Write Amount

- **Performance**
  - **InDeF improve the throughput by about 72% (fileserver) and 112% (varmail), compared with that before defragmentation**
- **Write amount**
  - **InDeF reduces the amount of writes by around 31.2%-44.8%(Fileserver) and 29.5%-38.1%(Varmail)**



(a)

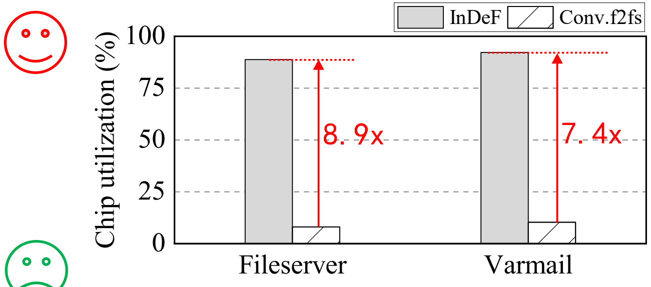
(b)



(C)

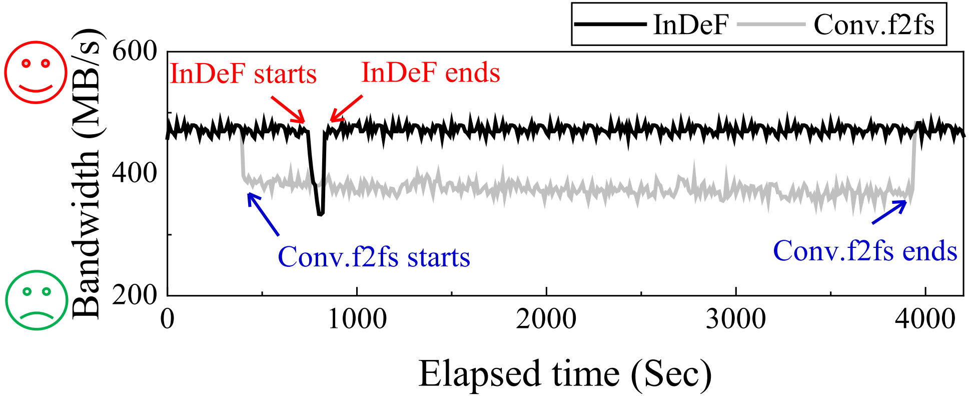
Experiment - Elapsed time of defragmentation

- **Elapsed time of defragmentation**
  - **InDeF increased chip utilization during migration by 8.9x (fileserver) and 7.4x (varmail)**
  - **InDeF decreases the elapsed time of defragmentation time by 94.2% (fileserver) and 91.6% (varmail) due to the higher chip utilization**



Elapsed time (s)	InDeF	Conv.f2fs
Fileserver	21	360
Varmail	35	417

(a)



(b)

More experiments in our paper!

- **Synthetic Benchmarks**
- **Database workloads**
  - **RocksDB YCSB-C**

  …

Conclusion

➢ **Main goals**

- ✓ Minimizes the amount of migration data for defragmentation to reduce the write traffic of the underlying device
- ✓ Decreases elapsed time of defragmentation to reduce the impact on co-running application performance

➢ **InDeF for ZNS SSD**

- ✓ Combines the degree of fragmentation and access hotness to find out the most suitable data set to migrate in each file
- ✓ By the In_Device_Defragment command, InDeF offloads the data migration from the host to the ZNS SSD to improve the efficiency of defragmentation
- ✓ Decreases the elapsed time of defragmentation significantly while minimizing the amount of data migration for defragmentation

Thanks!