

# FragPicker: A New Defragmentation Tool for Modern Storage Devices

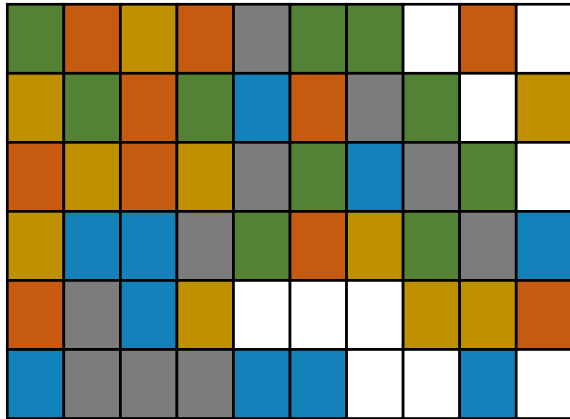
**Jonggyu Park** and Young Ik Eom  
Sungkyunkwan University



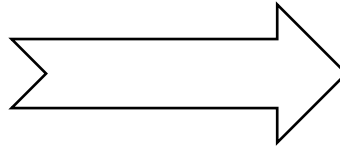
# What is fragmentation?

a.k.a file fragmentation, filesystem fragmentation, and filesystem aging

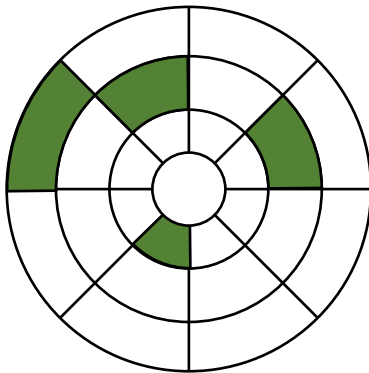
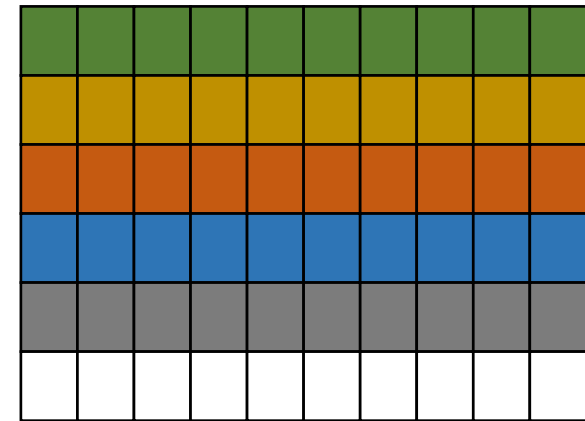
Filesystem view



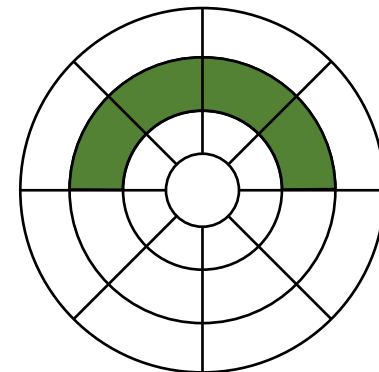
Defragment!



Filesystem view

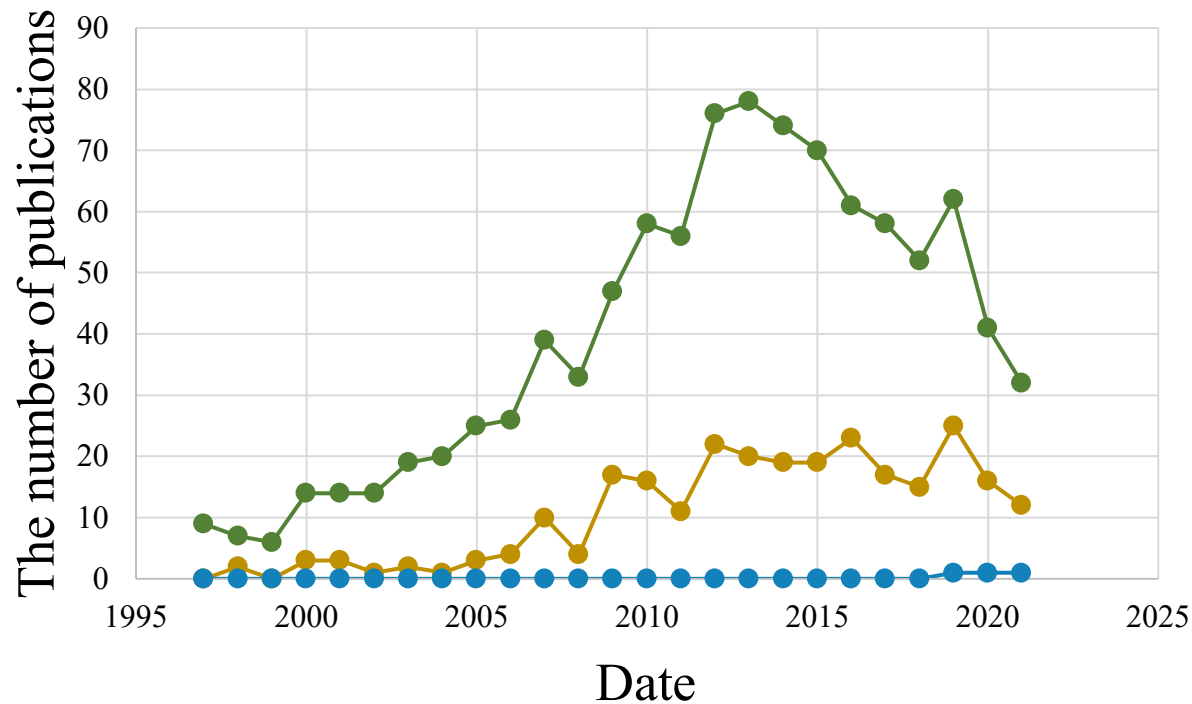


Storage view



Storage view

# Fragmentation, is the case closed?



Google Scholar

"file fragmentation"



☒ Articles ☐ Case law

"file fragmentation" "flash"



☒ Articles ☐ Case law

"file fragmentation" "optane"



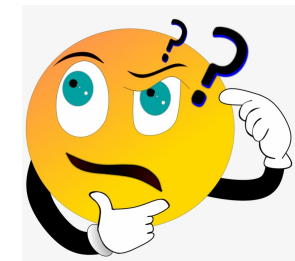
☒ Articles ☐ Case law

# Fragmentation, is the case closed?

Let's ask about it to the SSD vendors

For sure...?

They said sth different...

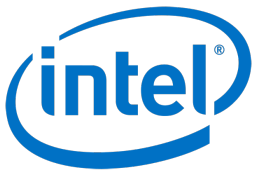


## Do SSDs require defragmentation?



**No. Defragmentation is not required.**

Because SSDs have no moving parts, they can access any data location equally fast. You should disable automatic defragmentation on your computer. Frequently defragmenting your SSD will reduce its lifespan. Please visit the OS Optimization section of Samsung Magician for help disabling automatic defragmentation.



## ✓ Should I defrag my Intel® Solid State Drive with Windows\* Disk Defragmenter or similar program?

No. Traditional hard disk drive **defragmentation tools do not show increased SSD device performance.**

- For legacy operating systems, disable any automatic or scheduled defragmentation utilities for your Intel® SSD. The tools add unnecessary wear.
- Newer Windows\* operating systems can detect the presence of an SSD and disable the defrag function.

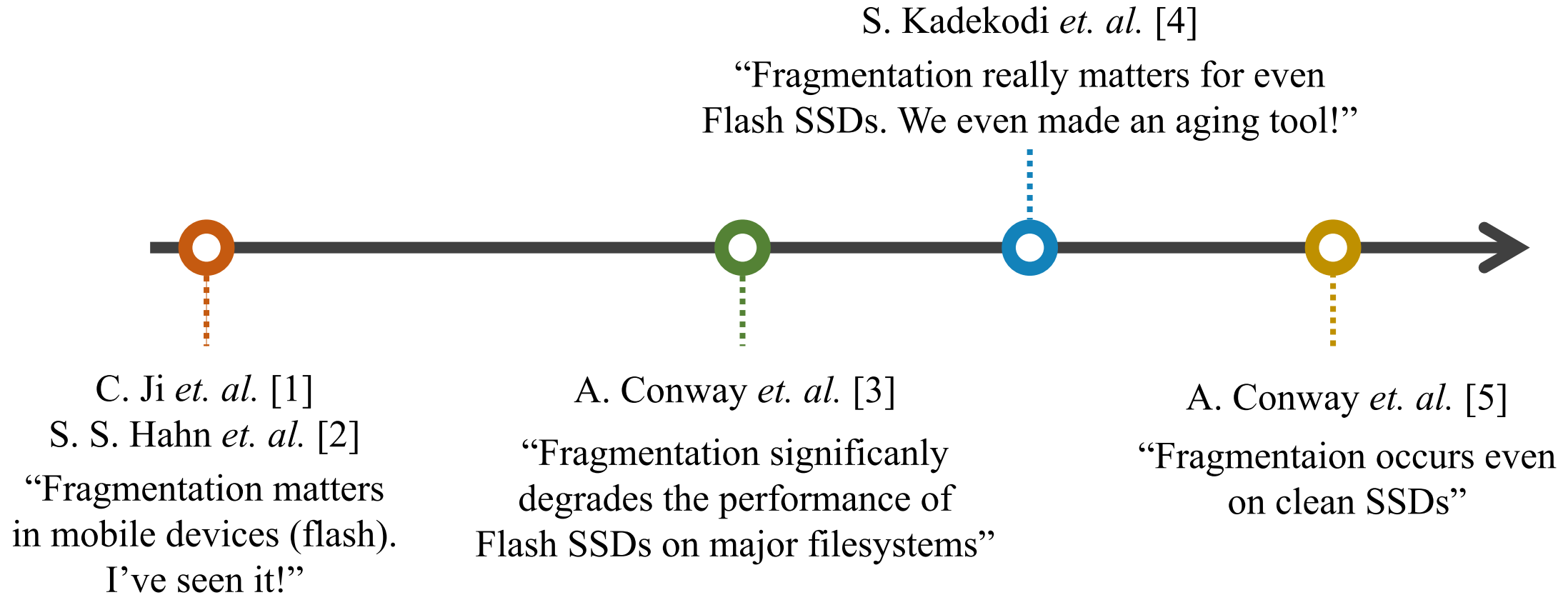


Fourth, **SSDs do not need to be defragmented.**

Instead of finding data mechanically as HDDs do, the data stored on the memory chip of an SSD is read using electric signals, which means there is no need for defragmentation.



# Fragmentation: the case is not closed yet!



[1] An empirical study of file-system fragmentation in mobile storage systems, HotStorage 2016

[2] Improving file system performance of mobile storage systems using a decoupled defragmenter, ATC 2017

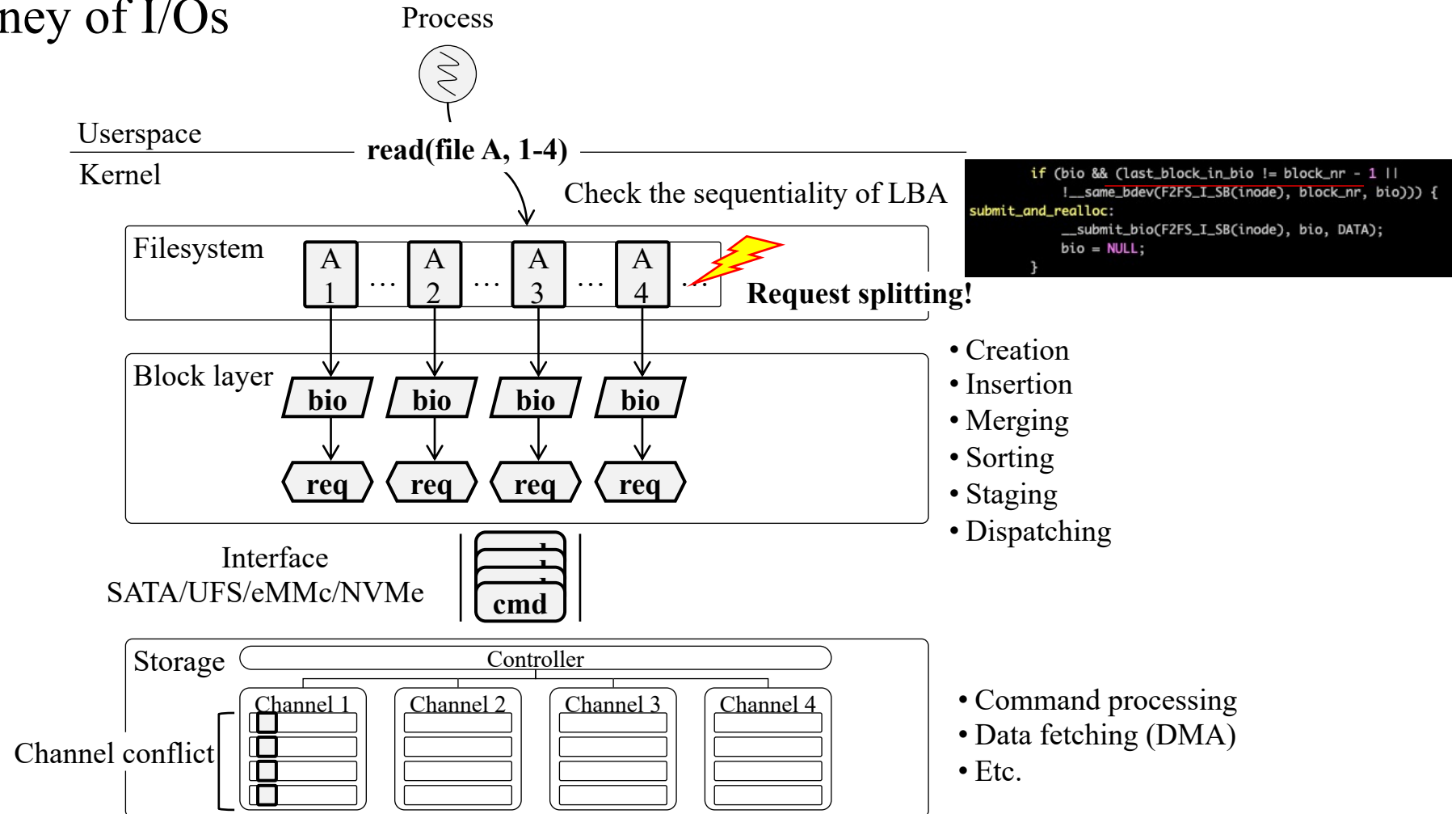
[3] File systems fated for senescence? nonsense, says science!, FAST 2017

[4] Geriatrics: Aging what you see and what you don’t see. A file system aging approach for modern storage systems, ATC 2018

[5] Filesystem aging: It’s more usage than fullness, HotStorage 2019

# But... the vendors said no moving parts....

Let's follow the journey of I/Os

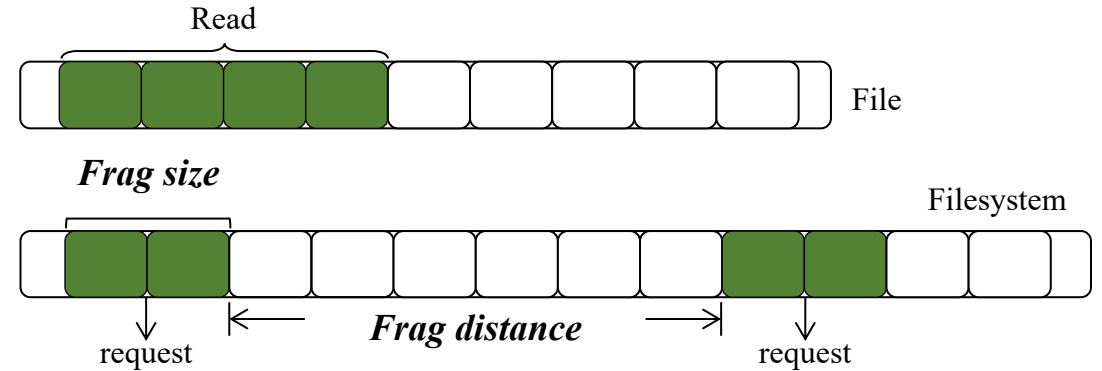


# Let's verify this “scientifically”

New terminology and metrics

**Frag size:** *the size of each fragment that are contiguous in terms of LBA*

**Frag distance:** *the distance between two consecutive fragments*



**CC (Correlation Coefficient):**

how related the value is to the performance

**NLRS (Normalized Linear Regression Slope):**

how significantly the value influences the performance.

$$(1) \quad CC(X, Y) = \frac{\sum (x - \bar{x})(y - \bar{y})}{\sqrt{\sum (x - \bar{x})^2 \sum (y - \bar{y})^2}}$$

$$(2) \quad NLRS(X, Y) = \frac{\sum (x - \bar{x})(y - \bar{y})}{\sum (x - \bar{x})^2}$$

# Let's verify this “scientifically”

## Evaluation Setup.

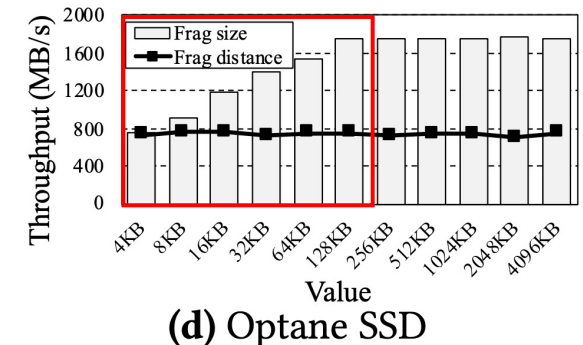
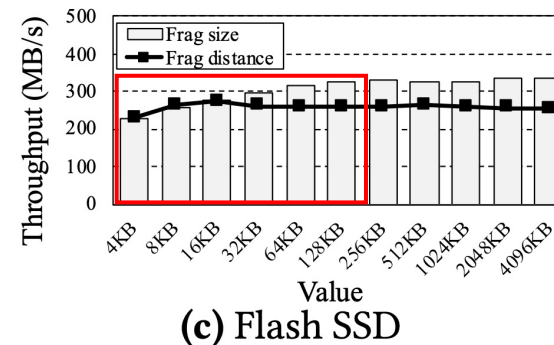
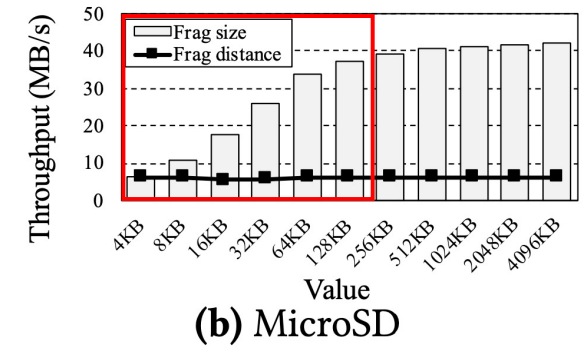
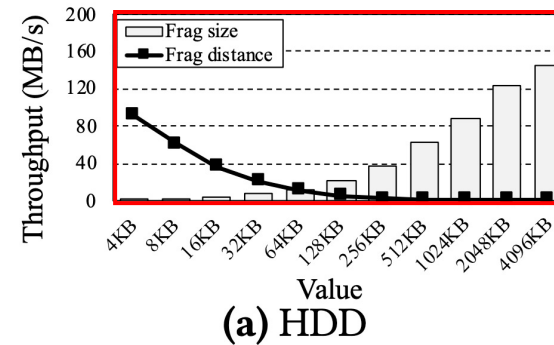
- 128KB O\_DIRECT sequential read on F2FS
- Varying frag size/frag distance

\* **Request splitting** occurs when frag size < 128KB

## Observations

1. The performance of HDD is **sensitive to both frag size and distance** (due to seek time)
2. Flash/Optane storage is **sensitive to frag size** when it's less than 128KB.
3. Flash/Optane storage is **not sensitive to frag size ( $\geq 128\text{KB}$ ) and frag distance.**

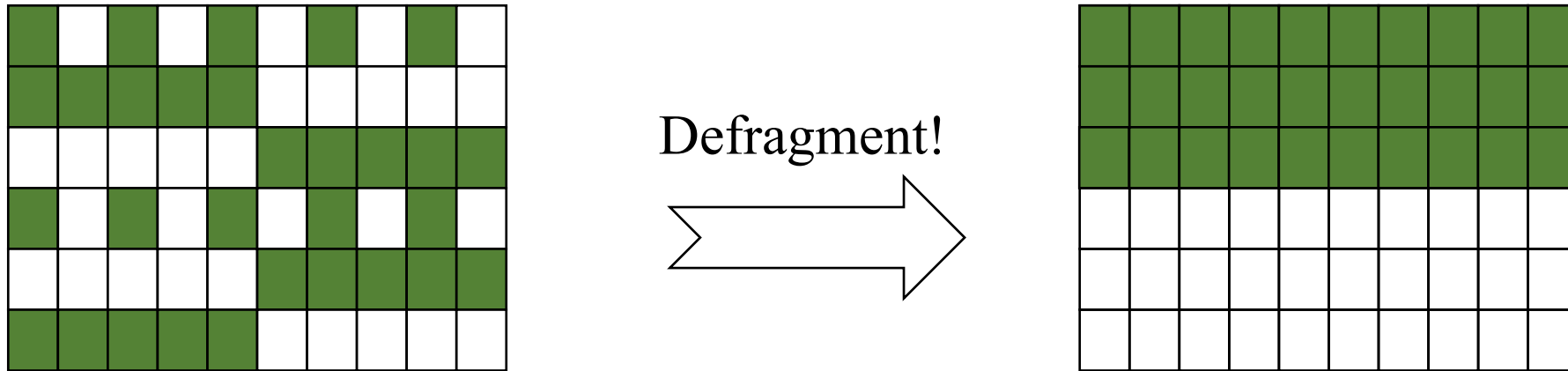
The performance of Flash/Optane storage, mostly depends on **the occurrence of request splitting** and is irrelevant to the distance between fragments.



Metric	Frag Size				Frag Distance	
	Correlation Coefficient (CC)		Normalized Linear Regression Slope (NLRs)		Correlation Coefficient (CC)	Normalized Linear Regression Slope (NLRs)
	Before 128KB	After 128KB	Before 128KB	After 128KB		
HDD	0.9898939	0.9219008	0.1027717	0.0204404	-0.5333534	-0.0345131
MicroSD	0.8889961	0.7488593	0.0380474	0.0001394	0.2562098	0.0000272
Flash SSD	0.8437115	0.7475056	0.0027932	0.0000129	0.3109179	0.0000581
Optane SSD	0.8883913	0.4166667	0.0096721	0.0000022	-0.1539585	-0.0000091

# Let's revisit defragmentation tools

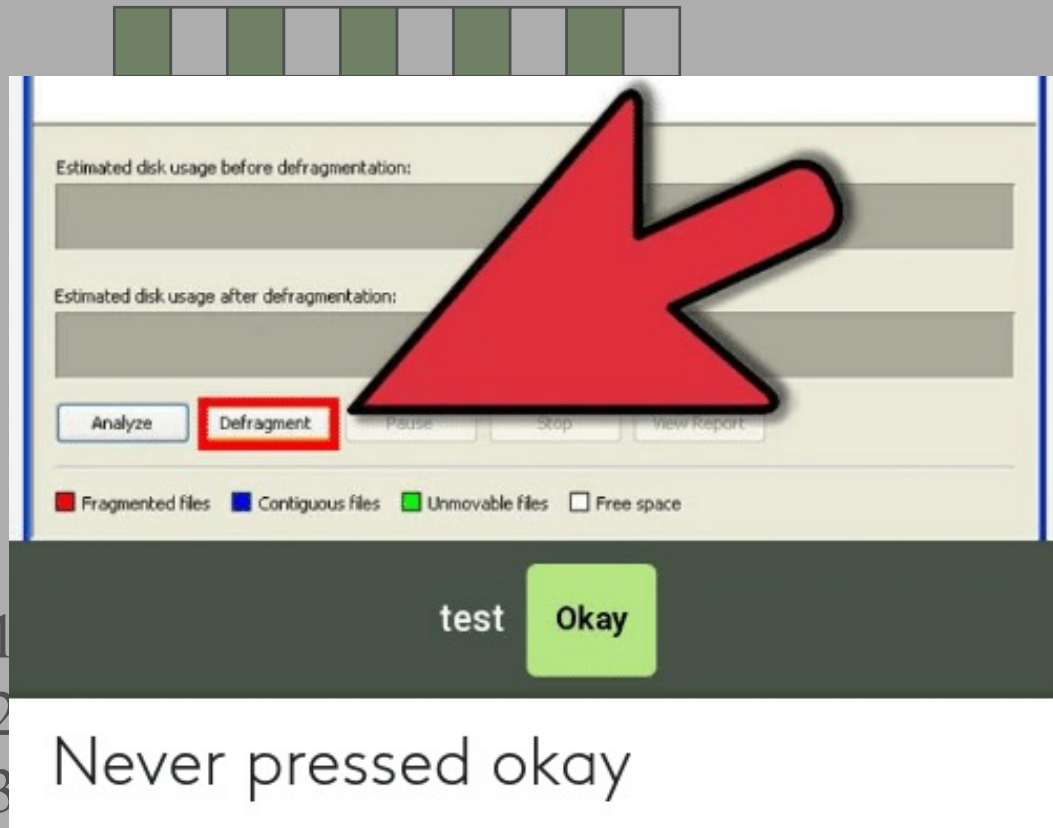
Conventional defragmentation tools migrates the entire contents of files



1. Filesystem-dependent (in-place update: ext4 and xfs, out-place update: f2fs and btrfs)
2. Significantly degrades the performance of co-running applications
3. The additional writes reduce the lifespan of modern storage devices.
4. Time-consuming

# Let's revisit defragmentation tools

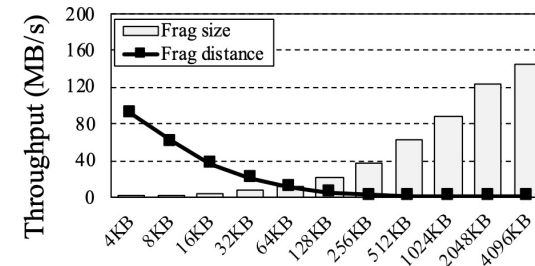
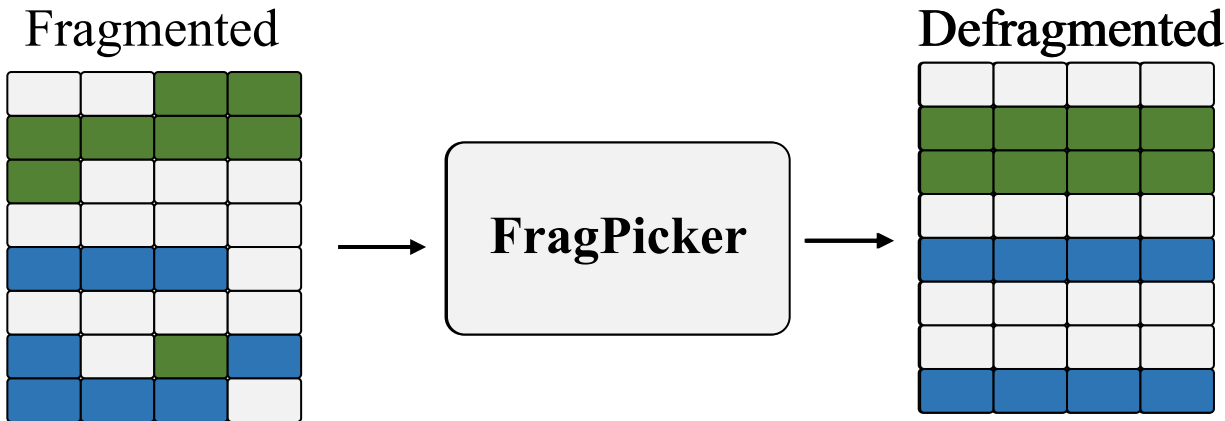
Conventional defragmentation tools migrates the entire contents of files



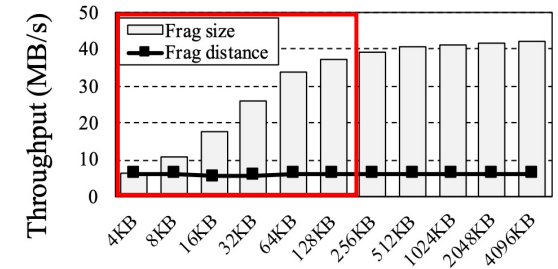
4. The additional writes will reduce the lifespan of modern storage devices.

# FragPicker for modern storage devices

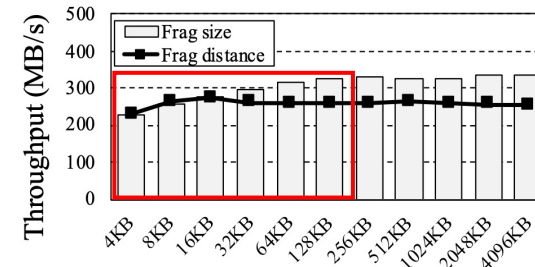
Let's ignore ~~frag distance~~ and do our best to prevent request splitting!



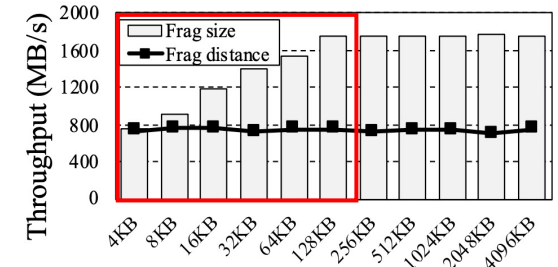
(a) HDD



(b) MicroSD



(c) Flash SSD



(d) Optane SSD

# FragPicker for modern storage devices

## Key Challenges

1. Which data to migrate?

2. How to migrate?



# FragPicker for modern storage devices

## Key Challenges

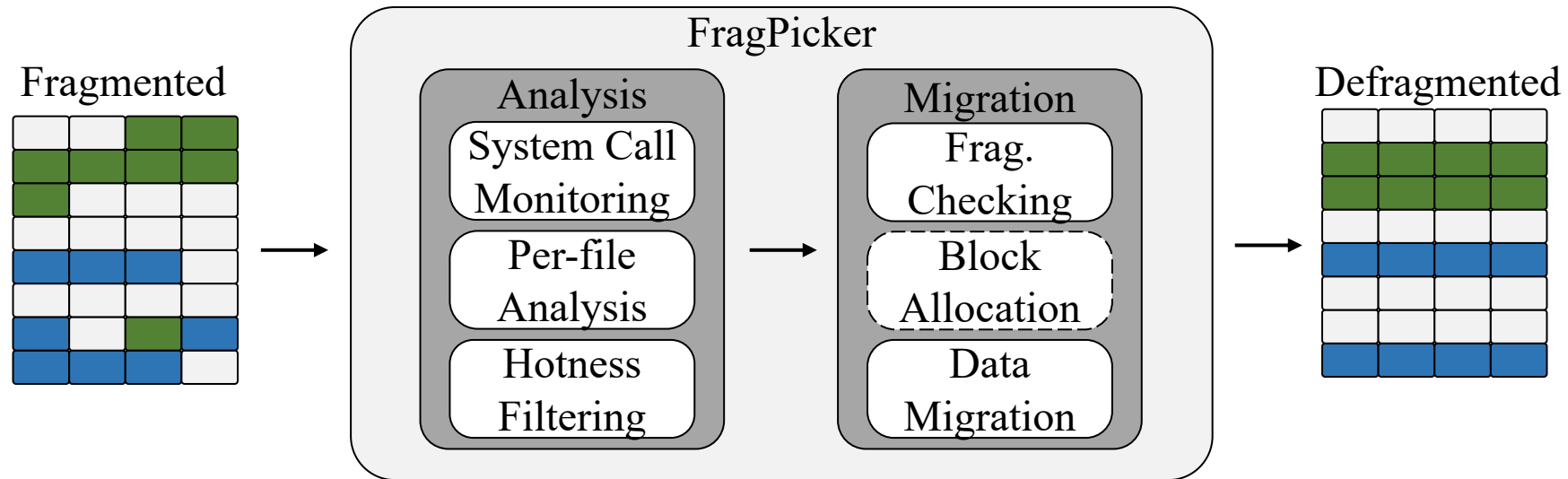
1. Which data to migrate?

➔ I/O Analysis to find the best pieces of data blocks to prevent request splitting

2. How to migrate?

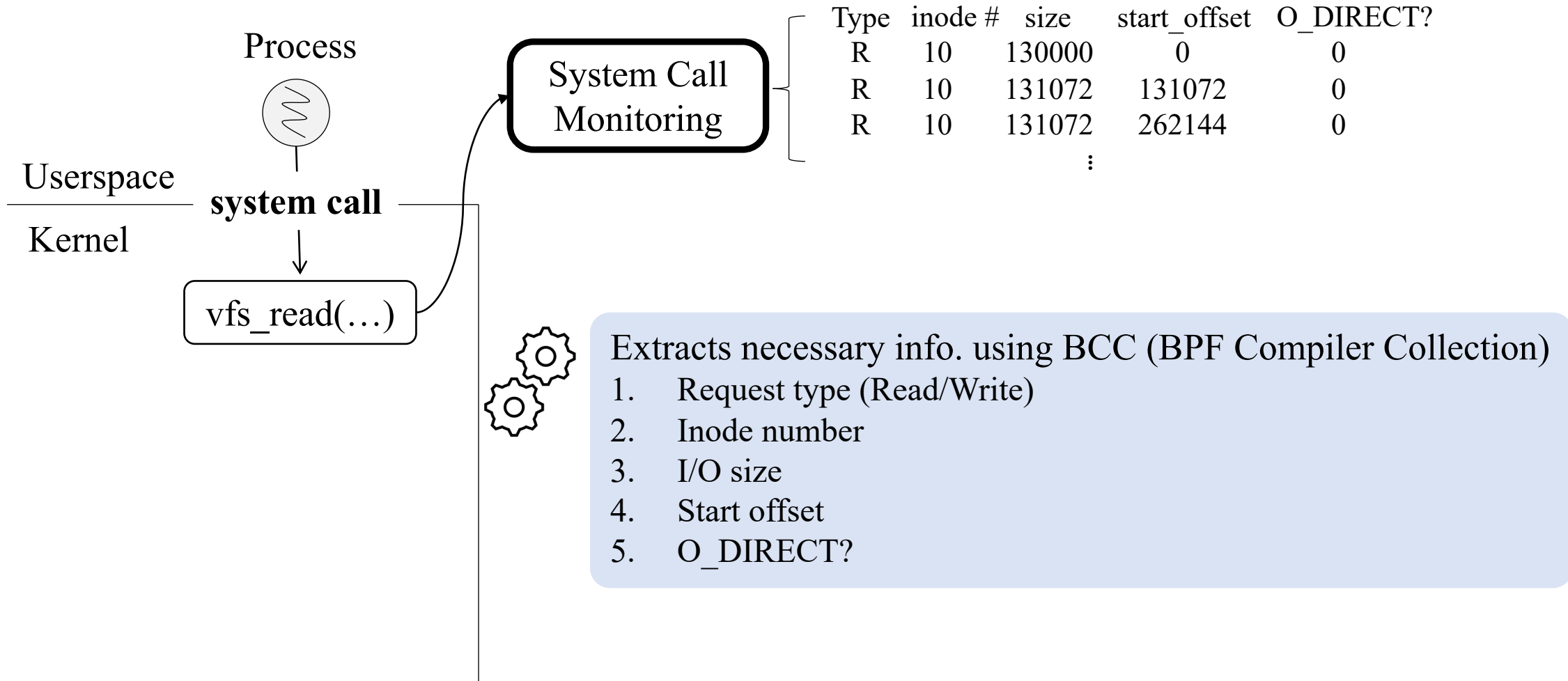
➔ Filesystem-agnostic migration while minimizing the amount of writes

# FragPicker for modern storage devices

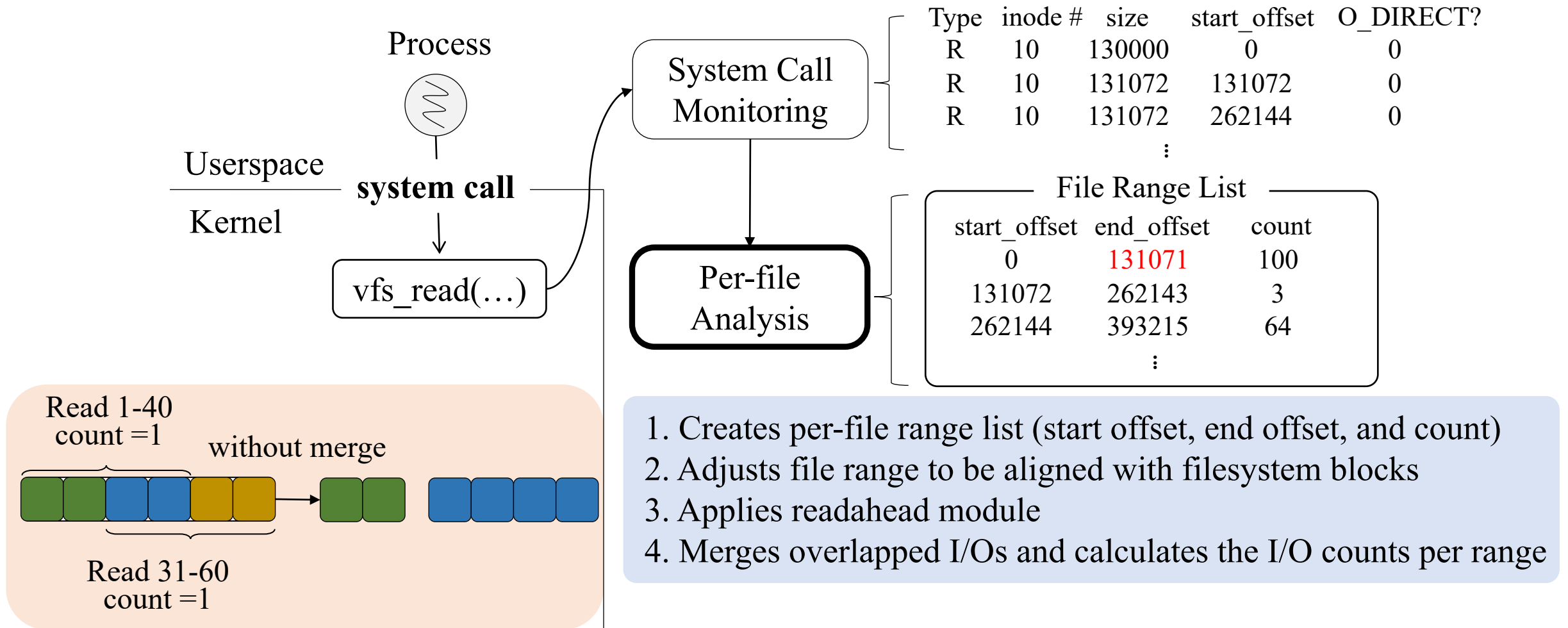


- ❖ Analysis Phase
  - Monitors system calls related I/Os
  - Analyzes I/O characteristics per file
  - Filters I/Os with hotness
- ❖ Migration Phase
  - Checks the fragmentation state
  - Allocates contiguous blocks
  - Migrates fragmented data into a new space

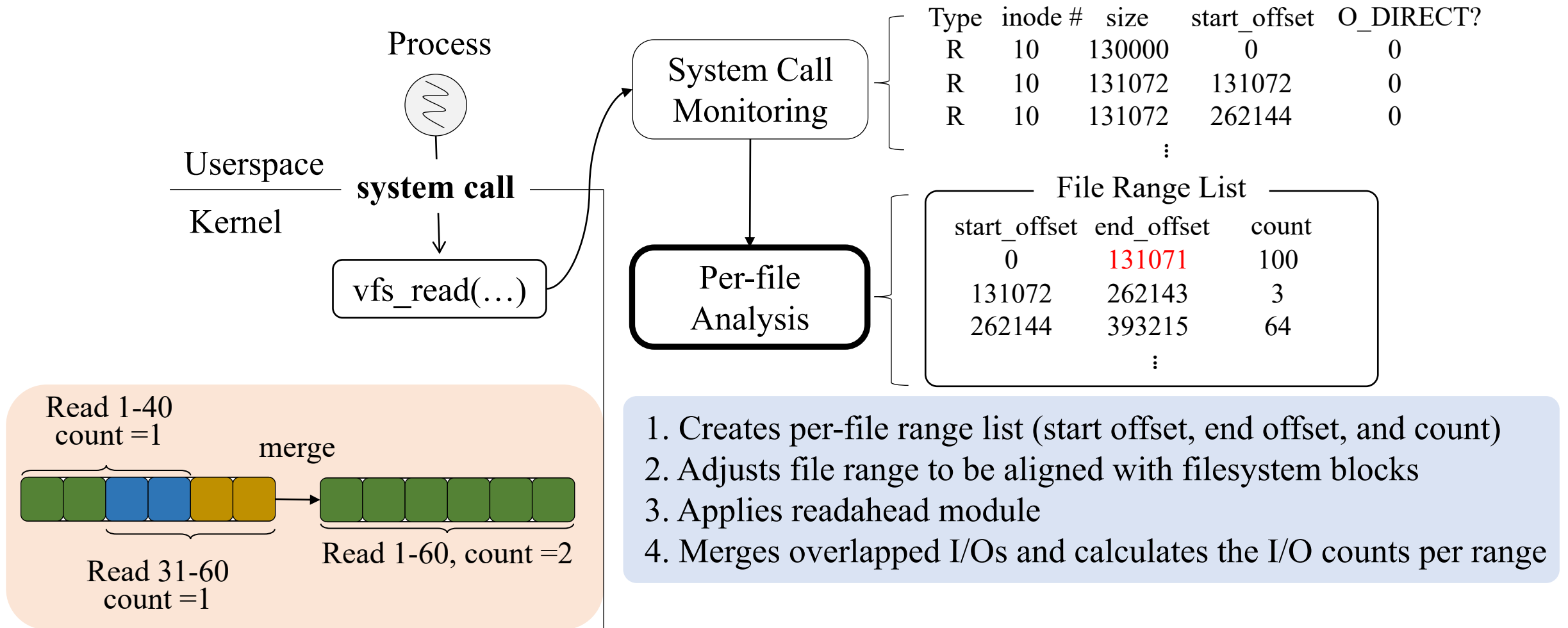
# The analysis phase of FragPicker



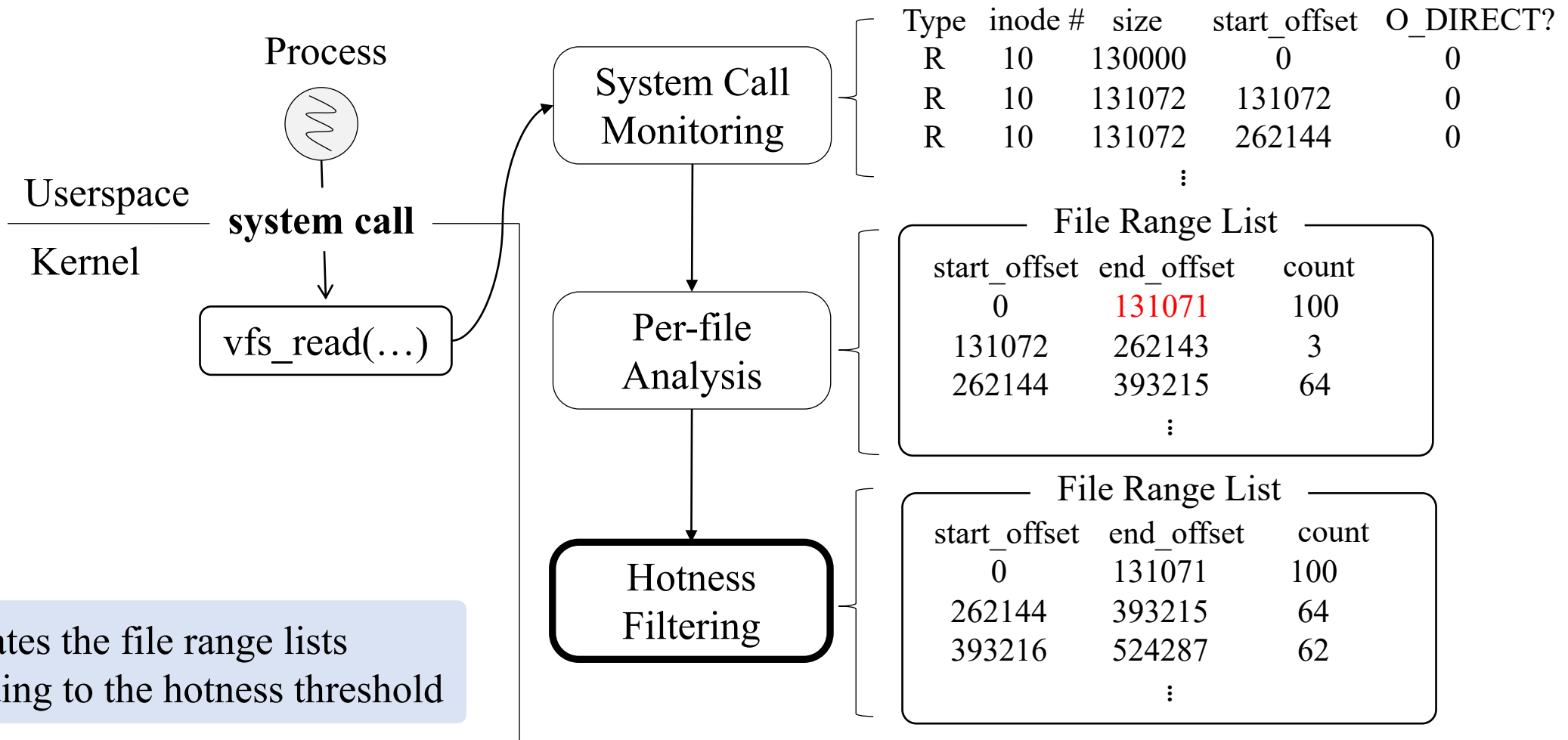
# The analysis phase of FragPicker



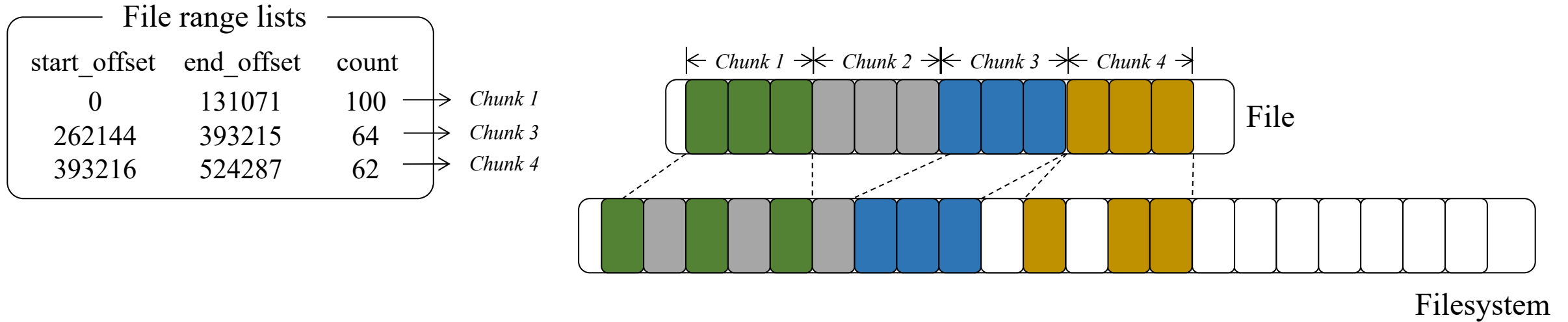
# The analysis phase of FragPicker



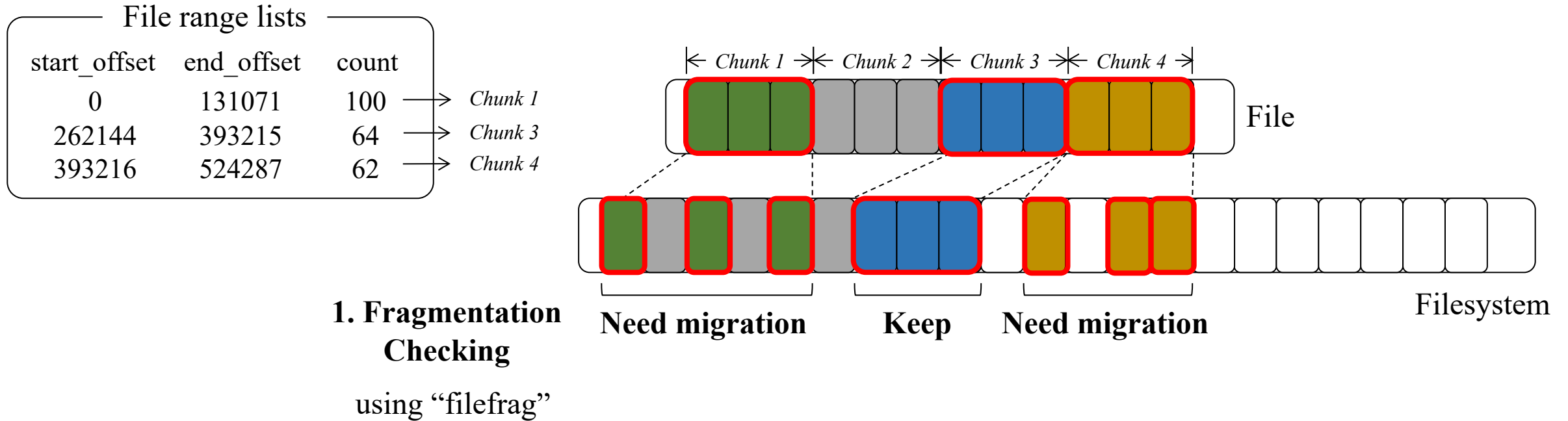
# The analysis phase of FragPicker



# The migration phase of FragPicker



# The migration phase of FragPicker

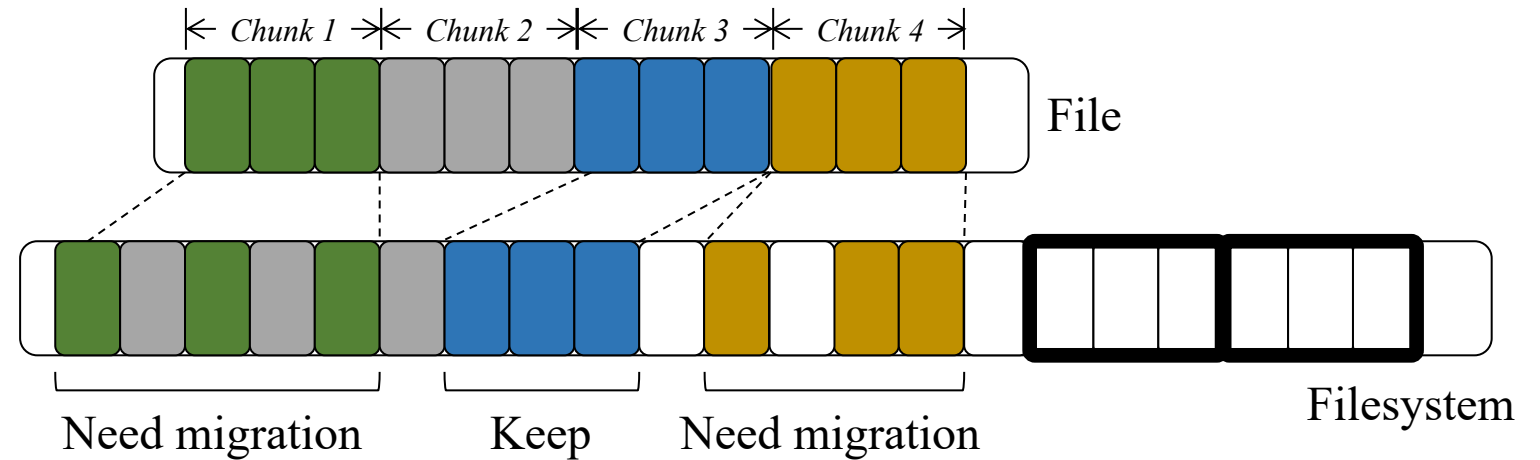




# The migration phase of FragPicker

File range lists		
start_offset	end_offset	count
0	131071	100
262144	393215	64
393216	524287	62

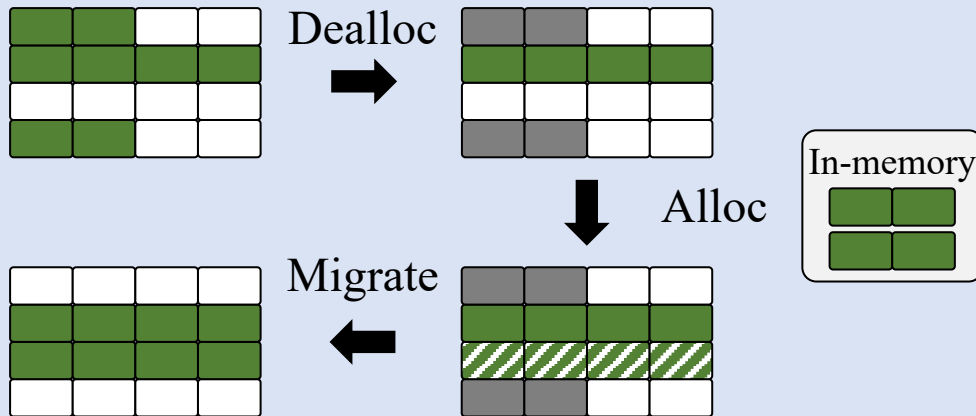
1. Fragmentation  
Checking



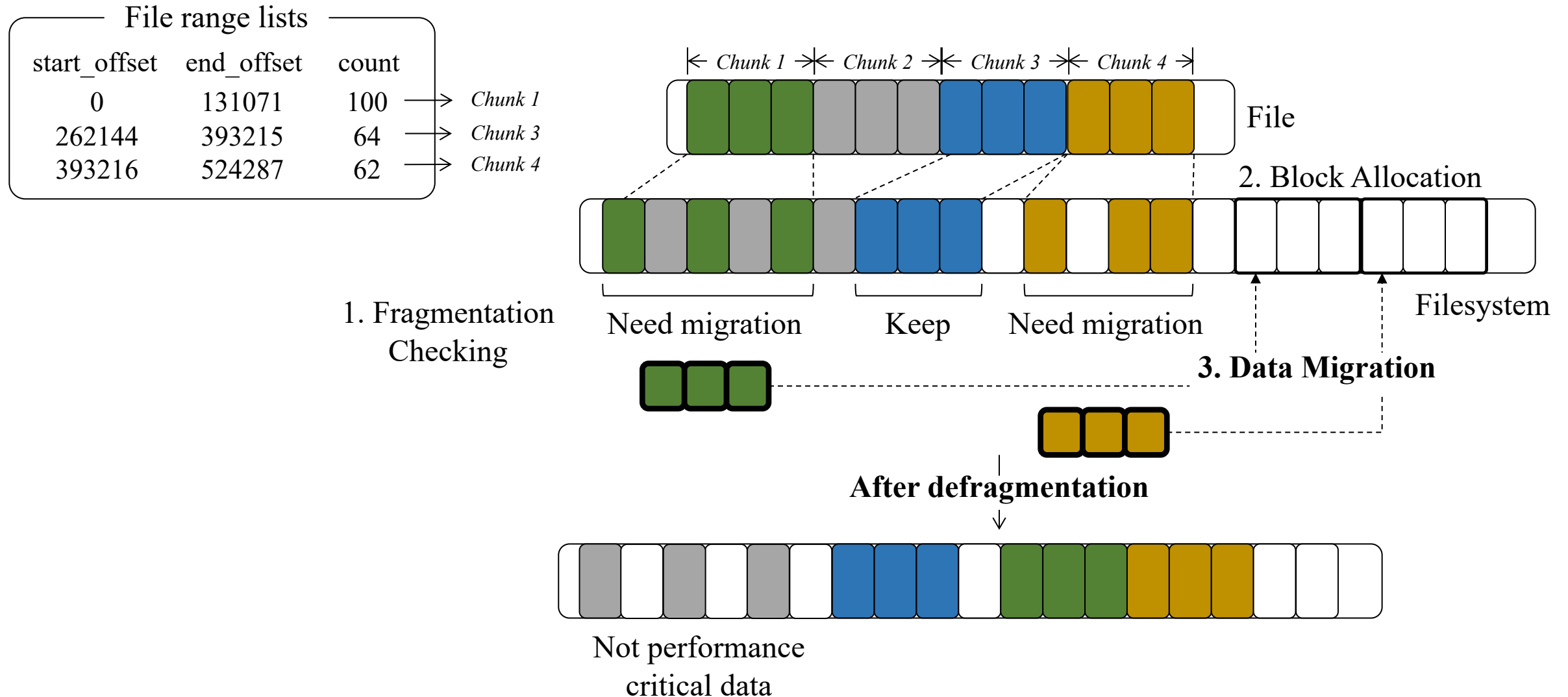
2. Block Allocation

Using fallocate()

In the case of out-place update filesystems,  
FragPicker can skip this process



# The migration phase of FragPicker



# Evaluation

## Storage

1. Samsung SATA Flash SSD 850 PRO 256GB
2. Intel NVMe Optane SSD 905P 960GB



## Workloads

1. Sequential read/update
2. Stride read/update

## Objectives

1. Does FragPicker reduce the **amount of writes** for defragmentation?
2. Does FragPicker achieve a similar level of **performance gain**, compared with conventional tools?

# Evaluation (Flash SSD)

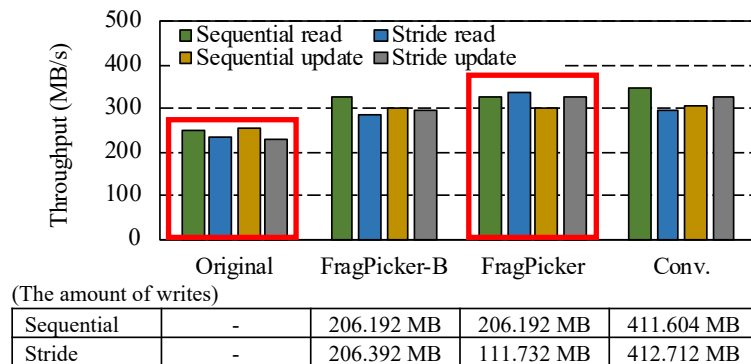
## 1) Write amount

FragPicker reduces the amount of writes by around 50% (sequential) and 75% (stride)

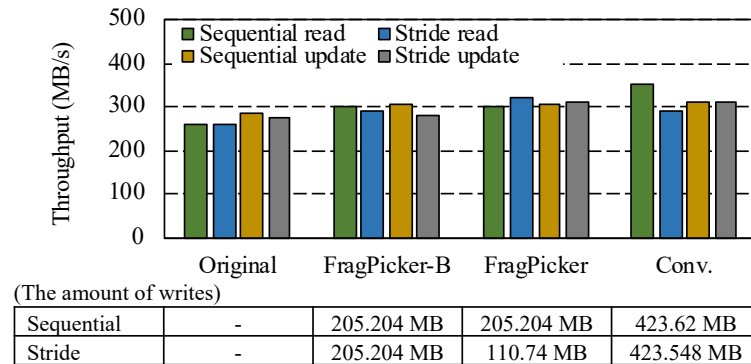
## 2) Performance

FragPicker improves sequential/stride I/O by around 30% and 42%, respectively

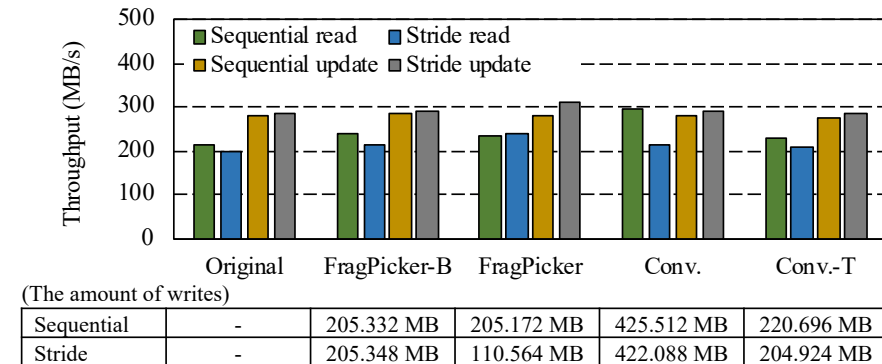
Regarding stride read performance, FragPicker outperforms conventional ones by 15%



Ext4



F2FS



Btrfs

# Evaluation (Optane SSD)

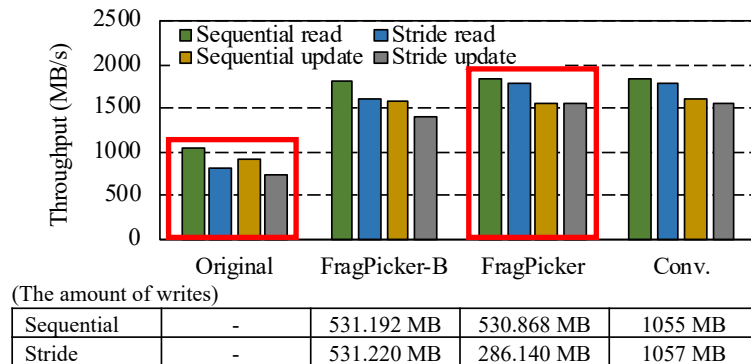
## 1) Write amount

FragPicker reduces the amount of writes by around 50% (sequential) and 75% (stride)

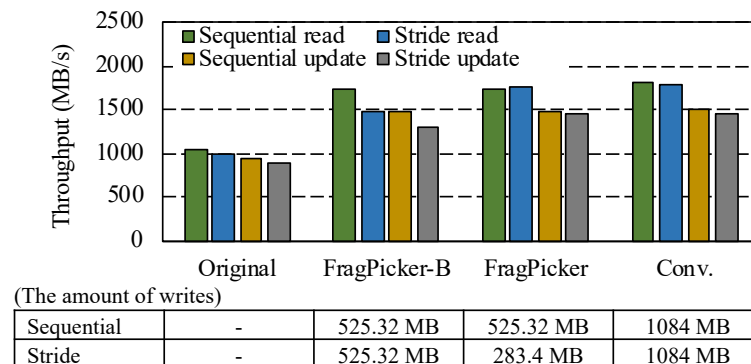
## 2) Performance

FragPicker improves sequential/stride I/O by around 75% and 120%, respectively

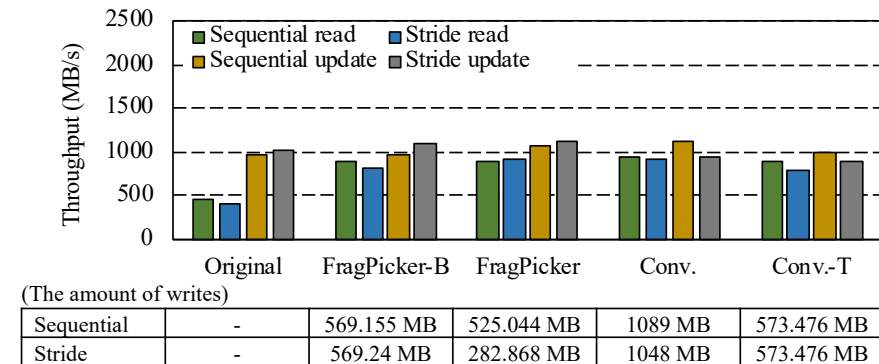
\* High performance devices suffer from kernel overheads



Ext4



F2FS



Btrfs

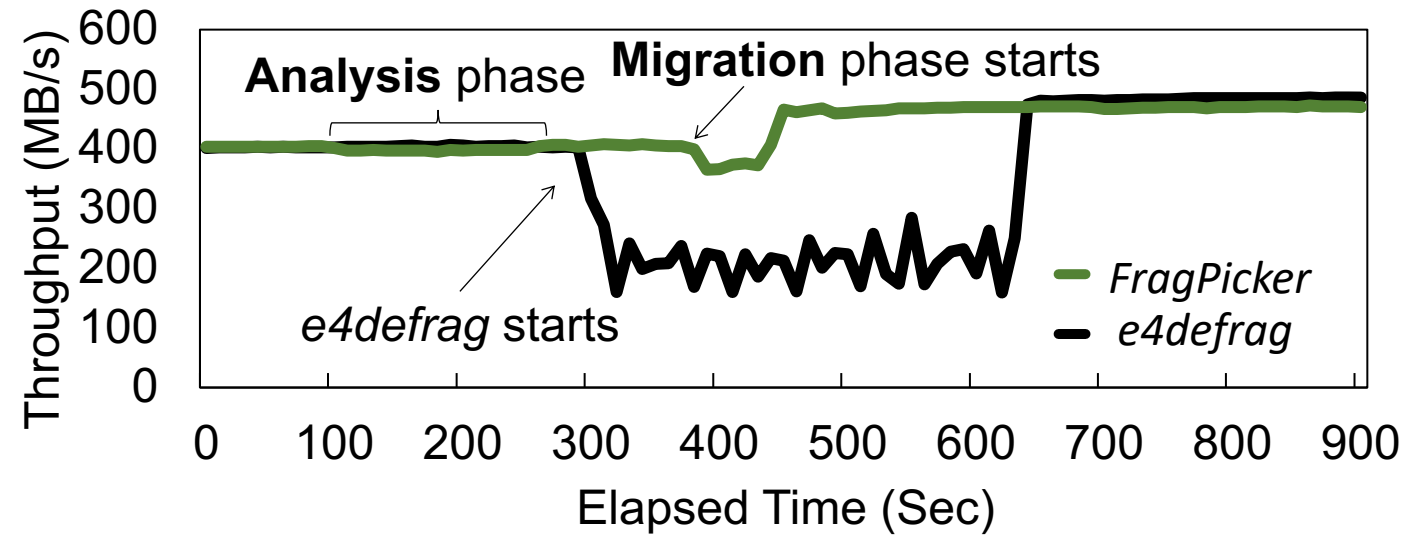
# Evaluation (Optane SSD)

## Evaluation Setup

- Pre-aged ext4 on Optane 905P SSD
- RocksDB ycsb-C (100% read, zipfian)

## Evaluation Results

	e4defrag	FragPicker
Elapsed Time (sec.)	331	54
Perf. Interference	47%	7.4% (1.4%)
Perf. After (MB/s)	483.5	467.3
Write Amount	23GB	7.2GB



# More experiments in our paper!

- Database workloads
  - SQLite
- Fileserver workloads
- Hotness filtering test
- ...

# Discussion

- SSDs are diverse
  - Each SSD has different FTL implementations especially depending on the vendors
  - Prefetching policy?
  - Mapping policy?
- What's the best data placement to exploit maximum performance?



# Conclusion

- FragPicker
  - A new defragmentation tool for modern storage devices
  - No kernel modification
  - Filesystem agnostic
- Requirements for FragPicker
  - eBPF support
  - Filesystem-level consistency
  - fallocate and filefrag
  - Filesystem support for providing a contiguous LBA region for a single write op.

# Thank you

Source code: <https://github.com/jonggyup/FragPicker>

Contact: [jonggyu@skku.edu](mailto:jonggyu@skku.edu)

