ZoneAlloy: Elastic Data and Space Management for Hybrid SMR Drives

Fenggang Wu, Bingzhe Li, Zhichao Cao, Baoquan Zhang

Ming-Hong Yang, Hao Wen, David H.C. Du

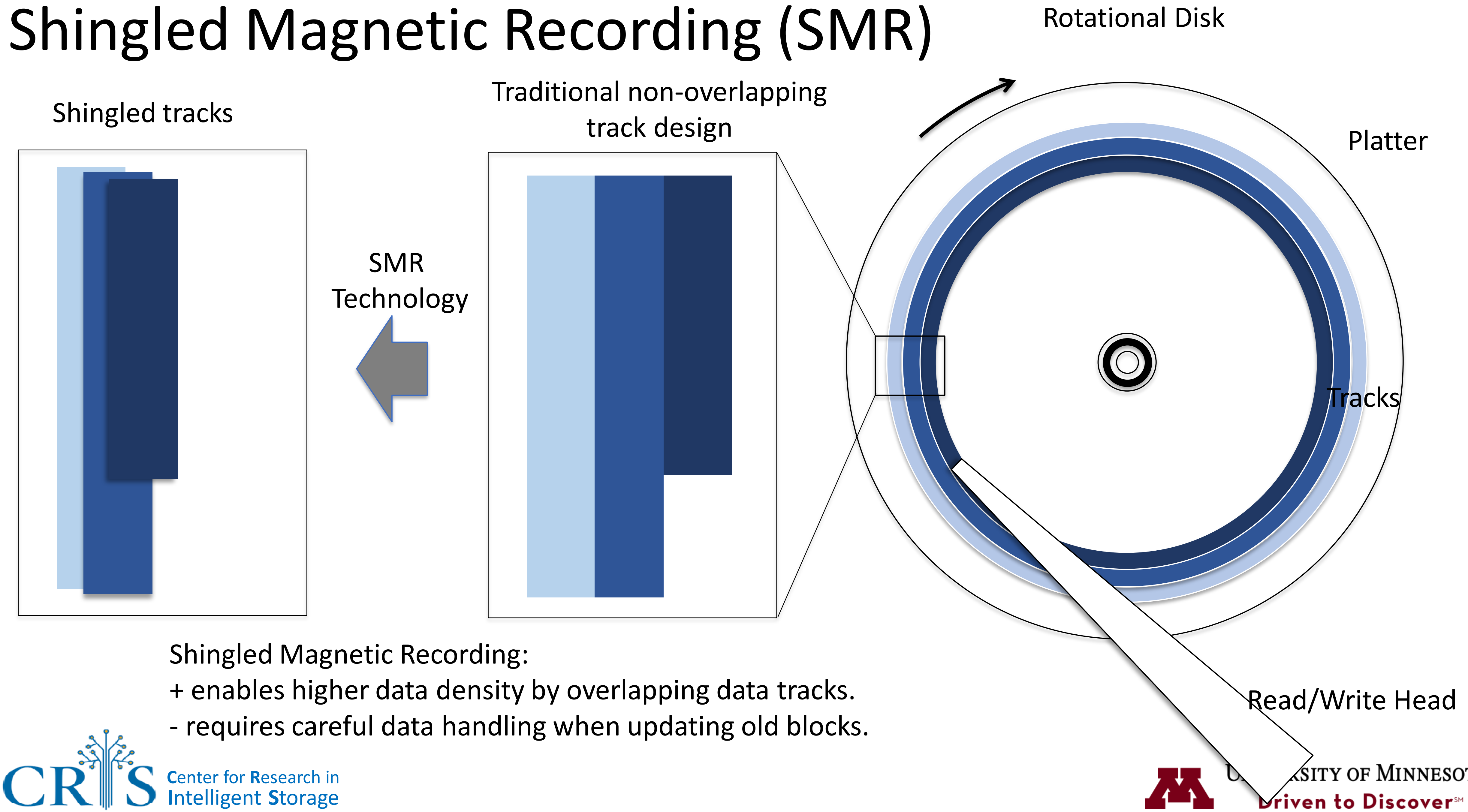
University of Minnesota, Twin Cities

Jul. 8, 2019. HotStorage'19

Center for Research in Intelligent Storage

University of Minnesota
Driven to Discover

Shingled Magnetic Recording (SMR)



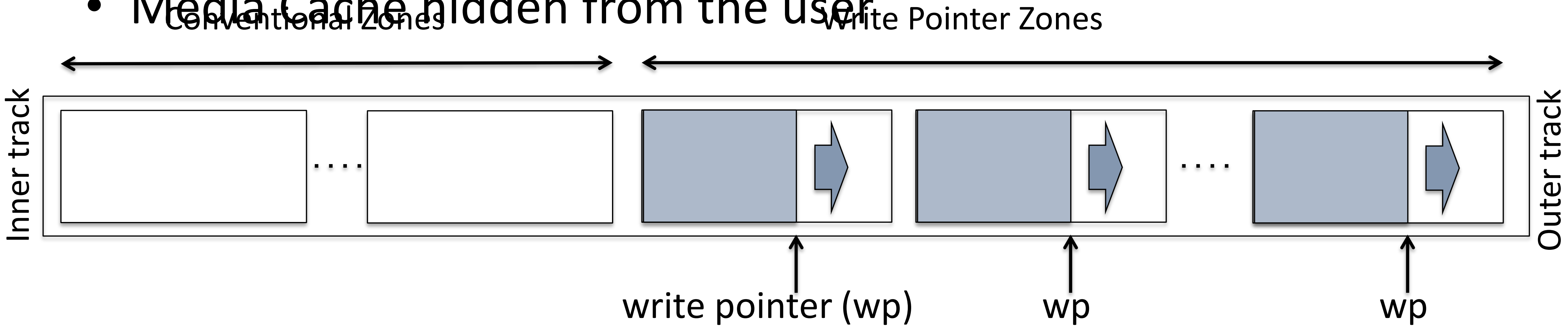
Shingled Magnetic Recording:
+ enables higher data density by overlapping data tracks.
- requires careful data handling when updating old blocks.

T10 SMR Drive Models

- Drive Managed
  - Black box/drop-in solution: the drive handles all out-of-order write operations.
- Host Managed
  - White box/application modification needed: the drive reports zone layout information; out-of-order writes will be rejected.
- Host Aware
  - Grey box: the drive reports zone layout information; out-of-order writes will still be handled internally.
  - Applications can use HA-SMR drive as is, and also have the opportunity for zone-layout aware optimizations.

Example: Seagate HA SMR Sample Drives

- Model: ST8000AS0022-1WL, prototype firmware revision ZN03.
- Small Conventional Zone 64GB/8TB ~= 1%
- Most disk space is sequential write preferred zone
- Media Cache hidden from the user

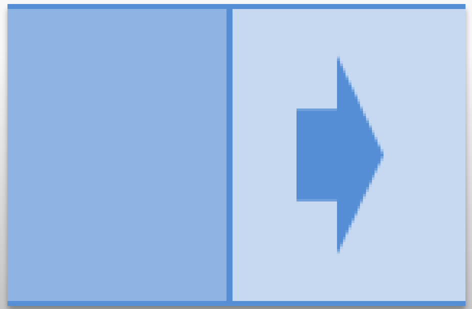


Center for Research in Intelligent Storage

University of Minnesota
Driven to Discover℠

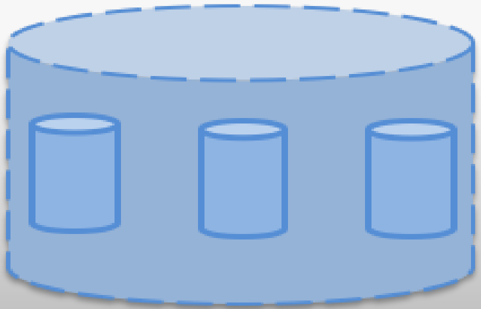
Motivation & Goals

- Motivation: To meet the challenge of using SMR drives in large-scale storage systems.



SMR Layout Awareness

- Exploring which level to be SMR zone layout aware to support different applications (FS, DB, etc.)

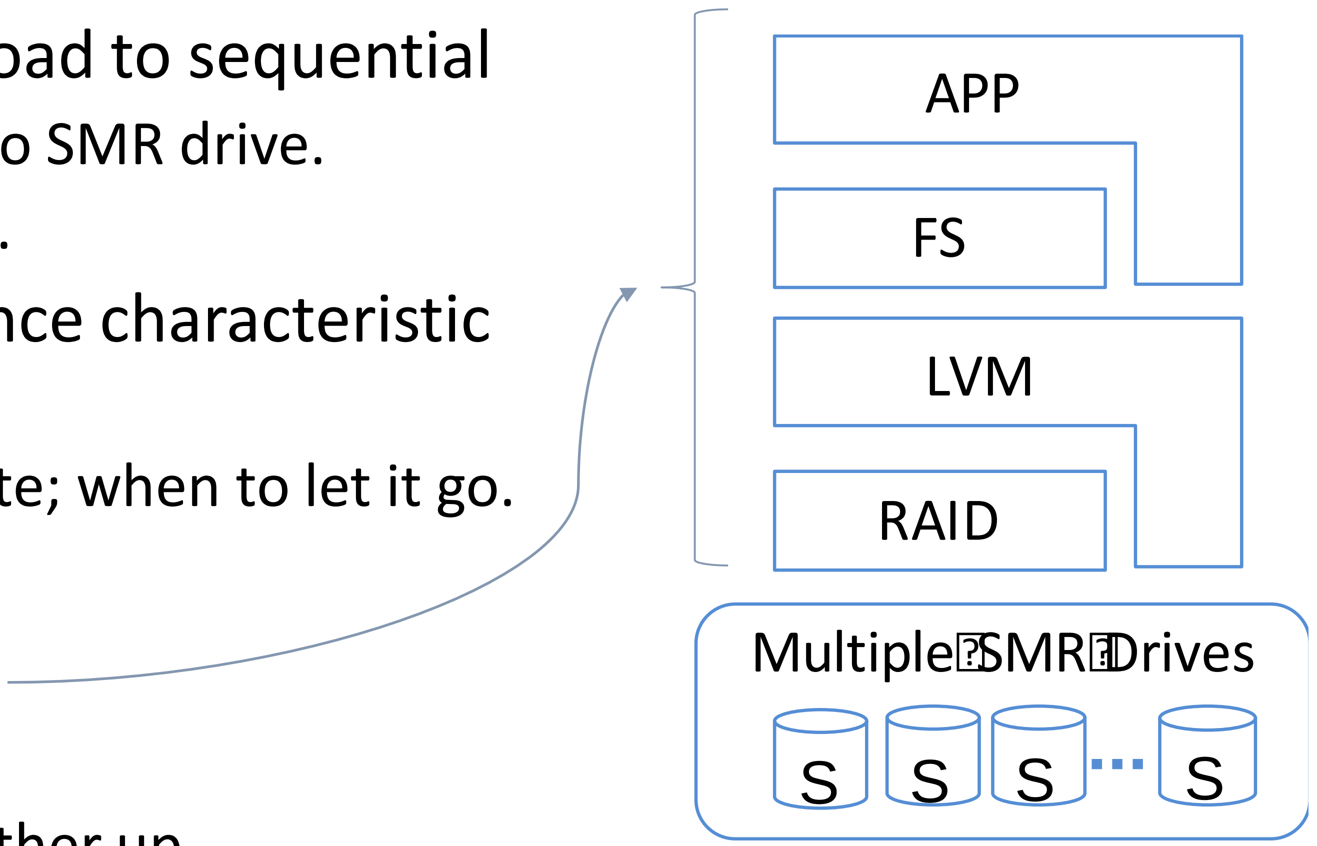


SMR Drive Aggregation

- How to reduce the design complexity for multiple SMR drive applications.

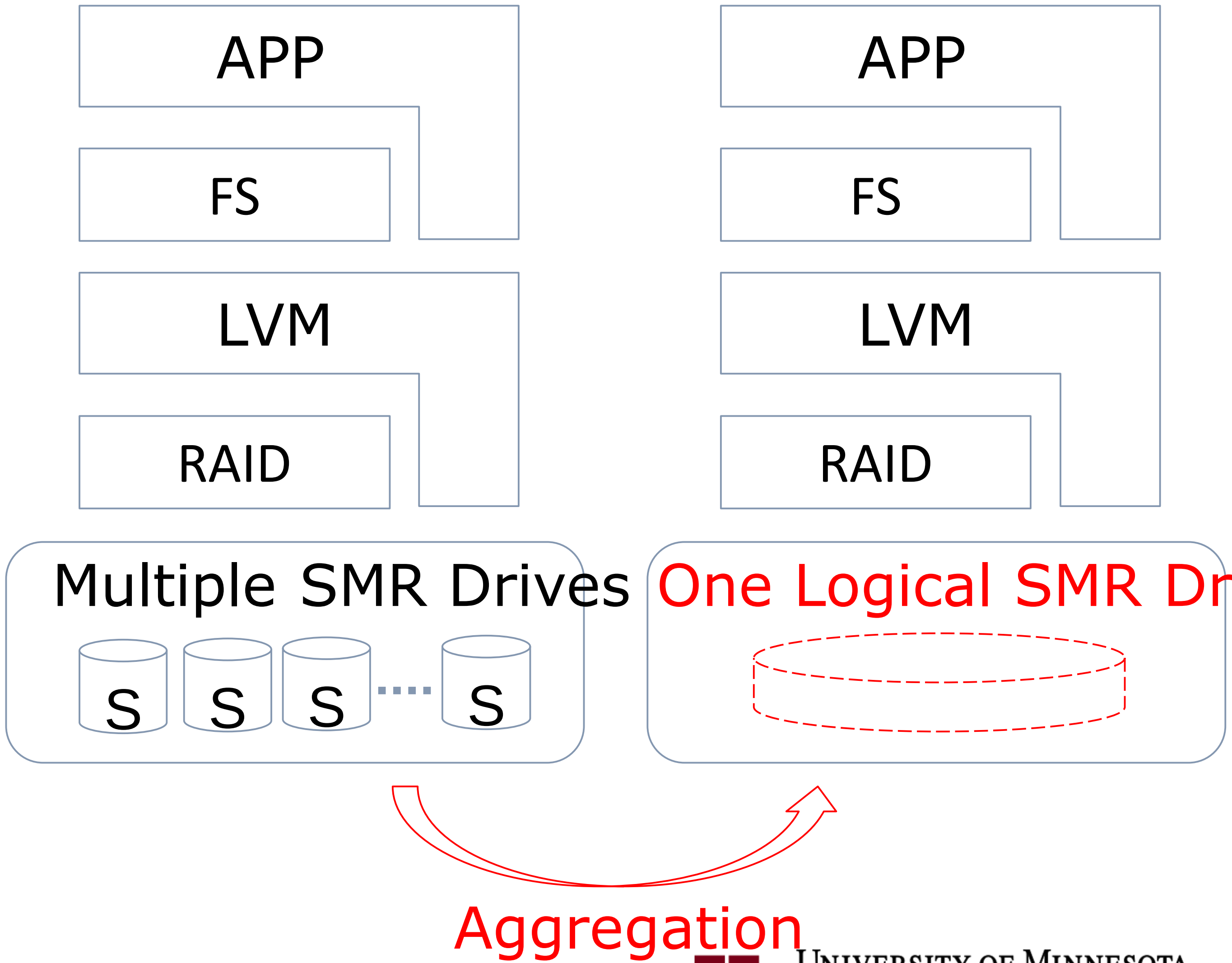
SMR Layout Awareness

- Avoid non-seq: Convert all workload to sequential
  - Always perform sequential write to SMR drive.
  - Achieving near-HDD performance.
- Accept non-seq: Know performance characteristic and tweak workload accordingly.
  - when to avoid non-sequential write; when to let it go.
  - reduce management overhead.
- Which layer to be SMR-Aware?
- Fully Aware or Partial Aware?
  - Hide/expose SMR information further up



SMR Drive Aggregation

- Abstract multiple physical SMR drives into logical one(s).
- Preserve the I/O characteristic
  - How much we can preserve?
  - Parametrized SMR drives
- Reduce design complexity, again.

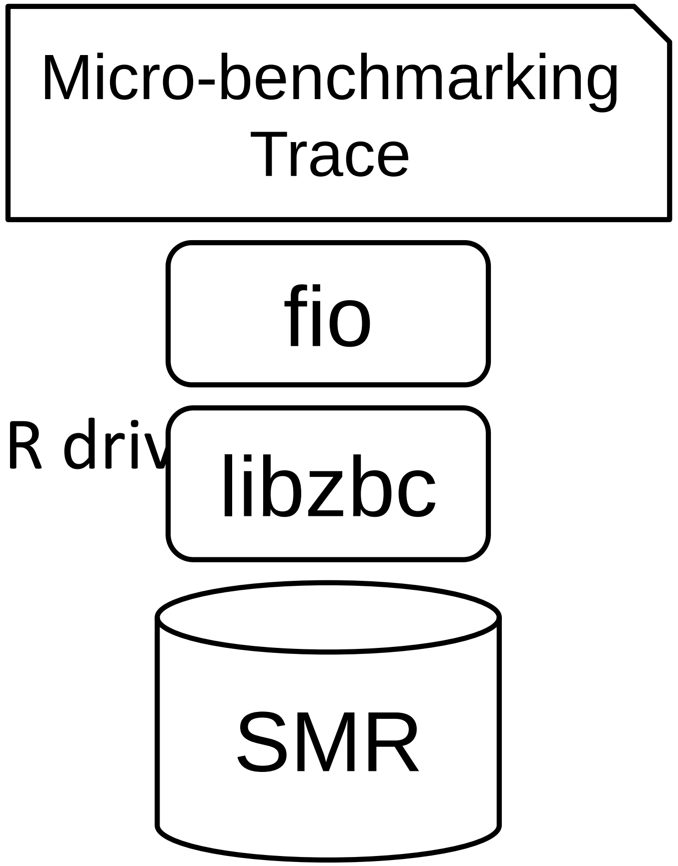


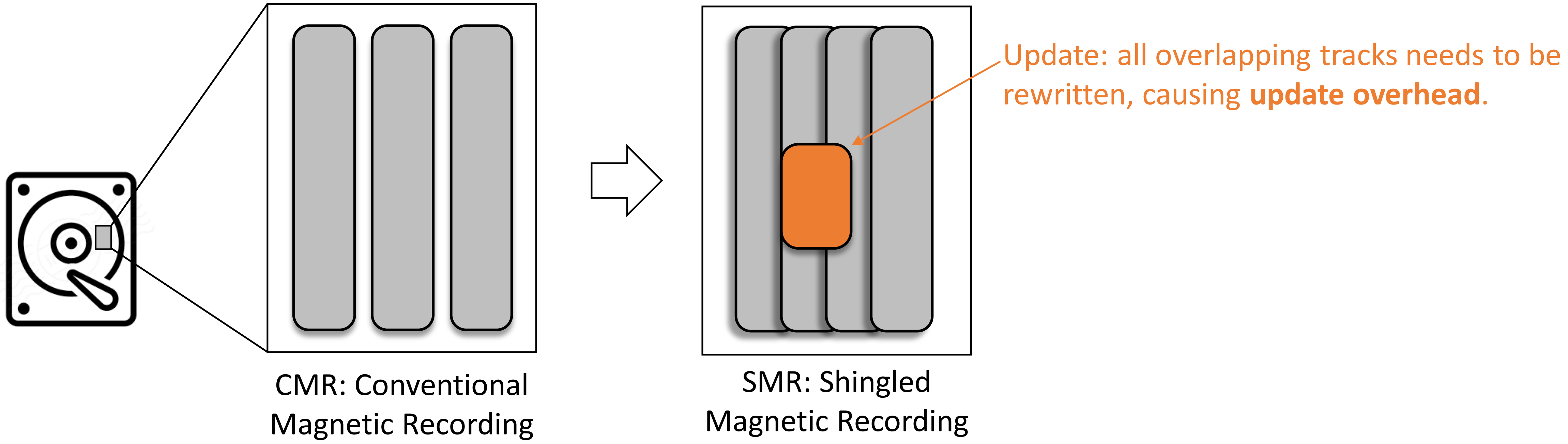
Progress So Far

- Understanding single SMR drive I/O performance for different workload
  - Defines “what to be aware of”.
  - Inspires “how to profile a logical aggregated SMR drive”.
- Indirection Buffer Design
  - One HA-SMR Awareness design.
  - Preliminary result shows its effectiveness.
- First version of prototype for aggregating SMR Drives (libvir)
  - User level implementation based on libzbc

Host Aware SMR (HA-SMR) Drive Testing

- Test goal focuses on unique features of HA-SMR:
  - Open zone issue
  - Non-sequential zone issue
  - Media cache cleaning efficiency
- Test Setup
  - Replay micro-benchmarking traces to HA-SMR driv

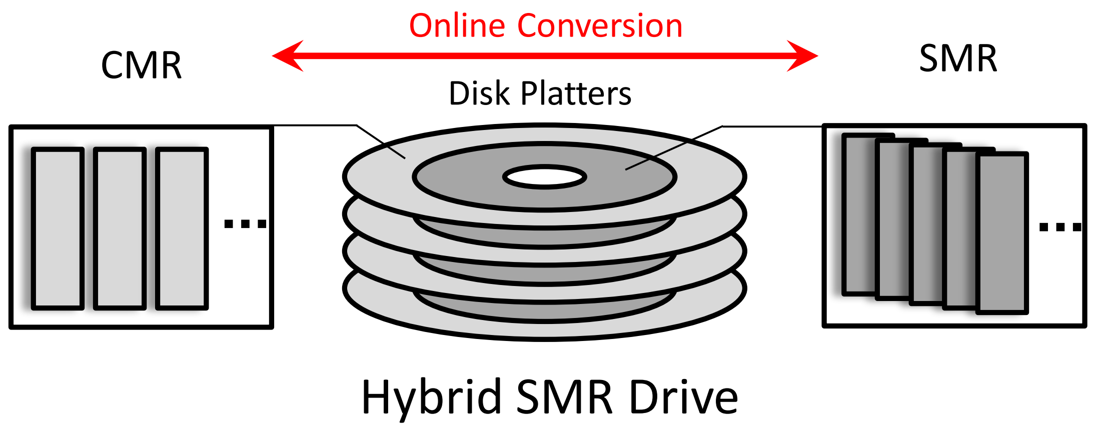



Update: all overlapping tracks needs to be rewritten, causing **update overhead**.
CMR: Conventional Magnetic Recording
SMR: Shingled Magnetic Recording

SMR: (+) more data density; (-) update overhead
CMR: (-) less data density; (+) no update overhead.

How about a **combination** of the two?

Emerging Hybrid SMR Drives



- **Hybrid SMR (H-SMR)**: mix of CMR and SMR; can be converted on line by H-SMR API.
- **Benefit**: utilize both IOPS and Capacity. Flexible and reconfigurable.

Objective: How do we efficiently manage the data and space for such Hybrid SMR drives?

Outline

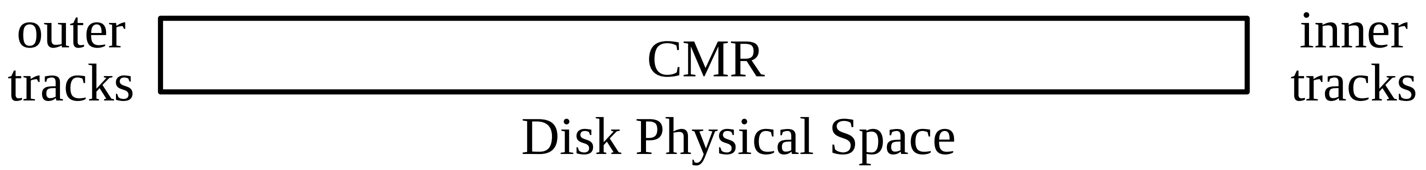
- Introduction
- Design Goals
- Background and Challenges
- Design and Evaluation
- Summary

Design Goal

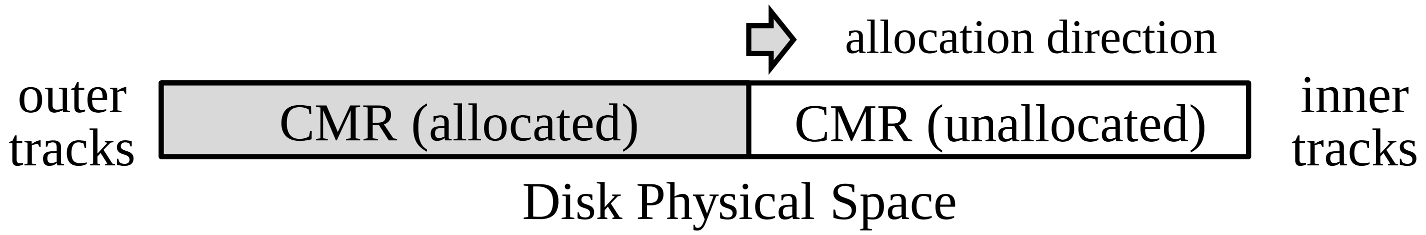
- Handle growing utilization.
- Reduce SMR update overhead.
- Adapt to dynamic workload.

Two-Phase Allocation

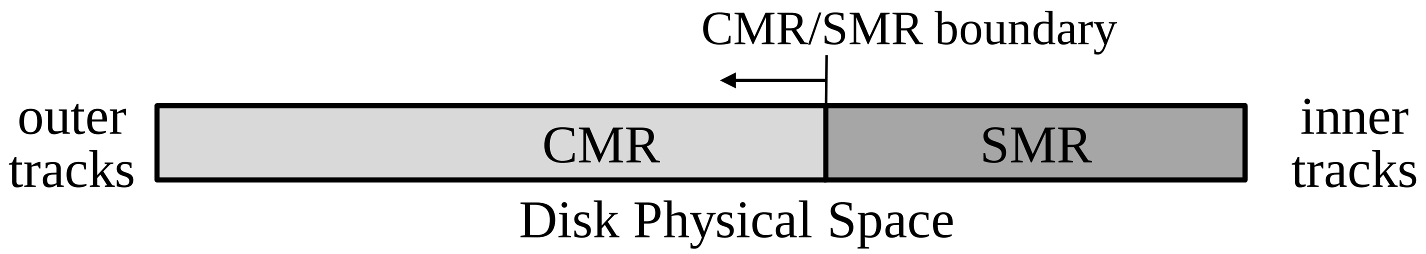
Initial State: Empty Disk



Phase I: CMR-only



Phase II: CMR + SMR

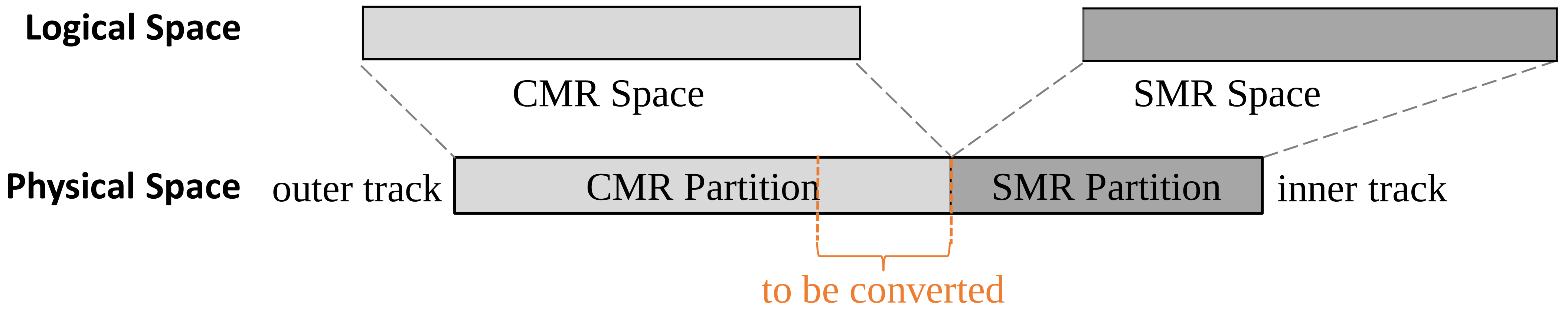


Intuition: use CMR first! then convert CMR to SMR when capacity is not enough.

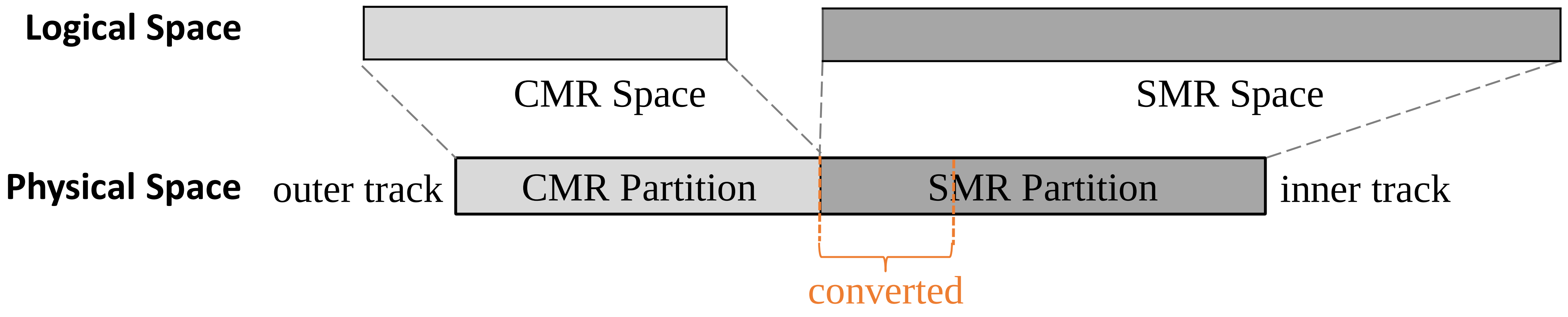
The devil is in the detail!

Buckle up. Challenge ahead!

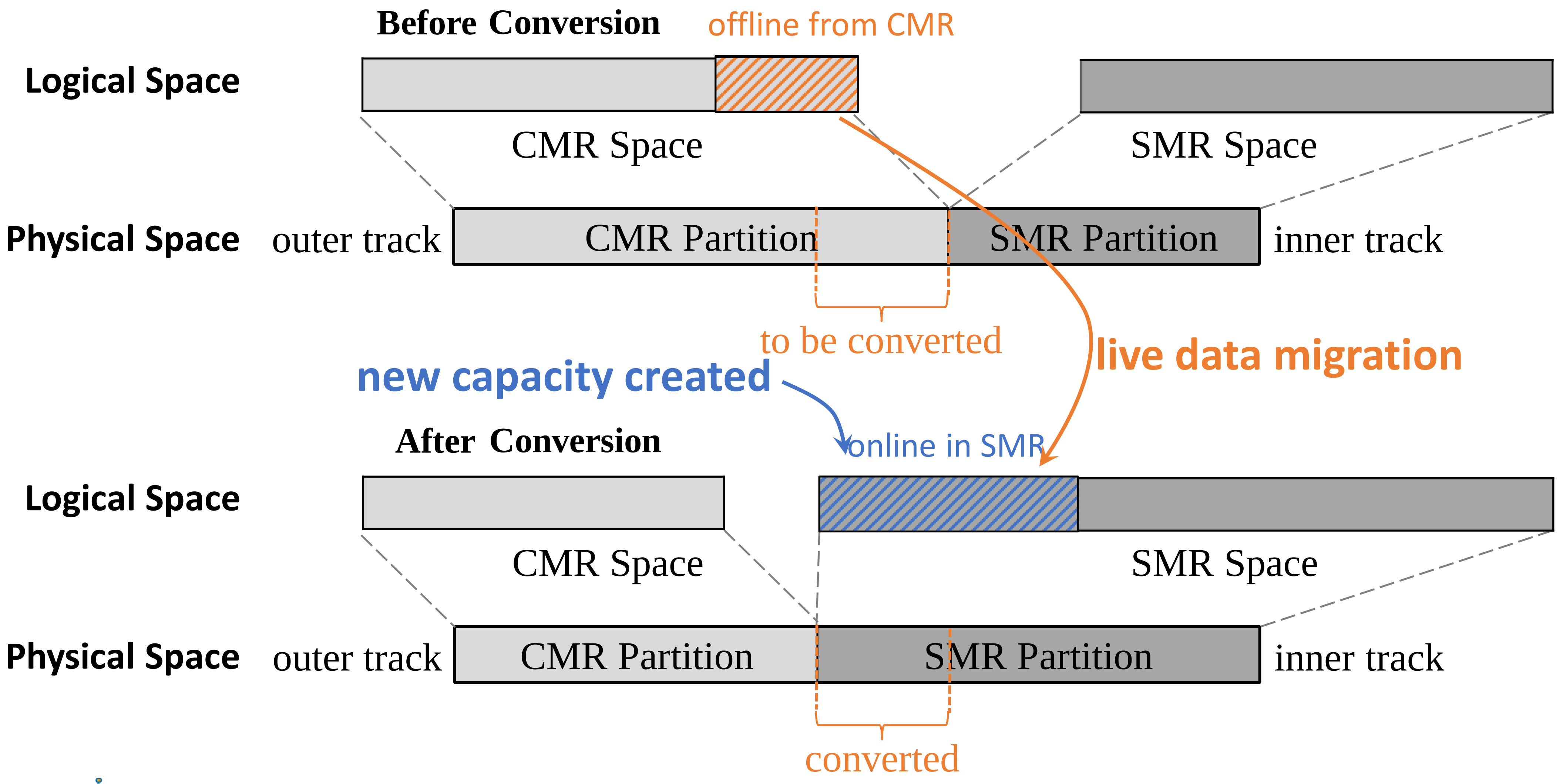
Background: Format Conversion



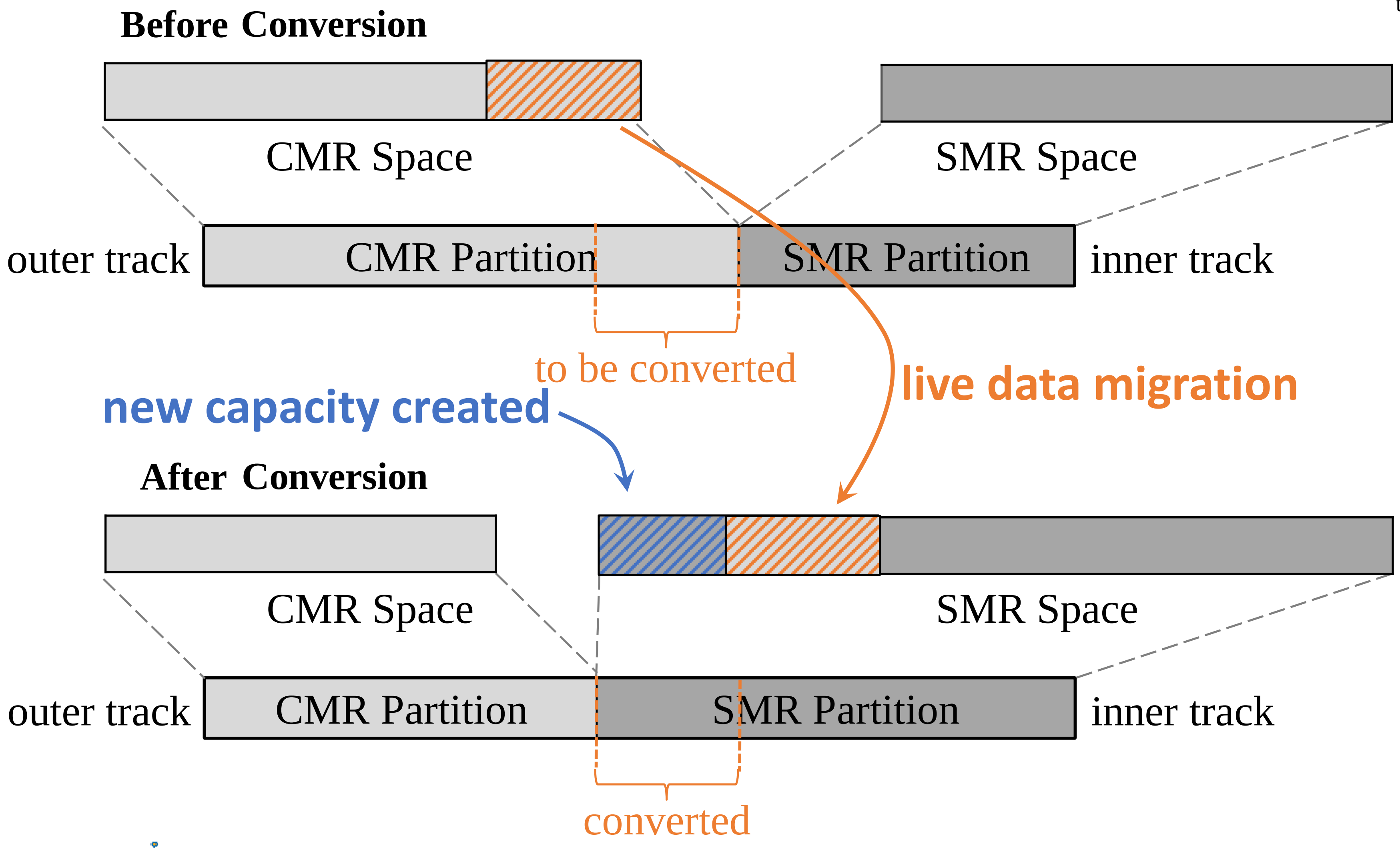
Background: Format Conversion

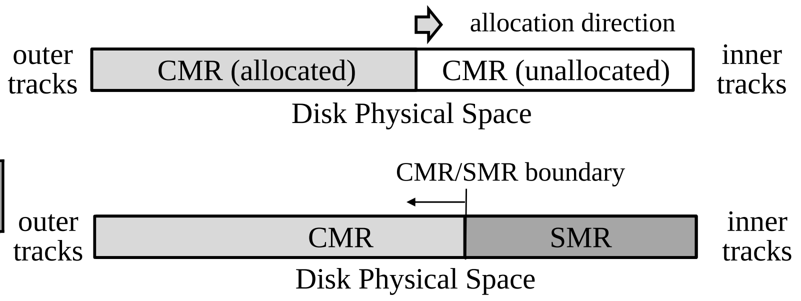


Background: Format Conversion



Format Conversion





Challenge:
- Address space / mapping
- SMR update overhead
- Live data migration cost

Center for Research in Intelligent Storage

UNIVERSITY OF MINNESOTA
Driven to Discover℠

Challenges and Solutions

Challenge:

- Address space / mapping
- SMR update overhead
- Live data migration cost

Solution:

- **Elastic** Address Space with Zone-level **Mapping**
- **H-Buffer** and **Zone-Swap** to reduce SMR update overhead
- **Quantized Migration** to mitigate live data migration cost in the format conversion

Challenges and Solutions

Challenge:

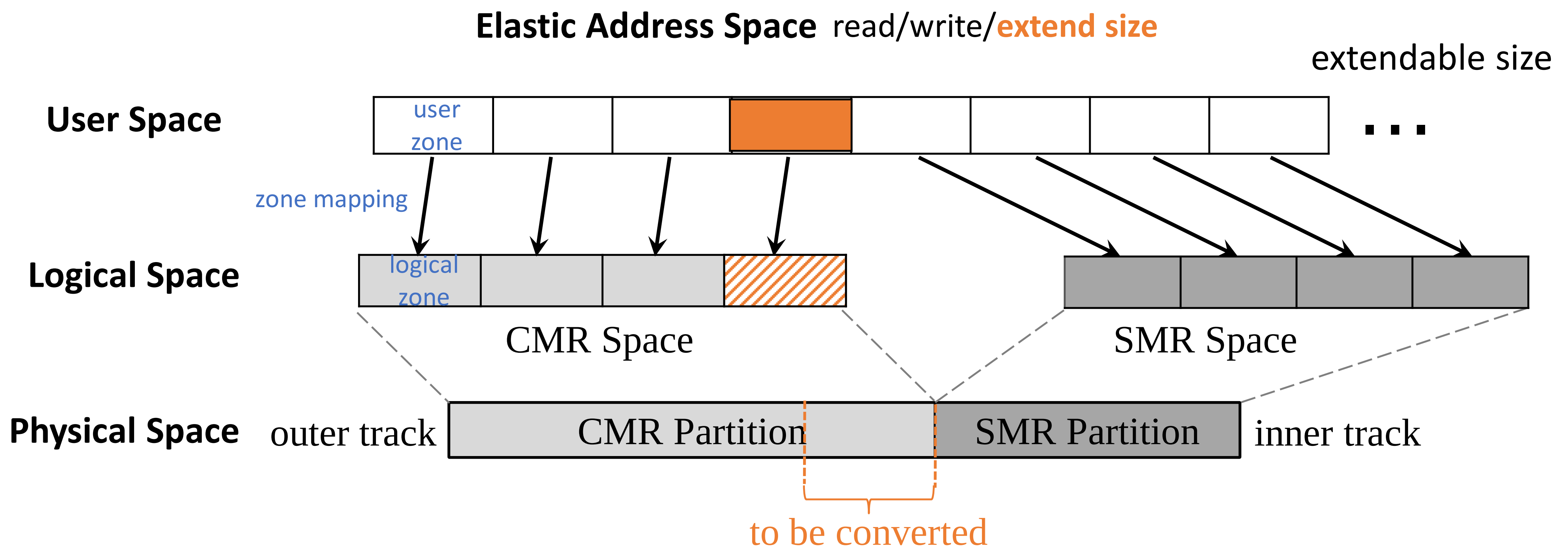
- Address space / mapping
- SMR update overhead
- Live data migration cost

Solution:

- **Elastic** Address Space with Zone-level **Mapping**
- **H-Buffer** and **Zone-Swap** to reduce SMR update overhead
- **Quantized Migration** to mitigate live data migration cost in the format conversion

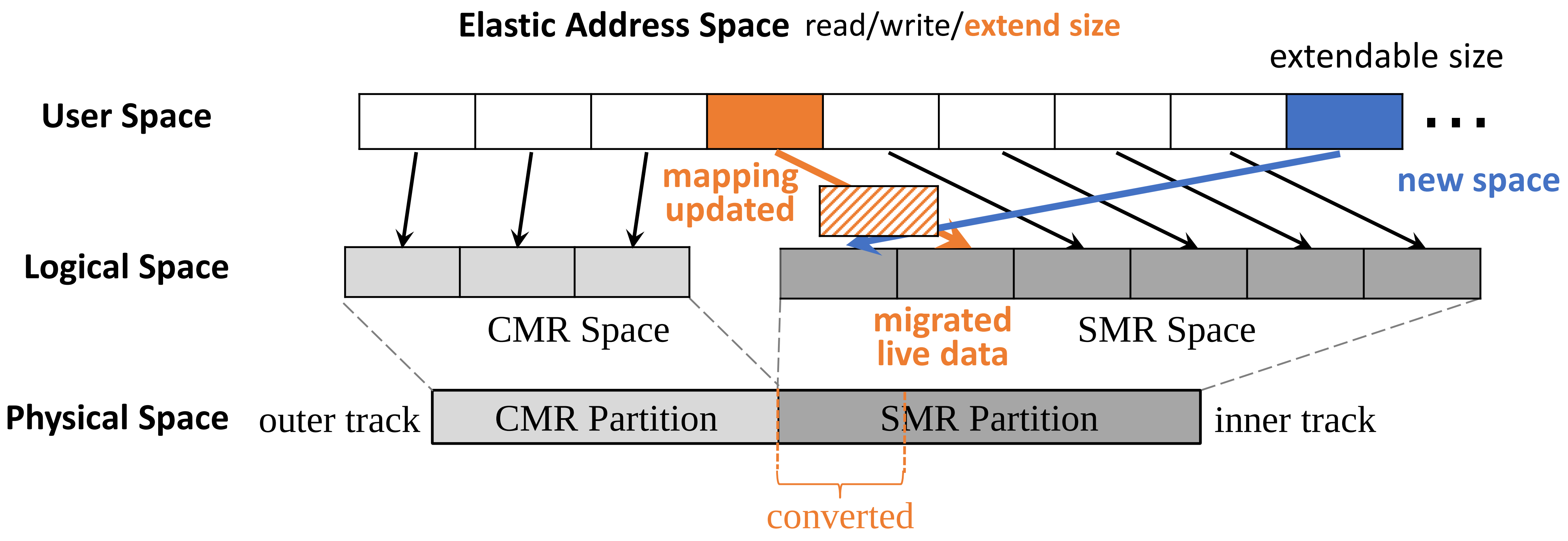
DESIGN I: ADDRESS SPACE / MAPPING

Elastic Address Space with Zone-level Mapping



Center for Research in Intelligent Storage

University of Minnesota
Driven to Discover℠

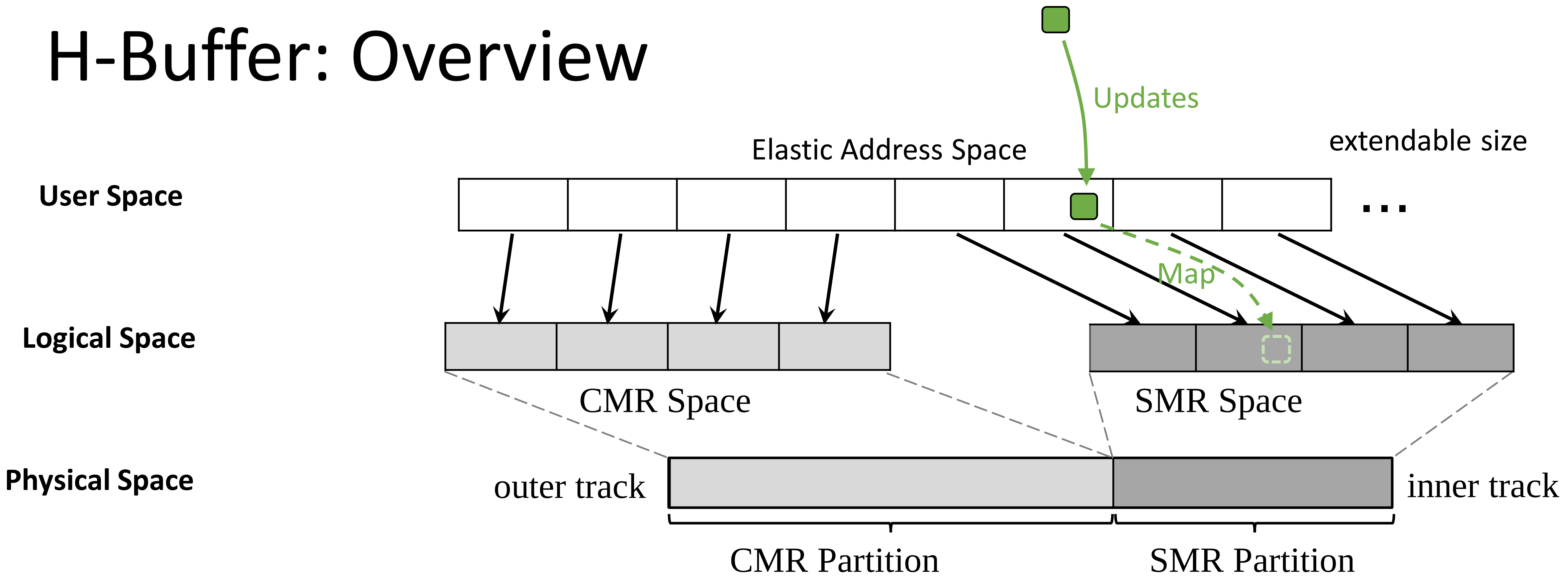


Elastic Address Space read/write/extend size
extendable size
User Space
. . .
mapping updated
new space
Logical Space
CMR Space
migrated live data
SMR Space
Physical Space
outer track
CMR Partition
SMR Partition
inner track
converted

DESIGN II: REDUCING SMR UPDATE OVERHEAD

H-Buffer and Zone-Swap

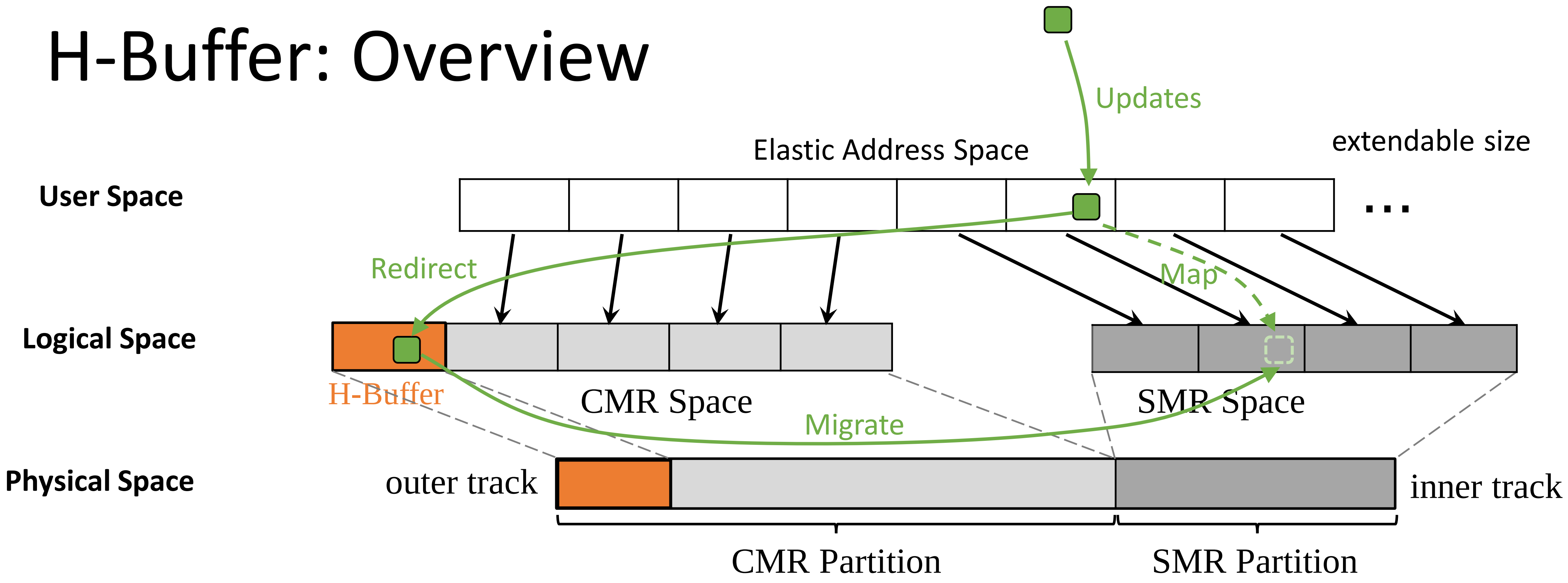
H-Buffer: Overview



Center for Research in Intelligent Storage

University of Minnesota Driven to Discover℠

H-Buffer: Overview

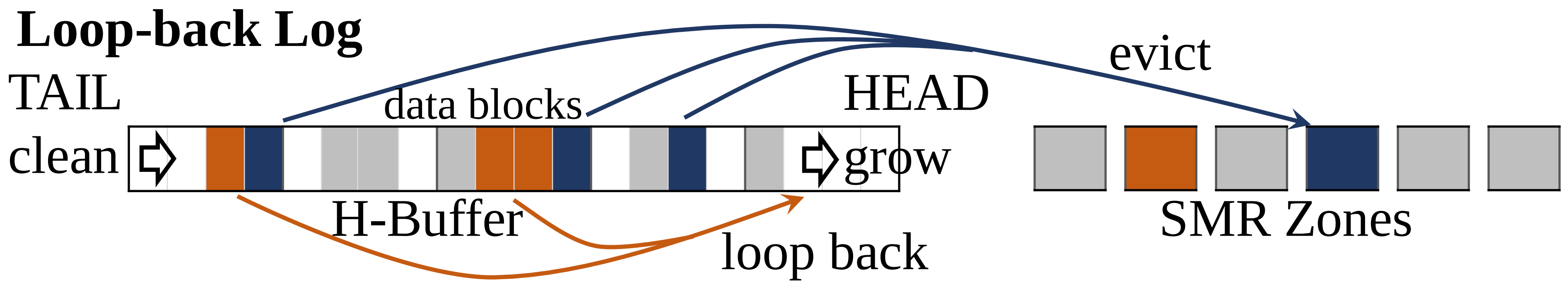


H-Buffer: Host-controlled Buffer

Basic Idea: using some reserved CMR space to buffer SMR updates and migrate to SMR zones later.

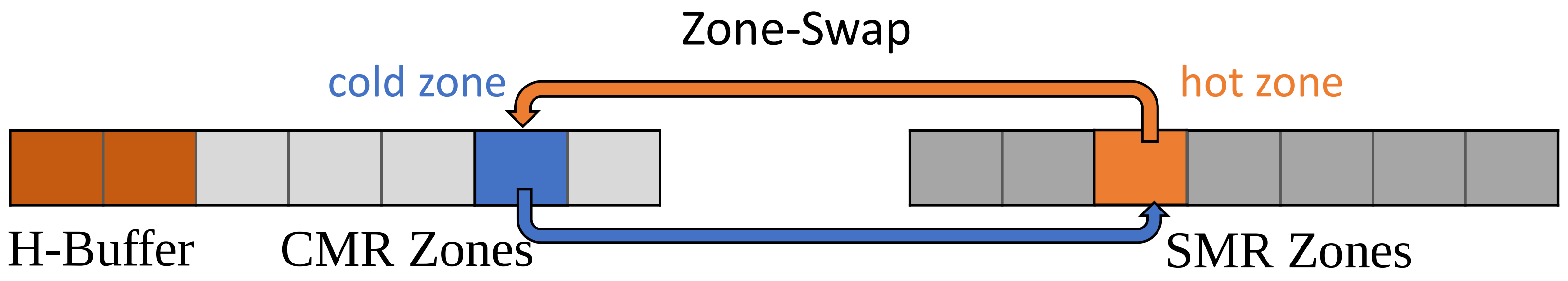
H-Buffer Management: Alternatives

- Block-~~based (e.g. LRU)~~ Problem: Random I/O in redirecting/cleaning
- Log-based
  - In-place FIFO
  - Loop-back Log (with hot/cold classification)



Idea: **Re-queue the hot data blocks** to the log head without evicting to SMR zones.

Zone-Swap: Overview

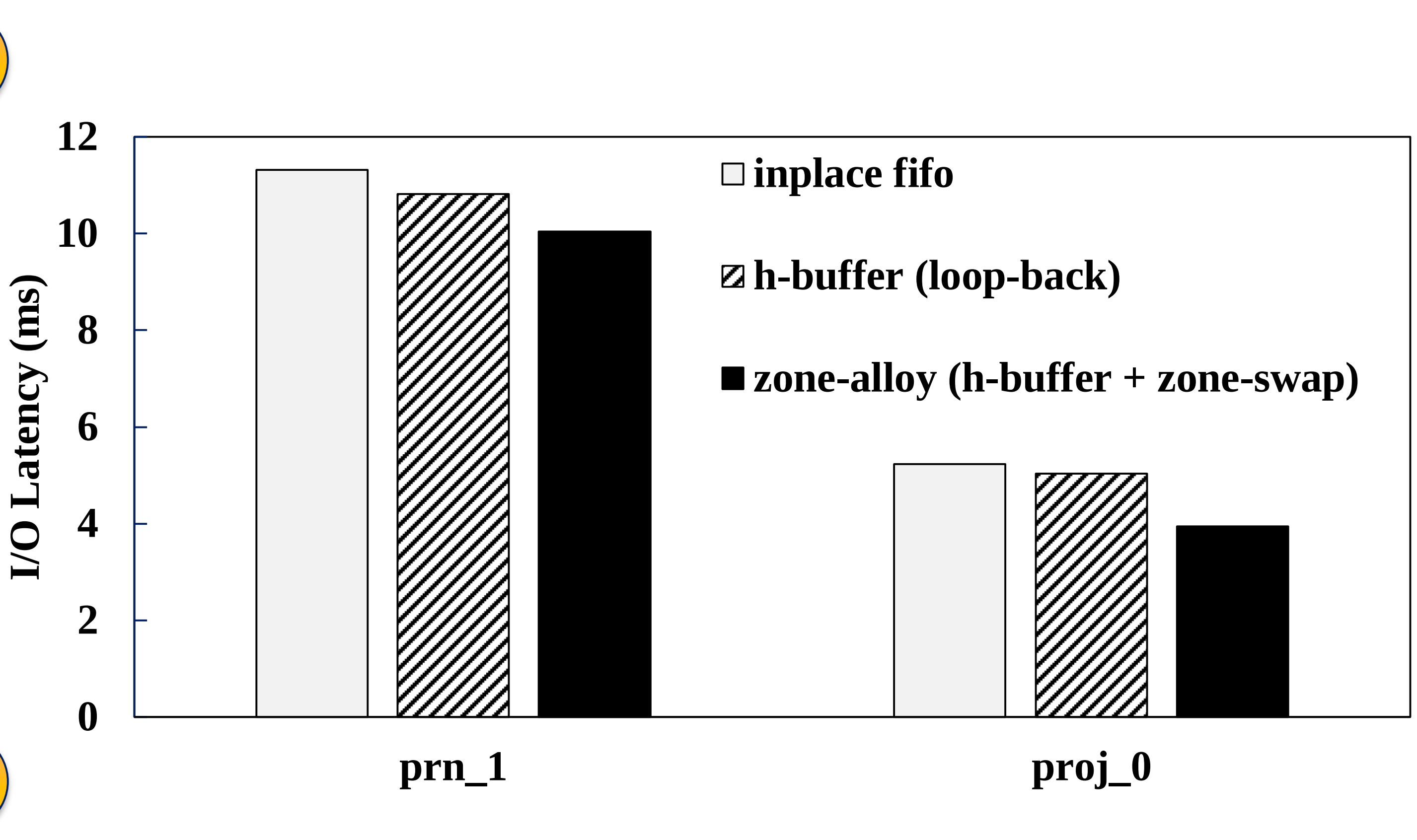


Co-design with H-Buffer:

- Swapping happens when H-Buffer evicts.
- H-Buffer eviction choice also depends on Zone-Swap decision.

Basic Idea: **swap** hot zones (heavily updated ones) from SMR to CMR to reduce SMR update overhead.

H-Buffer and Zone-Swap Evaluation

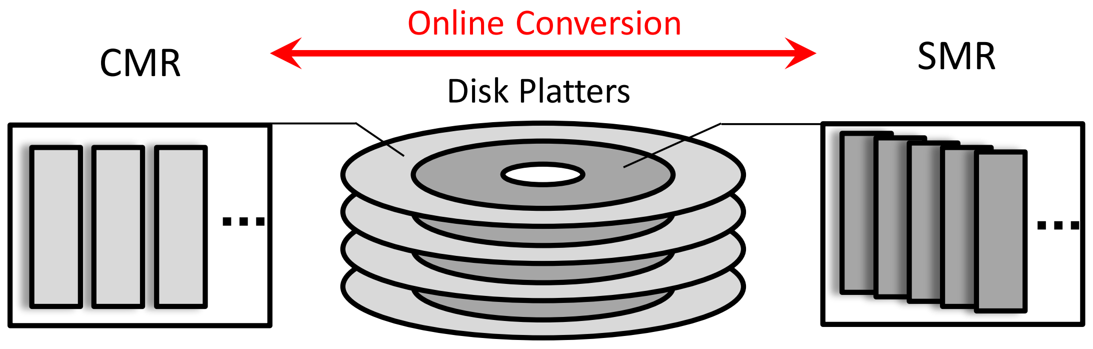


More result: in the poster session.

Summary: ZoneAlloy

-- Data management for Hybrid-SMR

Background



Hybrid SMR Drive

Problem

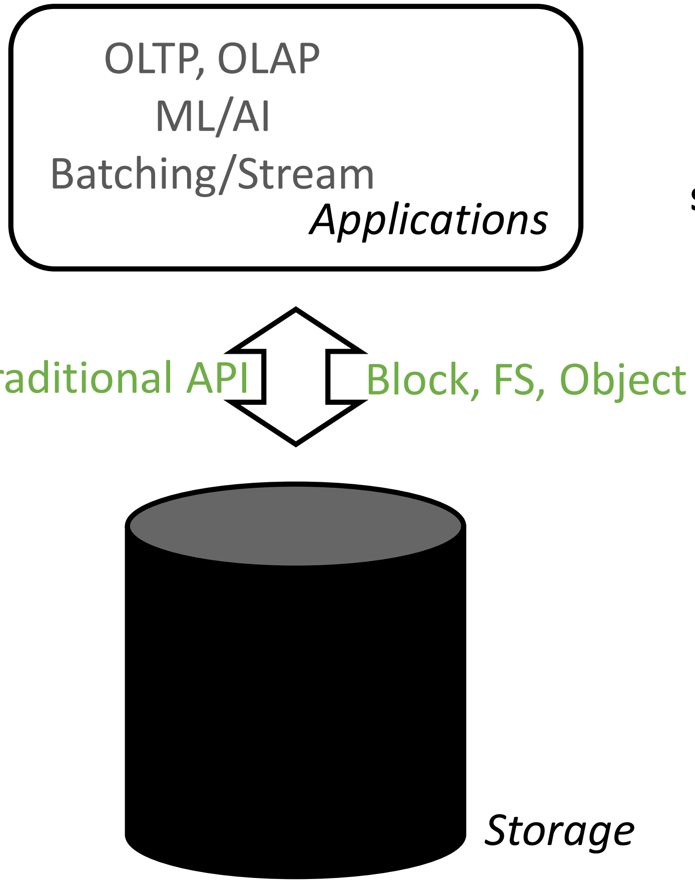
Data and Space Management in Hybrid SMR drives

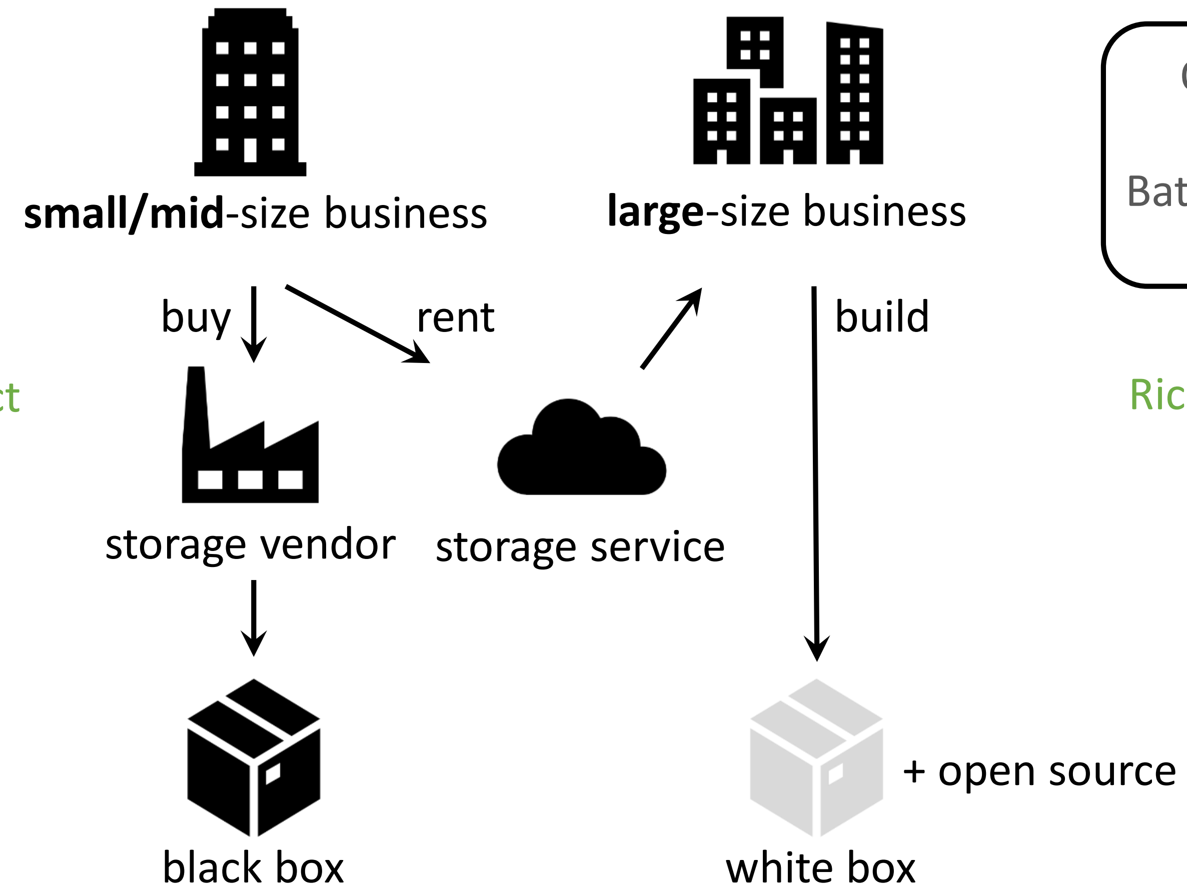
Solutions/Contributions

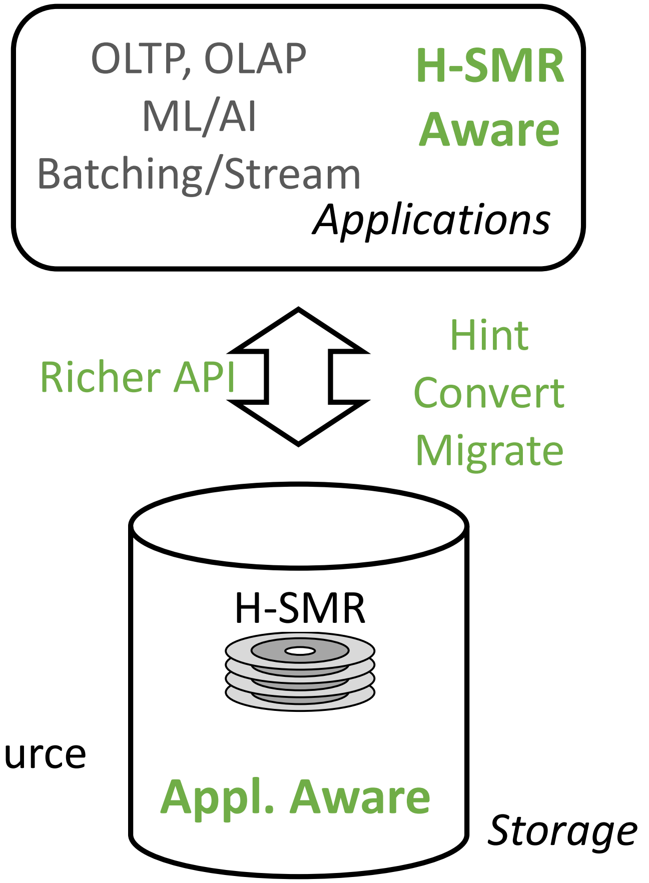
- Two-Phase Allocation
- Elastic Address Space/ Zone-level Mapping
- H-Buffer
- Zone-Swapping
- Quantized Migration

I/O stack change & API (ask for feedback)

Question: Which layer(s) of the I/O stack should do the heavy-lifting?

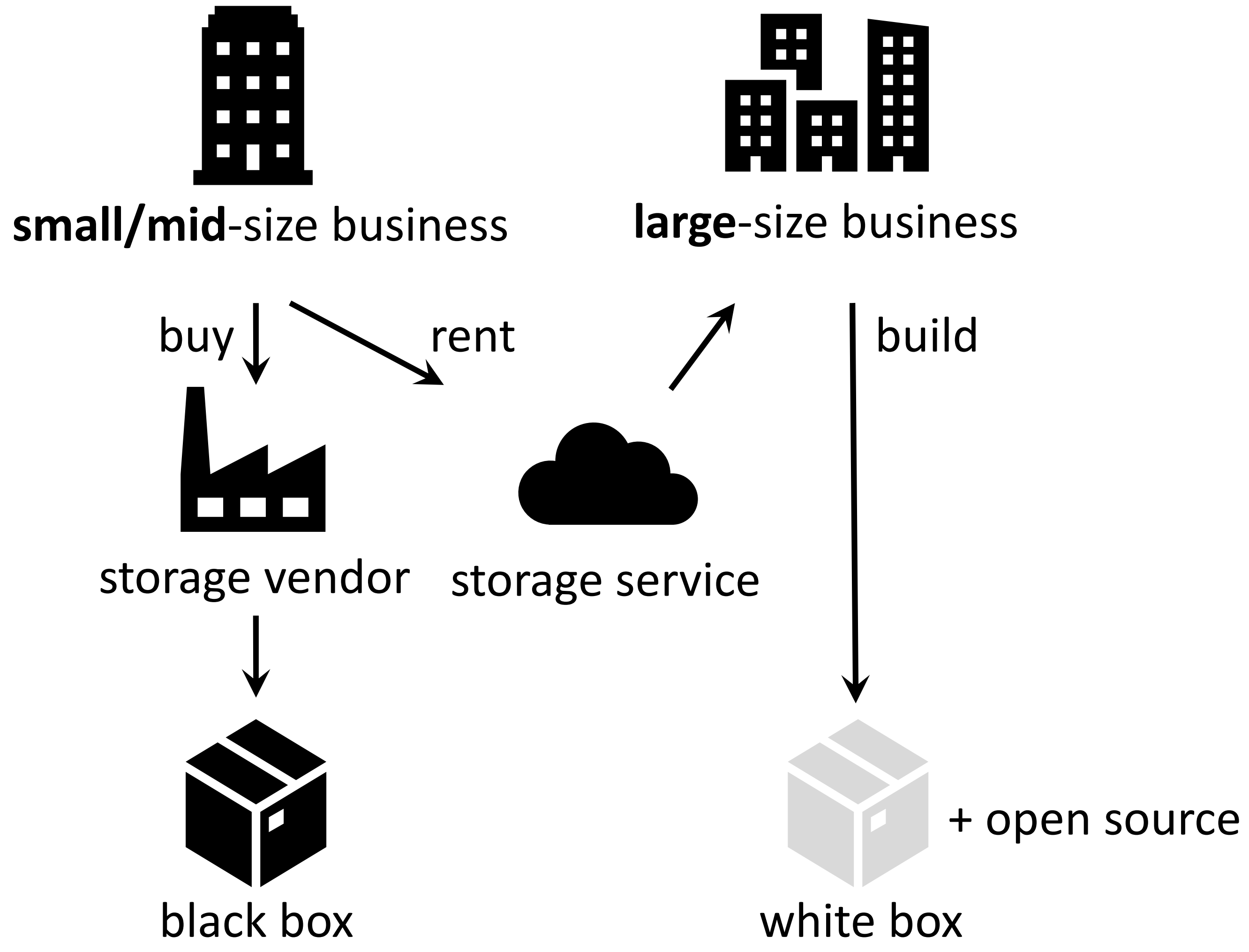






Thank You!
Questions?

Fenggang Wu
wuxx0835@umn.edu


small/mid-size business
large-size business
buy
rent
build
storage vendor
storage service
black box
white box
+ open source