BCW: Buffer-Controlled Writes to HDDs for SSD-HDD Hybrid Storage Server

Shucheng Wang¹, Ziyi Lu¹, Qiang Cao¹, Hong Jiang³, Jie Yao², Yuanyuan Dong⁴ and Puyuan Yang⁴

¹,² Huazhong University of Science and Technology,
³ University of Texas at Arlington,
⁴ Alibaba Group
Outline

• Background
  ➢ SSD-HDD hybrid storage

• Challenge and Motivation
  ➢ Unbalanced device utilization
  ➢ HDD buffered write behavior

• Design
  ➢ Buffer-Controlled Writes
  ➢ Mixed IO scheduler

• Evaluations
SSD-HDD hybrid storage

• SSD and HDD
  • SSD (Solid State Drive) has high-speed performance
  • HDD (Hard Disk Drive) has large capacity and low cost

• SSD-HDD hybrid storage
  ➢ Applications
  ➢ Properties
    ➢ Cost-effectiveness
    ➢ High-speed and low latency
Unbalanced Device Utilization

• SSDs suffer from high write pressure

Highly-intensive writes in each SSD
- Peak write requests per second > 10KRPS
- Data written per day > 3 TB

Long tail latency
- 99th-percentile write latency > 10ms
- 99.9th-percentile write latency > 50ms

Large queue length
- More than 100 blocked write requests in the queue
Unbalanced Device Utilization

- HDDs are underutilized

Low HDD utilization
- $\frac{1}{2} - \frac{1}{5}$ of SSD utilizations
- 90% – 95% of times are in idle state
We want to exploit the **underutilized HDD** to relieve the pressure of SSDs in hybrid storage nodes.
Challenges

Method is feasible when:

\[
\text{Average Latency of HDD} < \text{Tail Latency of SSD}
\]

Could the HDDs reach to $\mu$s-level write latency?
Motivational Test

• Issue **sequential** and **continuous** writes to HDDs
  
  *(Append-only) (Close-loop)*

• Tested HDD Devices

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Capacity</th>
<th>Model</th>
<th>Recording Technology</th>
<th>Sequential Write Bandwidth (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>West Digital</td>
<td>10TB</td>
<td>WD100EFAAX</td>
<td>PMR</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>8TB</td>
<td>WD8004FRYZ</td>
<td>PMR</td>
<td>180</td>
</tr>
<tr>
<td></td>
<td>4TB</td>
<td>WD40EZRZ</td>
<td>PMR</td>
<td>180</td>
</tr>
<tr>
<td>Seagate</td>
<td>8TB</td>
<td>ST8000DM0004</td>
<td>PMR</td>
<td>180</td>
</tr>
<tr>
<td></td>
<td>4TB</td>
<td>ST4000DM004</td>
<td>SMR</td>
<td>180</td>
</tr>
</tbody>
</table>
Results Overview

10TB West Digital HDD
HDD write behaviors

HDD can reach μs-level write latency, especially for small size requests.

- 35us for 4KB writes
- 66us for 16KB writes
- 180us for 64KB writes
HDD write behaviors

Reach **us-level** write latency, especially for small size requests

**ms-level** latency spikes

*Higher than 10ms for some spikes*
HDD write behaviors

Reach to **us-level** write latency, especially for small size requests

**ms-level latency spikes**

Fixed write latency period

The length of the fastest write stage is **16MB**

The interval between two highest latency spikes is **8MB**
How does this happen?

write IO

Built-in Buffer

disc

HDD Device
Challenges

- HDDs can reach $\mu$s-level write latency
- How to control these $\mu$s-level latency writes in HDDs?
HDD Buffered-Write Model

- Three types of HDD buffered writes
  - Fast write (low-latency)
  - Mid write (mid-latency)
  - Slow write (high-latency spike)
HDD Buffered-Write Model

- Three types of HDD buffered writes
  - \( F \)ast write (low-latency)
  - \( M \)id write (mid-latency)
  - \( S \)low write (high-latency spike)

- Buffered Write Sequence (after a Sync)
  Starts with a \textbf{Fast stage}, followed by one or more \textbf{Slow-and-Mid stage-pairs}
  - \textit{Fast} stage lasts for \( W_f \) data written
  - \textit{Mid} stage lasts for \( W_m \) data written
  - \textit{Slow} stage contains an \( S \)low write
Write-state Predictor

• The next HDD write state could be predicted, according to:
  ① Write state of the current request
    • Each write request can only be one of the \( F, M, \) or \( S \)
  ② Current \( ADW, W_f \) and \( W_m \) values
    • A: \( ADW < W_f \) and \( W_m \)
    • U: \( ADW \geq W_f \) and \( W_m \)
  ③ Sync operation
    • Takes the next write state back to \( F \)
Buffer-Controlled Writes (BCW)

• Goal:
  - Ensures user writes to be in the $F$ or $M$ write state, avoids writes in the $S$ low state.

• Steps:
  - Perform **profiling** for all key parameters (if unknown)
  - Invokes **Sync** operation when starting BCW
    - To flush all data in the HDD build-in buffer
Buffer-Controlled Writes (BCW)

• Write user data to HDD when there are user requests in the queue

• **Actively pads non-user data** to HDD when there are no user requests
  - **PF**: for padding the $F$ and $M$ stage (i.e., 4KB)
  - **PS**: for padding the $S$ stage (i.e., 64KB)
Buffer-Controlled Writes (BCW)

- When current data written (\(ADW\)) in the \(F\) and \(M\) states are close to the \(Wf\) and \(Wm\) values
  ① The next HDD write state will be **predicted as \(S\)**
Buffer-Controlled Writes (BCW)

• When current data written (ADW) in the $F$ and $M$ states are close to the $Wf$ and $Wm$ values
  ① The next HDD write state will be predicted as $S$
  ② Stop receiving user requests
Buffer-Controlled Writes (BCW)

- When current data written (\(ADW\)) in the \(F\) and \(M\) states are close to the \(W_f\) and \(W_m\) values
  ① The next HDD write state will be predicted as \(S\)
  ② Stop receiving user requests
  ③ Continuously pads \(PS\), until a \(S\) write state is detected
Buffer-Controlled Writes (BCW)

- When current data written (ADW) in the $F$ and $M$ states are close to the $Wf$ and $Wm$ values
  1. The next HDD write state will be predicted as $S$
  2. Stop receiving user requests
  3. Continuously pads PS, until a $S$ write state is detected
  4. **Start** receiving user requests

![Diagram of Buffer-Controlled Writes (BCW) with time line showing user, Flush, PS, and Sync states. The diagram illustrates the sequence of operations and states over time.]
Buffer-Controlled Writes (BCW)

• Same steps of padding PF and PS in $M$ stages
Challenges

- HDDs can reach $\mu$s-level write latency
- BCW provides a proactive and controllable buffer writing approach
- How to leverage BCW effectively in the hybrid storage nodes?
Mixed IO scheduler (MIOS)

• A scheduler that schedules mixed IOs at runtime

• Architecture
  • A request queue for each SSD and HDD, and monitors their length $l(t)_{SSD}$ and $l(t)_{HDD}$
  • Log file in each HDD, a device file storing BCW writes in an append-only manner.
Mixed IO scheduler (MIOS)

• Key parameters:

Threshold $L$: 
SSD write with $l(t)_{SSD}$ ≥ $M$ state in HDD

$flag_{HDD}$: is HDD available (BCW controlled)?
Scheduling strategies in MIOS

**MIOS_D:**
redirect SSD writes to HDDs when:
- $l_{SSD}$ is **higher** than the threshold $L$ **AND**
- HDD is the $F$ or $M$ write state with BCW

**MIOS_E:**
- **Same** as MIOS_D when $l_{SSD} > L$
- Further perform redirection with just $F$ write state when $l_{SSD} < L$
## Evaluation Setup

- **Comparisons**
  - **Baseline**: Pangu workload replay (writing all data into SSDs)
  - MIOS_D
  - MIOS_E

- **Evaluation environment**

<table>
<thead>
<tr>
<th>System</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel Xeon E5-2696 v4 (2.20 GHz, 22 CPUs)</td>
</tr>
<tr>
<td>Memory</td>
<td>128 GB</td>
</tr>
<tr>
<td>HDDs</td>
<td>West Digital 10TB (default)</td>
</tr>
<tr>
<td></td>
<td>West Digital 4TB</td>
</tr>
<tr>
<td></td>
<td>Seagate 4TB</td>
</tr>
<tr>
<td>SSDs</td>
<td>Samsung 960EVO 256GB (NVMe, 2000MB/s)</td>
</tr>
</tbody>
</table>
### Evaluation Setup

- Workload characteristics

<table>
<thead>
<tr>
<th>Workload Types</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business</td>
<td>Cloud Computing</td>
<td>Cloud Storage</td>
<td>Structured Storage</td>
<td>Structured Storage</td>
</tr>
<tr>
<td>SSD Writes (GB)</td>
<td>14.7</td>
<td>61.2</td>
<td>7.2</td>
<td>7.5</td>
</tr>
<tr>
<td>SSD Write Requests (millions)</td>
<td><strong>0.43</strong></td>
<td>4.4</td>
<td>4.8</td>
<td>4.7</td>
</tr>
<tr>
<td>Note</td>
<td>Lowest IO intensity</td>
<td>Most written data</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Write Performance

- Average, 99\textsuperscript{th} and 99.9\textsuperscript{th}-percentile tail latency
  - Reduced by 65%, 85%, and 95% respectively in workload B
  - Reduced by 2%, 3.5% and 30% respectively in workload A
Queue Length Reduction

Workload A
Minimum (15%) reduction in queue lengths

Workload B
Maximum (95%) reduction in queue lengths
SSD Written Data Reduction

• **MIOS_D:**
  - Reduced *5.5%* compared with *Baseline* in workload A
  - Reduced *15.3%* and *16%* in workload C and D

• **MIOS_E:**
  - Reduced *93.3%* compared with *Baseline* in workload A
  - *71%* and *72%* less than *Baseline* in workload C and D
Write Performance: MIOS_D vs MIOS_E

- MIOS_E leads to worse latency performance
  - MIOS_E leads to **1.4x** higher average latency, **1.7x** higher 99\(^{th}\)-percentile latency, **6.6x** higher 99.9\(^{th}\)-percentile latency than MIOS_D in workload A

**Tradeoff**

![Graph showing latency comparison between MIOS_E and MIOS_D](image-url)
Experiment with other HDDs

• Different types of HDDs do not have a significant impact
  • The maximum difference of average and tail latency is less than 3%
  • Amount of data written to and number of requests processed in SSD with different HDDs is less than 5%

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>WD 10TB</th>
<th>WD 4TB</th>
<th>SE 4TB</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSD written data (GB)</td>
<td></td>
<td>61.2</td>
<td>4.1</td>
<td>4.2</td>
</tr>
<tr>
<td>SSD write requests (thousands)</td>
<td>4453</td>
<td>720</td>
<td>724</td>
<td>769</td>
</tr>
</tbody>
</table>
Thanks! Q&A

Email: wsczq@hust.edu.cn