ADAPT: Efficient Workload-sensitive Flash Management Based on Adaptation, Prediction and Aggregation

Chundong Wang and Weng-Fai Wong
National University of Singapore
Outline

• Introduction
  ➢ NAND Flash Memory
  ➢ FTL and workload
• Background of hybrid mapping
• ADAPT
  ➢ Adaptive partitioning of log space
  ➢ Prediction and Aggregation
• Evaluation
• Conclusion
Outline

• Introduction
  ➢ NAND Flash Memory
  ➢ FTL and workload

• Background of hybrid mapping

• ADAPT
  ➢ Adaptive partitioning of log space
  ➢ Prediction and Aggregation

• Evaluation

• Conclusion
Characteristics of NAND flash memory

- Three operations: write, read and erase

- Out-of-place updating
  - A page cannot be programmed (written) unless its block is erased first;
  - A block usually comprises multiple pages.
Introduction

• FTL: Flash Translation Layer
  ➢ Embedded software for flash management
  ➢ Functionalities:
    □ Address mapping
    □ Wear leveling
    □ Garbage collection
    □ Bad block management
Introduction

- The roadmap of FTL (address mapping)
Introduction

• Various workloads
  ➢ Distinct access behaviors to secondary storage
Introduction

- Workloads’ characteristics: I/O request size.

<table>
<thead>
<tr>
<th>Workload</th>
<th>Small I/O (%)</th>
<th>Medium I/O (%)</th>
<th>Large I/O (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPC-C_20</td>
<td>99.17</td>
<td>0.83</td>
<td>0.00</td>
</tr>
<tr>
<td>SPC1</td>
<td>86.58</td>
<td>10.63</td>
<td>2.79</td>
</tr>
<tr>
<td>MSR-hm_0</td>
<td>76.70</td>
<td>13.72</td>
<td>9.58</td>
</tr>
<tr>
<td>MSR-mds_0</td>
<td>72.35</td>
<td>19.79</td>
<td>7.86</td>
</tr>
<tr>
<td>MSR-prn_0</td>
<td>79.46</td>
<td>8.88</td>
<td>11.66</td>
</tr>
<tr>
<td>MSR-prxy_0</td>
<td>87.91</td>
<td>6.82</td>
<td>5.27</td>
</tr>
<tr>
<td>MSR-rsrch_0</td>
<td>68.22</td>
<td>25.04</td>
<td>6.74</td>
</tr>
<tr>
<td>MSR-stg_0</td>
<td>72.33</td>
<td>18.62</td>
<td>9.05</td>
</tr>
<tr>
<td>MSR-ts_0</td>
<td>67.81</td>
<td>25.87</td>
<td>6.32</td>
</tr>
<tr>
<td>MSR-web_0</td>
<td>67.50</td>
<td>23.85</td>
<td>8.65</td>
</tr>
</tbody>
</table>
Introduction

• A workload:
  - is mixed by sequential and random requests;
  - but has stable access behaviors.

• The impact on FTL design by workloads:
  - An FTL may be designed for one type of workload, e.g., FASTer for OLTP systems (SNAPI 2010);
  - or be workload-adaptive, e.g., WAFTL (MSST 2011).
Outline

• Introduction
  ➢ NAND Flash Memory
  ➢ FTL and workload

• Background of hybrid mapping

• ADAPT
  ➢ Adaptive partitioning of log space
  ➢ Prediction and Aggregation

• Evaluation

• Conclusion
Hybrid Mapping

- Page-level mapping and block-level mapping;
- Hybrid mapping is a combination of them.

(a) Page Mapping

(b) Block Mapping
Hybrid Mapping

• Why to be “hybrid”:
  ➢ Block mapping
    □ Space economic
    □ Inflexible
  ➢ Page mapping
    □ Fine granularity
    □ Large mapping table;
  ➢ Hybrid mapping takes advantage of them.
Hybrid Mapping

• How to be “hybrid”:
  ➢ Partitions of physical blocks
    □ Data blocks: block mapping;
    □ Log space: page mapping;
    □ Free block pool: to provide clean blocks.
  ➢ Log space is like a cache to data blocks.
Hybrid Mapping

Block Mapping Table

<table>
<thead>
<tr>
<th>Logical Block No</th>
<th>Physical Block No</th>
</tr>
</thead>
<tbody>
<tr>
<td>LB₀</td>
<td>D₀</td>
</tr>
<tr>
<td>LB₁</td>
<td>D₁</td>
</tr>
<tr>
<td>......</td>
<td>......</td>
</tr>
</tbody>
</table>

Data Block No

D₀

Data Pages

1 6

Log Blocks

L₀ L₀

Log Space

L₀ L₀

Log Page Mapping Table

<table>
<thead>
<tr>
<th>Data Block No</th>
<th>Data Page No</th>
<th>Log Block No</th>
<th>Log Page No</th>
</tr>
</thead>
<tbody>
<tr>
<td>D₀</td>
<td>1</td>
<td>L₀</td>
<td>0</td>
</tr>
<tr>
<td>D₁</td>
<td>2</td>
<td>L₀</td>
<td>1</td>
</tr>
<tr>
<td>......</td>
<td>......</td>
<td>......</td>
<td>......</td>
</tr>
</tbody>
</table>
Hybrid Mapping

• Log space is further partitioned
  ➢ Sequential area for sequential requests
  ➢ Random area for random requests
• When log pages are used up, *merge* is called.
• Merge: to make room in log space
  ➢ Switch merge
  ➢ Partial merge
  ➢ Full merge *(random area)*
  ➢ More preferred (sequential area)
Hybrid Mapping

- Three types of merge

(This figure is adapted from LAST of Lee et.al. in SIGOPS Oper. Syst. Rev.)
Outline

• Introduction
  ➢ NAND Flash Memory
  ➢ FTL and workload
• Background of hybrid mapping
• ADAPT
  ➢ Adaptive partitioning of log space
  ➢ Prediction and Aggregation
• Evaluation
• Conclusion
• Overview
  - ADAPT is a hybrid mapping scheme
    - Fully-associative, like FAST and FASTer
    - Log blocks will be managed in a FIFO queue
  - ADAPT’s components:
    - Adaptive partitioning of log space
    - Predictive transfers to avoid premature merge
    - Aggregated data movements
ADAPT’s Adaptation

- Previously
  - Partitions of two areas were fixed;
  - How to identify a request to be random or sequential was not adaptive.

- ADAPT
  - dynamically adjusts two areas online;
  - identifies requests in an adaptive way.

- Workloads are dynamic.
• Key Idea:

If performance suffers from insufficient random log blocks, use blocks from sequential area,

and vice versa.
Two variables to detect performance:

- \( \delta \) = \( \frac{\text{count of switch and partial merge}}{\text{count of sequential log block allocation}} \) \( \in [0, 1] \);

- \( \varphi \) = \( \frac{\text{count of merged pages in full merges}}{\text{count of full merge}} \) \( \leq \text{Block size} \);

- \( \delta \): sequential area; \( \varphi \): random area.
ADAPT’s Adaptation

• How to use $\delta$ and $\varphi$
  ➢ In an interval, $\delta$ and $\varphi$ are measured;
  ➢ If $\delta > 0.4$, to enlarge sequential area;
  ➢ Else if $\varphi \geq \frac{\text{Block Size}}{2}$, to enlarge random area.

• Why?
  ➢ Larger $\delta \rightarrow$ a higher hit rate in sequential area;
  ➢ Larger $\varphi \rightarrow$ full merge to process more valid pages;
  ➢ Enlarging sequential area has a higher priority: switch/partial merge is less expensive.
ADAPT’s Adaptation

• Also adapts threshold for directing a request to sequential or random area:
  - Observation: over a long period, sequential requests tend to access a similar number of pages;
  - In the recent interval, a very small $\delta \Rightarrow$ sequential area was not very effective;
  - ADAPT adjusts the threshold accordingly.
ADAPT’s Predictive Transfer

- FASTer’s second chance scheme
ADAPT’s Predictive Transfer

• FASTer’s *second chance scheme*
  ➢ A page of valid data remains in log space;
  ➢ At least one merge is avoided if the page is accessed soon;
  ➢ If not, such movement would be wasteful.

• Why not predict a page’s update likelihood?
  ➢ Positive: move it;
  ➢ Negative: merge it.

Merge-or-move decision making
ADAPT’s Predictive Transfer

• How to do prediction
  ➢ Temporal locality:
    A recently-updated page is likely to be written to again.

• HAT: historical access table
  ➢ Records a history of recent writes to logical pages;
  ➢ Managed in LRU with fixed space.
ADAPT’s Predictive Transfer

Predictive Transfer

Base Page No. | Size
---|---
9 | 1
5 | 2
... | ...
14 | 2

Historical Access Table (HAT)
ADAPT’s Aggregated Movement

• Observation:
  • Non-OLTP workloads usually have big and sequential write requests;
  • Many log pages in the victim block to be merged are valid;
  • Inefficient to process one by one.

• ADAPT employs aggregated movement to give a second chance to a whole block.
ADAPT’s Aggregated Movement

\( \tau \): aggregated movement threshold;
- Upon a merge, \( L_0 \) and \( L_1 \) are checked;
- If \( L_0 \) has more valid pages than \( \tau \) and \( L_1 \) does not, move \( L_0 \) and merge \( L_1 \).
Outline

• Introduction
  ➢ NAND Flash Memory
  ➢ FTL and workload

• Background of hybrid mapping

• ADAPT
  ➢ Adaptive partitioning of log space
  ➢ Prediction and Aggregation

• Evaluation

• Conclusion
Evaluation

• Simulation setup
  • FlashSim simulator with GCC-4.6;
  • Ten public workloads;
  • DFTL (ASPLOS 2009), FASTer (SNAPI 2010) and WAFTL (MSST 2011) were implemented for comparisons;
  • The main metric is the elapsed time to finish each workload;
  • The default value of $\tau$ is 56; the length of the interval to measure $\delta$ and $\varphi$ is 4000 requests.
Experimental Results

• Elapsed time (performance)

![Bar chart showing normalized elapsed time for different workloads compared to FASTer, with categories DFTL, WAFTL, and ADAPT. The x-axis represents workloads, and the y-axis represents normalized elapsed time. The chart highlights performance metrics for various scenarios.]
Impact of interval length on $\varphi$ and $\delta$ (A)

Experimental Results

- Elapsed Time (unit: second)
- Interval length (unit: request)

Line graph showing the impact of interval length on $\varphi$ and $\delta$ with different data sets.
Experimental Results

- Impact of interval length on $\varphi$ and $\delta$ (B)

Elapsed Time (unit: second)

5500
4500
3500
2500
1500

Interval length (unit: request)

- MSR-mds_0
- MSR-rsch_0
- MSR-stg_0
- MSR-ts_0
- MSR-web_0
### Experimental Results

- **Prediction hit rates and aggregated moves**

<table>
<thead>
<tr>
<th>Workload</th>
<th>Prediction Hit Rate</th>
<th>Aggregated Movements</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPC-C_20</td>
<td>100.00%</td>
<td>0</td>
</tr>
<tr>
<td>SPC1</td>
<td>79.50%</td>
<td>132</td>
</tr>
<tr>
<td>MSR-hm_0</td>
<td>95.68%</td>
<td>233561</td>
</tr>
<tr>
<td>MSR-mds_0</td>
<td>96.49%</td>
<td>1727</td>
</tr>
<tr>
<td>MSR-prn_0</td>
<td>99.93%</td>
<td>124607</td>
</tr>
<tr>
<td>MSR-prxy_0</td>
<td>99.72%</td>
<td>8323</td>
</tr>
<tr>
<td>MSR-rsrch_0</td>
<td>98.75%</td>
<td>2050</td>
</tr>
<tr>
<td>MSR-stg_0</td>
<td>93.24%</td>
<td>1045</td>
</tr>
<tr>
<td>MSR-ts_0</td>
<td>95.16%</td>
<td>1165</td>
</tr>
<tr>
<td>MSR-web_0</td>
<td>96.99%</td>
<td>5408</td>
</tr>
</tbody>
</table>
Outlines

• Introduction
  ➢ NAND Flash Memory
  ➢ FTL and workload
• Background of hybrid mapping
• ADAPT
  ➢ Adaptive partitioning of log space
  ➢ Prediction and Aggregation
• Evaluation
• Conclusion
Conclusion

• ADAPT
  ➢ Fully-associative hybrid mapping scheme
  ➢ Employs
    □ adaptive partitioning of log space
    □ predictive transfer
    □ aggregated movements

• Simulation results show ADAPT can be faster than
  ➢ DFTL by as much as 44.2%
  ➢ WAFTL by as much as 23.5%
Thank you!

Questions?