Using Working Set Reorganization to Manage Storage Systems with Hard and Solid State Disks

Junjie Chen  Jialin Liu  Philip C. Roth  Yong Chen

Data-Intensive Scalable Computing Lab (DISCL)
Texas Tech University
Outline

• Background
• Motivations
• Working Set Model and Data Structure
• WS-ROS Algorithms
• Experimental Results and Analysis
• Conclusion and Future Work
Background

• HPC applications are increasingly data-intensive.
  • Scientific simulations have already reached 100TB – 1PB of data volume, and projected at the scale of 10PB – 100PB for upcoming exascale era.
  • Companies like Facebook manage 100+PB of storage system and 25TB growth per week.

• Such big volume of data brings a critical challenge.
  • Efficient I/O access demands
  • High efficient storage system
Background

- HDD Dominates the Storage Media
  - High capacity, high latency
  - Lower Price, ↑

- Emerging Solid State Drive (SSD)
  - Lower power consumption
  - Less latency, access time
  - Higher price, ↓
Motivating Observations

Looking for a Hybrid Storage System

- Simply Combining HDDs & SSDs under PVFS
  - Only 1.3x ~ 2.5 x speedup compare to HDD
- Why: *Activeness of HDDs and SSDs.*
  - `nmon`: Nigel's performance Monitor for Linux
  - 4 nodes, compute/storage, pvfs
  - 16GB, IOR benchmark
Motivating Observations

• Observation 1
  • The SSD has much less frequent activity than the HDDs for both reads and writes.

• Observation 2
  • The activeness of the SSDs was less than the HDDs when servicing write requests, but the gap was much smaller.
Motivating Analysis

- A performance model for read requests.

- Parameters:
  - # of read requests: $n$
  - Size of request: $s$
  - Read latency of SSD: $l$
  - Read latency of HDD: $\gamma l$

- $T = \max \{ n*s*l/2, n*s*\gamma l/2 \} = n s \gamma l / 2$.

- $T = \max \{ \gamma n * s * l / \gamma + 1, n * s * \gamma l / \gamma + 1 \} = n s \gamma l / \gamma + 1$. 
Our Idea

- We propose a new approach for managing storage systems that combine HDDs and SSDs
  - Working Set-based Reorganization Scheme (WS-ROS)
  - A background process reorganizes the data when devices are idle
- WS-ROS scheduling algorithm
  - Background data reorganization process uses the access history information
Our Contributions

• It characterizes the motivation and requirements for a well-managed storage system that combines HDDs and SSDs.

• It proposes and details Working Set-based Reorganization Scheme to manage a heterogeneous storage system.

• It evaluates the proposed approach using a prototype WS-ROS implementation and simulation with traces taken from real applications.
Working Set Model and Data Structure

- Workloads
  - Write Requests
  - Read Requests
    - I/O History
      - Write Region
      - Read Region

- Storage management
  - Working-Set Model
  - Parallel File System

- Data Reorganization
  - Hot: 1, 3
  - Cold: 2, 4

- Storage Nodes
  - Node 1: HDD, SSD
  - Node 2: SSD, HDD

- WS-ROS

- Intrac: intra
- Inter: inter
• Working Set Model in WS-ROS
  
  Let $d(t)$ denote the location of the data block accessed at a time $t$, and $\tau$ be the width of the WS-ROS sliding window.
  
  $$ws(t_0, \tau) = \{d(t) \mid (t_0 - \tau) \leq t \leq t_0\}.$$  

• Locality Table in Working Set

<table>
<thead>
<tr>
<th>Location</th>
<th>{loc1}</th>
<th>{loc2}</th>
<th>{loc3}</th>
<th>{loc4}</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hotness</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>...</td>
</tr>
<tr>
<td>Timestamp</td>
<td>T1</td>
<td>T2</td>
<td>T3</td>
<td>T4</td>
<td>...</td>
</tr>
</tbody>
</table>
• Read Region in WS-ROS
  • WS-ROS uses a read region to track and manage the workload’s read requests.
  • The read region is used by the WS-ROS to construct the working set model.
  • WS-ROS uses two thresholds to determine whether a data block is considered to be hot, \( \text{threshold-H} \) & \( \text{threshold-L} \).

• Write Region in WS-ROS
  • The WS-ROS write region functions similarly to the read region.
**Algorithm 1: WS-ROS I/O request handling algorithm**

**Input:** I/O requests in trace files.

**for Each request in the input trace do**

1. Get the access records from trace files;
2. **if** Write request **then**
   - **if** file already exists **then**
     - Get stripes from the metadata server through offset;
     - Schedule the request;
     - update the locality table in write region;
   - **else**
     - Create file on storage;
     - Schedule the request with striping;
     - Create an item in the locality table in the write region;
   **end**
3. **else**
   - Get stripes from the metadata server through offset;
   - Retrieve data from storage;
   - Update the locality table in the read region;
4. **end**

**end**
**WS-ROS Algorithms**

```plaintext
Input: Working set data with locality tables.

for Each item in the read region’s locality table do
    if Hotness > Threshold-H then
        Check data block location;
        if Data located on HDD then
            Distribute data from HDD to SSD;
        end
    else
        if Hotness < Threshold-L then
            Delete the item from the locality table;
            if Data located on SSD then
                Distribute data from SSD to HDD through a delayed strategy;
            end
        end
    end
end

for Each item in the write region’s locality table do
    if Hotness > Threshold-H then
        Check data block location;
        if Data located on SSD then
            Distribute data from SSD to HDD;
        end
    else
        if Hotness < Threshold-L then
            Delete the item from the locality table;
        end
    end
end

Algorithm 2: WS-ROS data reorganization algorithm
```
Experimental Results and Analysis

- PVFS file systems, 64 KB stripe size, RR
- Meta-data server: file handle, offset, request size
- 16-node Linux cluster, 8G memory
- RAID 5, 3TB storage
Experimental Results and Analysis

- DB2 Parallel Edition
  - Commercial-grade parallel RDBMS from IBM containing 5.2 GB of data.
  - Five consecutive queries, including join, set, aggregate, etc.

- Parallel Web Server
  - Approximately 1.5 million HTTP requests generated by four clients to multiple Apache servers, resulting in 36 GB of data.
### WS-ROS Simulator Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of SSDs</td>
<td>4</td>
</tr>
<tr>
<td>Number of HDDs</td>
<td>4</td>
</tr>
<tr>
<td>SSD capacity (GB)</td>
<td>64</td>
</tr>
<tr>
<td>HDD capacity (GB)</td>
<td>512</td>
</tr>
<tr>
<td>SSD read latency (s/GB)</td>
<td>0.1</td>
</tr>
<tr>
<td>SSD write latency (s/GB)</td>
<td>0.5</td>
</tr>
<tr>
<td>HDD read latency (s/GB)</td>
<td>1</td>
</tr>
<tr>
<td>HDD write latency (s/GB)</td>
<td>2</td>
</tr>
<tr>
<td>Sliding window size (%)</td>
<td>10</td>
</tr>
<tr>
<td>Stripe size (GB)</td>
<td>64</td>
</tr>
<tr>
<td>Threshold-H</td>
<td>3</td>
</tr>
<tr>
<td>Threshold-L</td>
<td>2</td>
</tr>
<tr>
<td>SSD capacity threshold (GB)</td>
<td>20</td>
</tr>
</tbody>
</table>
Results and Analysis

Performance of Heterogeneous Storage System with WS-ROS (left: DB2 & right: Parallel Web Server)
Results and Analysis

WS-ROS Sensitivity to Hotness (left: DB2 & right: Parallel Web Server)
WS-ROS Sensitivity to Windows Size (left: DB2 & right: Parallel Web Server)
Conclusion and Future Work

• Many scientific and engineering applications run on high-end computing (HEC) platforms consume and/or produce large amounts of data.

• Solid state drives (SSDs) using flash non-volatile memory have emerged as storage devices with complimentary characteristics to HDDs.

• We propose a Working Set-based Reorganization Scheme (WS-ROS) for managing heterogeneous storage systems.

• Our results suggest that heterogeneous storage systems using WS-ROS approach can substantially obtain performance gains.

• In the future, we will conduct finer evaluation of inter & intra communication among the hybrid storage and measure of its performance benefit and overhead in real-world settings.
Thank You

Please visit our website: http://discl.cs.ttu.edu

This research is sponsored in part by the Advanced Scientific Computing Research program, Office of Science, U.S. Department of Energy. This research is also sponsored in part by Texas Tech University startup grant and National Science Foundation under NSF grant CNS-1162488. The work was performed in part at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. De-AC05-00OR22725. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.
Q&A
Backup Slides-Experiment Setup

• **Platform**
  • 16-node linux testbed
  • One PowerEdge R515 rack server node and 15 PowerEdge R415 nodes
  • 32 processors and 128 cores.
  • 6 Crucial Technology RealSSD C300 SSDs with 64GB capacity and 6GB/s data transfer rate

• **Benchmark**
  • DB2 Parallel Edition
  • Parallel Web Server