Object-based Storage Device for High Performance Computing Applications

Biplob Debnath
Sarah Sharafkandi
David Du
David Lilja
HPC Overview

- HPC environment consists of a collection of heterogeneous computing resources connected via heterogeneous network links.

- Initially the source data for all the tasks resides in the root node.

- Other nodes in the system receive tasks from root node and/or some of their neighbors.

- Data Flow Graphs (DFG) provides an abstraction of how to decompose the application into tasks.
Data Flow Graph

Fig: DFG of card3d package [Provides an abstraction of how to decompose a application into tasks]

Figure is taken from “Data Flow Pattern Analysis of Scientific Applications”, Michael Frumkin, Intel, 2005

University of Minnesota  Digital Technology Center  Intelligent Storage Consortium
HPC Overview (cont.)

- Source data for a task can be either computing-intensive or I/O-bound

- Putting more processing power improve the performance of computation intensive part.

- What about I/O intensive data?

- There is significant a gap between server processor performance and disk I/O performance.

- Simply throwing more HW (storage) will not work. It will only mask the problem or move the problem somewhere else.

- More spindles adds hidden cost in the form of management, energy and reliability. More storage need to be managed, configured, tuned, and backed up.

- We will focus on improving the performance of I/O intensive part using data layout awareness of Object-based Storage Device.
Modern Disk Properties

- Tracks are grouped into zones based on their distance from the center of the disk.

- Each zone is assigned a number of sectors per track.

- Capacities and data transfer rates of the outer zones are higher than that of inner ones.

Figure is taken from http://www.pcguide.com/ref/hdd/geom/tracks_ZBR.htm
The difference in bandwidth per second between the outermost zone, which is the fastest, and the innermost zone can be 120% or more.

<table>
<thead>
<tr>
<th>Zone</th>
<th>Tracks in Zone</th>
<th>Sectors per Track</th>
<th>Data Transfer Rate (MB/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>624</td>
<td>792</td>
<td>46.5</td>
</tr>
<tr>
<td>1</td>
<td>1,424</td>
<td>780</td>
<td>45.8</td>
</tr>
<tr>
<td>2</td>
<td>1,680</td>
<td>760</td>
<td>44.6</td>
</tr>
<tr>
<td>3</td>
<td>1,616</td>
<td>740</td>
<td>43.5</td>
</tr>
<tr>
<td>4</td>
<td>2,752</td>
<td>720</td>
<td>42.3</td>
</tr>
<tr>
<td>5</td>
<td>2,880</td>
<td>680</td>
<td>39.9</td>
</tr>
<tr>
<td>6</td>
<td>1,904</td>
<td>660</td>
<td>38.8</td>
</tr>
<tr>
<td>7</td>
<td>2,384</td>
<td>630</td>
<td>37.0</td>
</tr>
<tr>
<td>8</td>
<td>3,328</td>
<td>600</td>
<td>35.2</td>
</tr>
<tr>
<td>9</td>
<td>4,432</td>
<td>540</td>
<td>31.7</td>
</tr>
<tr>
<td>10</td>
<td>4,528</td>
<td>480</td>
<td>28.2</td>
</tr>
<tr>
<td>11</td>
<td>2,192</td>
<td>440</td>
<td>25.8</td>
</tr>
<tr>
<td>12</td>
<td>1,600</td>
<td>420</td>
<td>24.7</td>
</tr>
<tr>
<td>13</td>
<td>1,168</td>
<td>400</td>
<td>23.5</td>
</tr>
<tr>
<td>14</td>
<td>1,815</td>
<td>370</td>
<td>21.7</td>
</tr>
</tbody>
</table>

Data is taken from From Deskstar 40GV and 75GXP Product Manual, © 2000 IBM
Object Based Storage Device

- OSD moves the low level storage functionalities to storage device.
- Provides a simple yet powerful interface to access the data.
- An OSD aware of zoning bandwidth info can improve performance of the applications.
- I/O-intensive data can be placed on the high bandwidth zones.
- Computing-intensive data can be placed in lower bandwidth zones.
Problem Statement

Given
- Data flow graph of a HPC application
- User expected execution time
- Object based storage Device
- Zone information on disks

Find a physical on-disk placement plan for data objects that can achieve the expected execution time for a HPC application.

How to Achieve it?
- Get information regarding computation time and data dependencies.
- Estimate BW requirements of HPC objects.
- Based on BW requirements OSD will place the objects in the proper matching zones.
Is Data Flow Graph Adequate?

- Provides a very higher level abstraction task decomposition of a HPC application.

- What is Missing?
  - Computing time info
  - Fine granular Data dependency info
  - BW required

- How to get the missing data?
  - I/O throttling
  - Similar to //Trace [Fast ’07]

Fig: DFG of card3d package

Figure is taken from “Data Flow Pattern Analysis of Scientific Applications”, Michael Frumkin, Intel, 2005
How to Determine BW?

- Using
  - Data flow graph
  - Size of data objects
  - System configurations
  - Dependency among different nodes processing
  - User expected maximum time

- Estimate the **bandwidths** of the different objects
- Determine the **priorities** of the objects.

- Still needs to be investigated?
How to Place Data Objects on Disk?

- Using **extended QoS attributes** of OSD
  - Higher level application will communicate bandwidth requirements and priorities.

- OSD will try to place the objects in the corresponding bandwidth matching zones.

- If enough space is not available in a zone OSD will use priority for intelligent placement of the data.

- Still needs to be investigated?
OSD vs. Using Outer Zones of Multiple Disks

- **Cost:** doubles as are now using the high-bandwidth zones, we need at least doubles number of disks.

- **Disk Utilization:** disks are now less utilized and as low-bandwidth zones remain unused.

- **Power Consumption:** more powers will be consumed as number disks have increased.

- **Heat:** more power causes more heat.

- **Failure Rate:** usually more heat causes more disk

- **Recovery Times:** rebuild times from failures increases

- **Management Cost:** increases.
OSD vs. File-System Implementation

Traditional File System can maintain

- Metadata of the current allocation
- Utilization of blocks of the different disk zone
- Remap/reallocate data in different zones to match the application bandwidth requirements.
- Issue read and write request in certain pattern to get the required bandwidth, but there is no guarantee of getting it.
  - Disk scheduler can reorder the requests.
  - Multiple FSs share the storage device.

If storage device (OSD) is aware of application requirements

- Can dynamically placing data in high bandwidth zones, buffering, and pre-fetching.

- If underlying disk characteristics change, code needs to be rewritten.

- OSD provides ease of management by removing the overhead of monitoring underlying storage technology change.
Implementation Plan

- **Implementation has Three Main Parts**
  - Estimating Bandwidth Requirement for data objects
  - Extracting zoning information
  - Laying out data on disk based on bandwidth requirements.

- **How to Demonstrate?**
  - Placing data without considering BW and Zone info
    - VS.
  - Placing data on OSD with considering BW and Zone info
Conclusion

- We can use OSD to improve the performance of the HPC applications with some support from the higher layers.

- OSD approach looks promising than Multiple Disks, and current File System alternatives.