Peta-scale Storage for High End Computing

Eisenhauer, Gavrilovska, Schwan, Wolf

Joint work with U. New Mexico
Problem

- Data-intensive HPC applications are increasingly common
- I/O is supported only in a rudimentary way
- Needs for data manipulation are more complex
  - Filtering/conversion
  - Data remeshing
  - Refactoring
  - Associating meta-data and derived data
System Hardware Architecture
Partitioning of MPP Operating System Functionality
Approach

- Introduce a “cloud” of data-driven processing between the MPP and the storage engine
  - “Processing Cloud” is actually a directed graph specifying computation to occur during I/O
  - Initial focus on output I/O, but applicability to input and direct application coupling
IO Graph Goals

- Represent data movement as streams of data
  - Carry out staging, buffering, routing, adornment and morphing of data “in-flight”
  - Allows utilization of any spare cycles on compute nodes (while waiting for I/O), on I/O nodes or on storage nodes
  - Keep type information with data throughout to facilitate processing
  - IO Graph elements are mapped onto processors based upon resource availability
Graph of I/O operations
Implementation Path

- Utilize EVPath
  - High-performance event processing infrastructure derived from earlier ECho work
  - PBIO binary data format with meta-information
  - Dynamic code generation for processing

- LWFS
  - Light-Weight File System from UNM
  - Really a capability-based object-store
LWFS Architecture
Approach (Part 2)

- LWFS supports fast parallel data storage by eliminating higher-level abstractions
  - File abstractions
  - Name mapping
- IO Graphs handle only data manipulation that can be accomplished “in-flight” without introducing significant latency
  - Cannot support many global abstractions
- Introduce “Meta-Bots” for off-line metadata generation
Metabots crawl the Storage Engine

Generator Metabot
- inspect results of structured streams
- generate metadata
- insert metadata into Storage Engine

Reorganizer Metabot
- inspect results of structured streams
- reorganize data in Storage Engine per applications
- insert metadata into Storage Engine
Metabot Locations
SSDS Software Architecture
Summary

- A lightweight application-level specification of I/O requirements
- Run-time creation, optimization, and management of efficient I/O Graphs that implement this specification
- Scalable storage through LWFS, coupled with the ability to interoperate with other storage systems
- Off-line metadata enhancement of data in the storage engine through metabots
Status

- **Year 1 tasks**
  - EVPath/Catamount integration
  - Design and implement APIs for IOGraphs and metabots
  - Create device-spanning metadata representations
  - Finalize infrastructure design WRT filesystem virtualization
  - Augment LWFS with naming support