Tiering and Life Cycle Management with AI/ML Workloads

Jacob Farmer
• CTO, Cambridge Computer
• Chief Evangelist, Starfish Storage

MSST - 2019
Thesis Statement:

Machine learning creates demand for new classes of storage and thus provides impetus for the adoption of better practices for data life cycle management.
When are Tiering and Lifecycle Management Most Interesting?

- When there is a wide spread in costs between the fastest tiers and the cheapest tiers.
- When the data has a life cycle or usage pattern that allows for meaningful savings when files can be moved between tiers.
  - When the cost and complexity of data life cycle management does not outweigh the savings of putting all data on a single tier.
Suitable performance is a MUST HAVE
- Compute resources are expensive and sit idle when the storage system fails to bring the data

There are tons of new solutions
- Super fast, massively scalable flash storage systems
- Specialized software interfaces for bypassing the kernel to load GPUs
- In memory compute using capacity optimized RAM disks

How much top tier do you need?
- How do you measure?

How do you avoid wasting top tier capacity?
- Stale data should not sit in expensive storage
Secondary Tiers

- The main purpose of the secondary tiers is to swap files with the primary tier
  - They need to be optimized for suitable data transfer performance

- This is different from typical HPC life cycle management
  - In conventional HPC many workloads are happy on a middle level tier
  - Many conventional workloads can take the latency hit reading the file from a lower tier while promoting the bit to a higher tier.
Most large datasets in science fall into the WORSE or WORN category:

- WORN – Write Once Read Never
- WORSE – Write Once Read Seldom if Ever

Machine learning sets are much more likely to be reused and when they are reused, very large data sets need to be retrieved from archive.
In short:

Machine learning workloads require more aggressive staging and de-staging between tiers than traditional scientific computing workloads.
There are Many Solutions for Federating Tiers of Storage

- **File systems with multiple tiers**
  - Newer file systems leverage SSDs and cloud tiers natively
  - Many file systems can subsume external storage devices and incorporate them into their name space.

- **PNFS is back and can provide a global namespace across multiple devices**

- **You can simply have multiple storage devices (even local staging disk), each with their own namespace and move files yourself.**
  - Logical namespace in middleware
  - Logical namespace in application software
Fundamental Questions

- Where is my file?
- Where do I want it to be instead?
- Is it there yet?
The OOO Model For Data Life Cycle Management

- **Organization**
  - Namespace
  - Metadata
  - Content Index

- **Orchestration**
  - Policy Engine
  - User-driven
  - API

- **Operation**
  - Copy - Move - Delete
  - Other Data Processes

- Storage Options:
  - File Systems
  - Object Stores
  - Containers
  - Tape Archives
  - Cloud Storage
A unified POSIX-like namespace is perhaps less important because Machine Learning is driven by the machine.

- The machine does not need a friendly pathname in a hierarchical structure

The workflow will likely be driven by metadata stored in an application
Orchestration – The Tricky Part

- You can’t rely on file touches to trigger migration.
  - You have to be able to stage and de-stage in advance

- What are you going to do? Chances are you will do the following (because this is what everyone pretty much does)
  - Make a database of your files
  - Add metadata to your database to make it easy to specify which files you want
  - Query the database to generate a migration script
  - Run the script on a scheduled basis or integrate with job scheduler
Starfish (*FS)

A Software Company Spun Out of Cambridge Computer
What Does Starfish Do?

- Starfish makes and maintains a database of your file system
- Starfish allows you to associate metadata with files and directories
  - Gather metadata from file system, individual files, from the workflow, or wherever.
  - Copy select metadata from application software
- Starfish uses the query result to feed batch processor that executes code against the files
  - Batch processing runs in parallel across multiple agents
  - Agents are ordinary LINUX machines (Windows agent later this year).
    - You can borrow nodes from the compute farm
- Of course, all of this is API driven, easy to use, and feature rich
What Makes Our Database Implementation Special

- It is open. We use PostGres.
- We handle extreme scale
  - Billions of files
  - Thousands of change events per second
- File and Directory Metadata
  - Simple tags on files and directories (inheritable or not)
  - Key-value pairs for individual files
- We keep the version history of the directory tree and of individual files
- We aggregate values for lightning fast insights
- We take action on the query results
  - COPY, MOVE, DELETE, GET, PUT, etc.
Starfish is Made Up of Three Main Components

- **Core database**
  - We synchronize the metadata in your POSIX file systems with a database.
  - We allow additional metadata to be added to files and directories

- **Jobs engine**
  - A batch processor that takes the results of the query and does “stuff”.
    - Copy, move, delete
    - Calculate hashes
    - Extract metadata
    - Your code or ours
  - Work is divvied up among any number of agents

- **User Interface**
  - HTML-5 file system browser
  - Discovery and system monitoring
  - (Beta) User portal that allows users to participate in storage management policies
A “Virtual” Global File System