

## Introduction

Modern scientific computing generates petabytes of data and billions of files that users must manage. Files are then organized into hierarchical file systems, by name, into a tree of directories that must be manually navigated. Tools such as Google Desktop and Apple Spotlight have allowed for organization of files on metadata such as file owner, file type, file size, etc. This work looks to improve on these tools in the following way:

- ▶ Minimize need for custom user interfaces.
- ▶ Provide a known command line interface with the search tool.
- ▶ Expand file queries beyond metadata attributes.

A tool that is common to many scientists is the POSIX API. This API is not optimized to work with the amounts of data at the scale that exists at LANL. This poster presents the foundational work for two ideas:

1. **Access Method Support File System (AMS-FS):** A file system that provides querying of file system information integrated with the well known and time tested POSIX API.
2. **Incremental Growth Index (IGI):** An indexing structure designed to operate within the very large environments that are present here at LANL and at many facilities around the world.

The final goal of AMS-FS is to alleviate problems facing many scientists. Files are often organized using esoteric naming conventions that must be navigated in order to find relevant information. These conventions can generate large amounts of work if they happen to change. With AMS-FS, instead of needing to follow a complicated path to get to experimental data, this information can be obtained using a simple 'ls' command.

## Current Approaches

This work aims to examine three problems with current file organization in scientific computing:

1. **Metadata Search:** Current tools require custom user interfaces that do not always meet the needs of the user.
2. **File Content Search:** Many file types used in scientific computing are structured in such a way that information in the file can be queried. In order to query these files more custom interfaces must be designed.
3. **Multiple Output Files:** Current computational methods such as parallel computing and MapReduce often produce multiple output files for a single task. Once again, custom programs are needed to extract relevant information.

AMS-FS looks to take these three issues and combine them into a single processing paradigm, the POSIX API (ls, mv, cp, etc.). This is a well know set of tools that most researchers will be familiar with. AMS-FS is implemented as a FUSE module [1].

## AMS-FS Architecture Overview

When a query enters AMS-FS, its execution goes through a two step process:

1. **Query Tree Construction:** A tree based on the given query is constructed for evaluation.
2. **Query Evaluation:** Using the query tree, files and their information are then read from disk and only those files satisfying the query are reported in the end result.

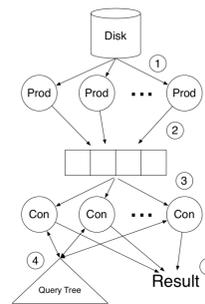


Figure: AMS-FS Parallel Query Evaluation Architecture

Query evaluation follows a five step process:

1. Producers read file information from the disk
2. The information is buffered in memory
3. Consumers read this information from the buffer
4. The file information is processed in the query tree
5. If the file satisfies the query, it is reported in the final result

## Incremental Growth Index (IGI)

Important in many I/O bound applications is the presence of an indexing structure. An indexing structure aims to reduce reads from the disk by pruning the data space that one must access. Here we present a new indexing structure, IGI, based on the Incomplete Pyramid Structure [2].

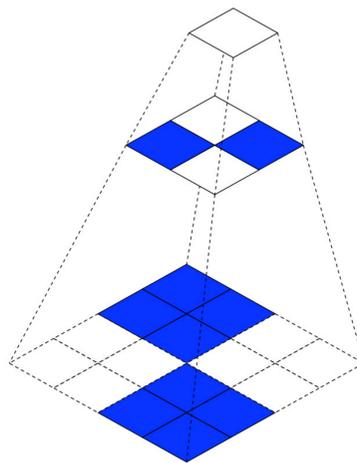


Figure: The Incomplete Pyramid Structure

## Incremental Growth Index (IGI) cont.

Due to the large amount of data stored at LANL, any structure aiming to index the data must be able to construct itself quickly. Current research [3] does not cover the problem of fast index construction at the petabyte scale. To accomplish this, IGI begins as a small and lightweight, but coarse grained index. As query load increases to various areas of the index IGI is able to dynamically expand to become more fine grained, and thus, prune more of the search space.

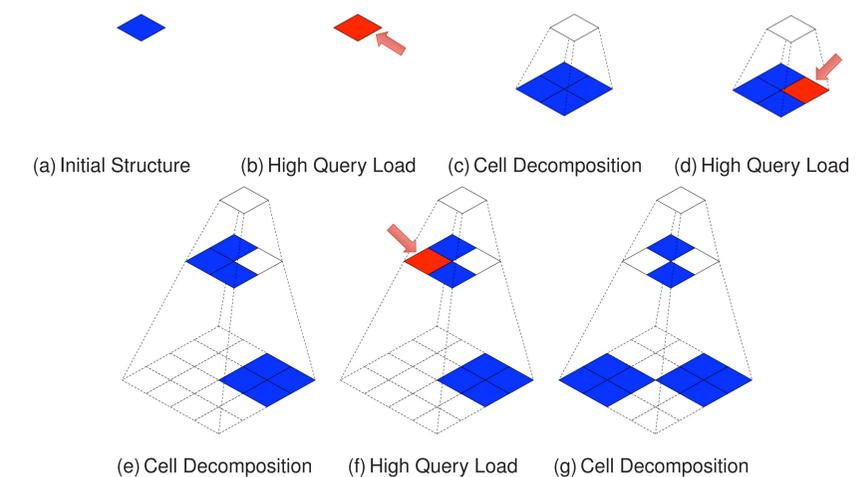


Figure: IGI Execution Example

## Future Work

Future work on AMS-FS fits into three categories:

1. **Large Scale Testing:** AMS-FS must be tested, evaluated, and modified to work efficiently on the large scale systems present at LANL.
2. **Index Development:** The Incremental Growth Index must be fully developed.
3. **Expansion Of Operations:** The current architecture only will operate on metadata queries. Functionality must be added to allow for file content queries over single and multiple files.

## Selected References

- [1] <http://fuse.sourceforge.net/>
- [2] W. G. Aref and H. Samet. Efficient Processing of Window Queries in The Pyramid Data Structure. In *PODS*, 1990.
- [3] A. W. Leung, M. Shao, T. Bisson, S. Pasupathy, E. L. Miller. Spyglass: Fast, Scalable Metadata Search for Large-Scale Storage Systems. In *USENIX* 2009.