Tutorial

Modeling and Visualizing Big and Complex Data
A Tutorial on HDF5 and an Extended Example Based on F5

Werner Benger (CCT/LSU),
Albert Cheng, Neil Fortner, Elena Pourmal (The HDF Group)

Overview

Linux clusters have become an important tool in doing today's science and made it possible to study very complex phenomena in a wide range of scientific fields. Data modeling, efficient data management, and data visualization have become a critical part of the scientific discovery process and present new challenges to the users.

While multiple cores create gigabytes of data, and parallel file systems in combination with the parallel I/O libraries, such as MPI-IO, provide scalable and manageable I/O, having computational power alone is not enough to deal with data complexity, extensibility, and portability. Scientists need a way to describe complex data and relationships between data components that often extend beyond one application and one computational platform.

This tutorial introduces two software packages HDF5 (Hierarchical Data Format 5) and F5 (The Fiber Bundle HDF5 library), a thin library layer on top of HDF5 for storing, managing and visualizing big and complex data.

Created more than a decade ago at the National Center for Supercomputing Applications at University of Illinois and currently maintained by the nonprofit company "The HDF Group", HDF5 addresses data complexity and data management needs in today's high-performance computational environments including Linux clusters. It provides a simple yet powerful data model along with the portable and scalable access to data.

Inspired by the mathematical concept of Fiber Bundles, F5 was created by Dr. Werner Benger at the Albert Einstein Institute and further developed at the Center for Computation and Technology at Louisiana State University. It offers a Common Data Model implemented using HDF5 and provides a simple Application Programming Interface (API) to formulate a wide range of data types used for scientific visualization, thereby achieving interoperability among applications and to make possible the modeling and visualization of diverse physical phenomena. HDF5 and F5 were successfully used to model hurricane Katrina, collisions of galaxies, and rotations of neutron stars and black holes, to name a few.

The full-day Tutorial will be equally useful to anyone who wants to learn HDF5 or to master HDF5 to go from data to vision and beyond.

HDF5

HDF5 is designed to store, access, manage, exchange, and archive diverse and complex data. It supports any type of data suitable for digital storage, regardless of its origin or size. For example, petabytes of remote sensing data received from satellites, terabytes of computational results from weather models, and gigabytes of high-resolution MRI brain scans are stored in HDF5 files along with additional information necessary for efficient data exchange, data processing, visualization, and archiving.
It has a rich and sophisticated set of features for optimizing storage space and access time, including compression, chunking, metadata caching, and an extensible set of I/O drivers.

In recent years, the number of applications that successfully use HDF5 in fields other than physical science and engineering has increased. Sony Pictures Imageworks, the award-winning visual effects and digital character animation unit of Sony Pictures Digital Productions, has launched an open source development program, which includes the Field3D, a voxel data storage library, based on HDF5**.** Field3D was used in the production of visual effects for *Spider-Man 2* and *I Am Legend*. Many applications in the field of bioinformatics use HDF5 to manage an avalanche of DNA sequencing data. More applications use HDF5 as a container for heterogeneous data, for example, for storing audio and video streams along with the analysis data and visualization results. One of the more unorthodox examples is an application in the field of Behavioral Neurobiology that uses HDF5 to study animal vocal behavior.

The robustness of HDF5, and the availability of open source and commercial tools for analysis and visualization of data stored in the HDF5 format, has made HDF5 an attractive choice for data format standards used within companies and government organizations aimed at reducing data management costs. In June 2008, NASA endorsed HDF5 as a data standard for Earth Science Data Systems.

HDF5 runs on a variety of platforms from Windows desktops to high-performance Linux clusters. The HDF5 library comes with C, C++, Fortran, and Java programming interfaces. It is developed and supported by The HDF Group, a non-profit corporation with a mission to ensure the long-term accessibility of HDF data (www.hdfgroup.org).

F5

HDF5 by itself only provides the syntax to store data, but not the semantics where to store them, and under what naming scheme. This may well lead to disappointed expectations, as users of HDF5 expect ``magic" to happen at the ``semantic" level just because they are using HDF5 at the ``syntax" level.

The F5 library implements such a semantic layer on top of HDF5, specifying a concrete data layout, which is able to cover a wide range of types of scientific data. Its design is inspired by the mathematical theory of fiber bundles, but the user does not need to be burdened with the math behind the model. The F5 library stores data utilizing the hierarchical group structure of HDF5 and organizes a dataset in five levels. Each of these levels has a particular meaning:

1. Time
2. Geometrical entity
3. Topological property
4. Coordinate system
5. Field name

A file converter using the F5 data layout will have to determine where exactly to place a particular data set using the scheme. The F5 library provides support for doing so via an easy user-friendly API. The resulting files can be inspected later on a syntax level using HDF5 tools, such as ``h5ls" and ``h5dump", and on the semantic level using F5 tools such as ``F5ls".

Tutorial Topics

1. Introduction to HDF5

The tutorial will explain the HDF5 data model and show how applications can take advantage of the model to represent their data structures. The data model discussion will include an overview of HDF5 abstractions such as datasets, groups, attributes, and datatypes. Simple C examples will cover the programming model, basic features of the API, and will give new users the knowledge they need to navigate through the rich collection of HDF5 interfaces. Audience members will be guided through an interactive demonstration of the fundamentals of HDF5. Audience members will use the graphical tool HDFView to view, modify, and create new HDF5 files, use simple features of some command line tools, and compile and run a basic program that uses HDF5.

2. F5

In this part of the Tutorial participants will be shown how data from a scientific application can be organized in HDF5 for further analysis and visualization. We will use the F5 library as a layer on top of HDF5, and explore the basic applications that are provided. This includes converters from existing file formats, as well as example applications that create an F5 file from a simple numerical simulation. The HDF5 tools will be used to inspect the contents of the file. The F5 library uses HDF5 group, HDF5 dimensional datasets, HDF5 symbolic links, HDF5 named datatypes, and HDF5 attributes. The use and purpose of these HDF5 features in the context of F5 will be discussed. Finally, we will load these datasets into a visualization application and display them graphically in 3D.

3. Performance tuning for HDF5

To achieve good performance with HDF5, applications developers need to understand HDF5's advanced optimization features including partial I/O, chunking, compression, and metadata cache management. It is important to use these features appropriately to

achieve good performance and efficient storage**.** A substantial amount of time will be spent on detailing the benefits of different chunking strategies and the workings of the chunk cache, in recognition of their critical importance to developers of high-performance applications. The tutorial will explain how HDF5 handles application data, and discuss how to use HDF5's performance tuning capabilities to improve sequential I/O, to handle large numbers of objects in HDF5 files, and to match data layouts to application access patterns.
Audience members will then be given a sample program using HDF5, which shows poor performance, and will be asked to investigate the reasons for the poor performance. Audience members will then modify the program parameters to improve performance without changing its functionality.

4. Parallel HDF5

This part of the tutorial is designed for users who have had exposure to MPI I/O and would like to learn about the parallel HDF5 library. It will cover parallel HDF5 design and programming models and API functions. C and Fortran examples will be used to demonstrate the capabilities of the HDF5 parallel library. The tutorial will discuss the performance of the parallel HDF5 library and its tuning capabilities to improve parallel I/O. The h5perf tool, which comes with the parallel HDF5 library, will be used to compare the performance of parallel HDF5, MPI I/O and POSIX I/O for different access patterns and storage layouts. HDF5 parallel applications developers can use the tool to evaluate the performance of each layer on their HPC systems and tune their applications.