HDF5 Metadata

Introduction

HDF5 files can contain several types of metadata:

- Library metadata
- Static user metadata
- Dynamic user metadata

Library Metadata

Library metadata is metadata that the user does not have any direct interaction with or control over. It is generated by the HDF5 Library to describe the structure of the file and structure and contents of objects in the file. For example, library metadata includes information such as:

- Most elements of the header block (superblock), which sets up the file, sets up the initial structures, and identifies the file as a valid HDF5 file
- Object headers, which set up objects in an HDF5 file
- B-trees that describe the location of and provide access to groups and members of groups

HDF5 natively interprets and understands library metadata. Library metadata is always present; even an otherwise-empty file must contain a superblock and a root group object header to be a valid HDF5 file.

Static and Dynamic User Metadata

User metadata is defined and provided by the user application.

HDF5 does not always natively understand user metadata; much of it must be understood and interpreted by the application. For example, the only thing the library understands in the dynamic user metadata list below is the extent of the dataset in the last bullet.

User metadata is technically optional but is almost universally used.

Static User Metadata

Static user metadata is information that the user can control but that is not generally dynamic. It is stored in the file superblock or an object header and does not usually change through the life of a file or object.

Examples of static user metadata include:

- Property lists: For example, `H5Pset_fapl_family` sets a file access property specifying that file I/O will use the family driver.
- Link names
- A dataset's datatype and dataspace (modulo the potential to extend or shrink it)
- Dataset fill values
- Dataset or group storage properties

Static user metadata does not usually change through the life of a file or object. In some cases, it can change but just doesn't tend to to do so; for example, the name of a hard link to an object can be changed only by creating a new hard link and removing the old link. In the more general case, static user metadata can be changed only when making a new copy of an HDF5 file or object. For example, file creation and dataset creation properties can be changed only when making a new copy of a file or dataset, respectively.

Dynamic User Metadata

Dynamic user metadata is metadata that the user or application can change at will. It is often stored in an HDF5 attribute, may describe virtually anything, and can easily change over time.

The following are instances of dynamic user metadata:

- Minimum and maximum valid values in a dataset
- Conditions under which data was collected
- Data history and/or provenance
- Relationships among datasets
- Scales or other interpretive information
- The extent of a chunked dataset within the bounds of its maximum extent

Metadata Types and Mechanisms

Table 1 lists several examples of each type of HDF5 metadata, where it is stored, how it is set, and whether it is natively interpreted by the HDF5 Library or must be interpreted by the user application. This is a representative subset, not a complete list of HDF5 metadata.

Table 1. Examples of HDF5 metadata by type

Element	Where stored	How set	Interpreted by
Library metadata			
Superblock	Header block at beginning of file	Created with file; always present	HDF5 Library
File driver information	Superblock and driver information block	`H5Pset_fapl_*`	HDF5 Library
B-trees	At various locations within file	Library	HDF5 Library
Object offsets	B-tree	Library	HDF5 Library
Object headers	Header block for each object in an HDF5 file	Created with object; present as long as object exists	HDF5 Library
Static user metadata			
Dataset storage layout	Dataset object header	`H5Pset_layout`	HDF5 Library
Shared object header messages	Superblock and global heap	`H5Pset_shared_mesg_*`	HDF5 Library
Link names and hierarchical structure	Group symbol table entries	H5G, H5L interfaces	HDF5 Library
Permanent property lists	Dataset object header, data layout message,	H5P interface	HDF5 Library
Transient property lists	Not stored	H5P interface	HDF5 Library
Checksum	Dataset object header plus a checksum value accompanying each compressed dataset chunk	`H5Pset_fletcher32`	HDF5 Library and application
Datatype	Dataset object header	H5T interface	HDF5 Library
Dataspace (contiguous dimensions or chunked maximum dimensions)	Dataset object header	H5S interface	HDF5 Library
Dynamic user metadata			
Min/max dataset values	Attribute(s)	H5A interface	Application
Data collection conditions	Attribute(s)	H5A interface	Application
Data provenance	Attribute(s)	H5A interface	Application
Object relationships (other than hierarchical structure)	Attributes	H5A interface	Application
Measurement scales	Attribute(s)	H5A interface	Application
Dataspace (current chunked dimensions)	Dataset object header	`H5Dset_extent`	HDF5 Library and application

Additional Information

See the following texts for additional details, usage information, and examples.

Properties

HDF5 property lists are used for both static and dynamic user metadata. Object creation property lists are static; they are stored with the object and cannot be changed without rewriting that object. Object access property lists are dynamic, or transient. They must be defined when an object is first created and redefined every time the object is opened; they are not stored.

Both static and dynamic user properties associated with the following classes of objects are discussed in specific chapters of the HDF5 User's Guide:

Object	Chapter
File properties	"The HDF5 File"
Group properties	"HDF5 Groups"
Dataset properties	"HDF5 Datasets"

The "H5P: Property List Interface" section in the HDF5 Reference Manual lists and describes the usage details for all of the interfaces used to manage the above types of properties and several additional types.

Attributes

HDF5 attributes, which offer nearly infinite flexibility for dynamic user metadata, are discussed in:

- The "HDF5 Attributes" chapter of the HDF5 User's Guide
- The "H5A: Attribute Interface" section in the HDF5 Reference Manual .

Library Metadata

Aside from the hierarchical structure of a file, library metadata is generally opaque to the user.

If a file's structure is unknown, it can be determined through functions described in the "H5L: Link Interface," "H5O: Object Interface," and "H5G: Group Interface" sections of the HDF5 Reference Manual .

Managing Metadata

Metadata Cache

Significant performance gains can be achieved in certain circumstances by directly managing metadata I/O. This occurs most frequently in a high-performance computing (HPC) environment or when working with large data and complex access patterns. Managing the metadata cache can be a complex undertaking and should not be undertaken without careful study.

These issues are discussed in detail in "Metadata Caching in HDF5," a document in the collection Advanced Topics in HDF5 .

The HDF5 functions used to manage metadata caching are described in the "H5F: File Interface" and "H5P: Property List Interface" sections of the HDF5 Reference Manual . Look for functions with names containing the string `'_mdc'`.

Metadata Journaling

Some HDF5 applications can run for a very long time, sometimes for several days or even weeks. In such cases, an unexpected failure can cause the loss of all computed results that have not yet been written to storage. HDF5 provides the ability to periodically flush raw data to storage to guard against complete loss, but preserving metadata in the case of such a failure has been more problematic. HDF5 will introduce metadata journaling in HDF5 Release 1.10 to address this issue, making it possible to reconstruct metadata in the event of such an event.

In release 1.10, metadata journaling functions will be described in the HDF5 Reference Manual and a detailed discussion of their use is expected to be included in the collection Advanced Topics in HDF5.