Investigation report for Allowing NetCDF-4 to Access HDF-EOS5 Files

Kent Yang, Choonghwan Lee November 30, 2008
Revised on January 27, 2009

---

1. Introduction

In the past year, we received a request from NASA Aura team to help support the use of netCDF-4 APIs to access HDF-EOS5 grid files. According to Aura File Format Guidelines (1), only geographic projection is used to generate NASA Aura HDF-EOS5 grid files. In this report, we present three possible solutions that allow netCDF-4 APIs to access HDF-EOS5 geographic projection grid files. We also provide a recommended solution based on pros and cons of these solutions.

2. Background

HDF5 (2) is a file format and I/O library for manipulating scientific and other data. HDF5's support for user-defined metadata and hierarchical organization of datasets within a single file make it well-suited for organizing collections of related scientific data.

The HDF5 library defines two primary types of objects: *groups* and *datasets*. An *HDF5 Group* is a structure containing zero or more HDF5 objects and an *HDF5 Dataset* is a multidimensional array of data elements. Both *HDF5 Datasets* and *HDF5 Groups* can have *HDF5 Attributes*. An *HDF5 Attribute* is a small meta object describing the nature and/or usage of HDF5 objects.

An *HDF5 Dimension Scale* is a special *HDF5 Dataset* that is associated with a dimension of another *HDF5 Dataset* to provide temporal or spatial information such as time, latitude and longitude. The HDF5 library also defines another type of objects, *HDF5 Link*, which is analogous to links in UNIX file system.

Both HDF-EOS5 (3) and netCDF-4 (4) are built on HDF5. The HDF-EOS5 library defines a few data types – grid, swath, point and zonal average – specialized for the earth observing system. This document only covers the grid data type using geographic projection. The HDF-EOS5 library also defines *groups*, *fields*, *attributes* and *dimensions*. An instance of the grid data type consists of one group, multiple fields, multiple attributes, and multiple dimensions. An *HDF-EOS5 Group*, an *HDF-EOS5 Field* and an *HDF-EOS5 Attribute* are represented using an *HDF5 Group*, an *HDF5 Dataset* and an *HDF5 Attribute*, respectively. However, an *HDF-EOS5 Dimension* is not implemented by following the *HDF5 Dimension Scale model* (5). Figure 1 shows the file structure of a typical HDF-EOS5 file viewed as an HDF5 file. In this figure, `CloudFraction` and `CloudPressure` are *HDF5 Datasets* corresponding to two *HDF-EOS5 Fields*. `CloudFractionAndPressure` is an *HDF5 Group* corresponding to an *HDF-EOS5 Group*. Other *HDF5 Groups* including `HDFEOS`, `GRIDS` and `Data Fields`, are groups created by the HDF-EOS5 library but invisible by HDF-EOS5 applications.

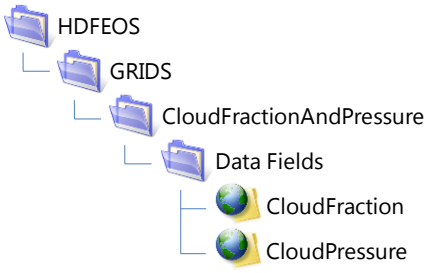

Figure 1 File structure of an HDF-EOS5 file viewed as an HDF5 file

Figure 2 shows an HDF-EOS5 grid using geographic projection. Network Common Data Form Language (CDL)-like descriptions of the grid structure are also inserted in the figure. The example data field shown in the figure is Ozone. It is a 8-by-14, two-dimensional array represented by red dots. All data points along each horizontal line (parallel to XDim) share the same latitude, and all data points along each vertical line (parallel to YDim) share the same longitude. Therefore, a one-dimensional array with 14 elements can be used to describe the longitude of this 2-D data field. A one-dimensional array with 8 elements can be used to describe the latitude of this 2-D data field.

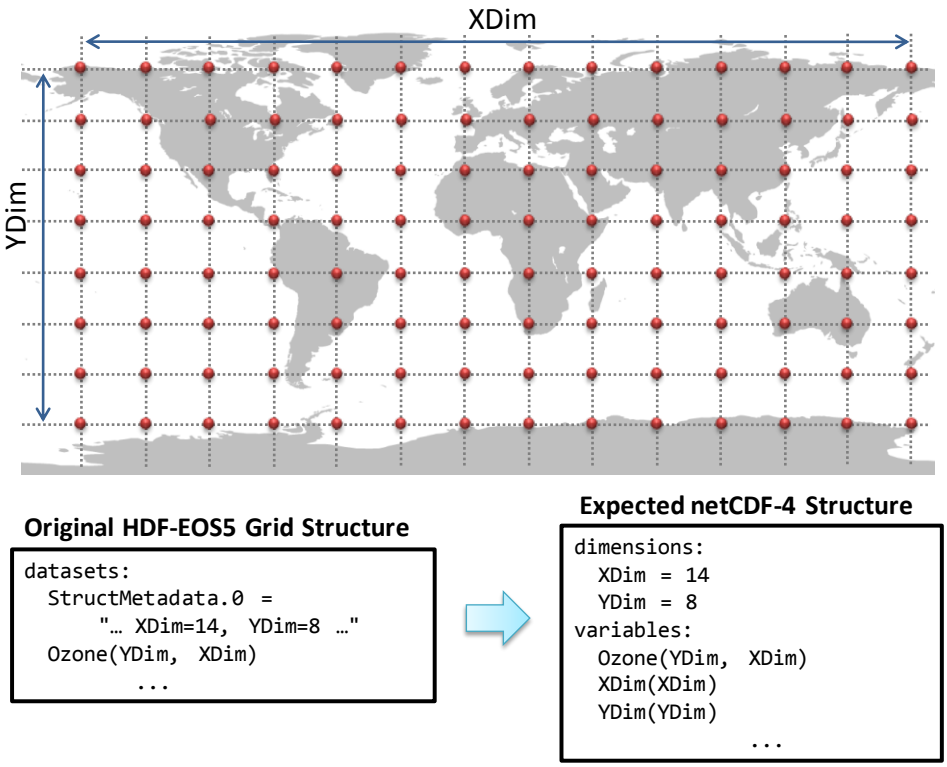

Figure 2 The geographic projection and the expected netCDF-4 structure

The netCDF-4 library defines *groups*, *variables*, *attributes* and *dimensions*. A *netCDF-4 Group*, a *netCDF-4 Variable* and a *netCDF-4 Attribute* are implemented using an *HDF5 Group*, an *HDF5 Dataset* and an *HDF5 Attribute, respectively.* A *netCDF-4 Dimension* is implemented as an *HDF5 Dimension Scale* by following HDF5 Dimension Scale model. The netCDF-4 library requires that every dimension of any single *netCDF-4* variable is associated with a *netCDF-4 Dimension.*

3. The issue

Although two *HDF-EOS5 Dimensions* are actually associated with both *HDF-EOS5 Fields*, they are invisible in Figure 1 because they are not *HDF5 Dimension Scales*.

The grid data type is unique in that geolocation field (latitude, longitude) values are not directly stored as data arrays. Instead, by calling corresponding HDF-EOS5 APIs, these values can be calculated from several parameters contained in one *HDF5 Dataset* predefined by the HDF-EOS5 library. For other dimensions such as Time, however, values may not be retrieved through HDF-EOS5 APIs.

The current netCDF-4 library cannot access an HDF-EOS5 file mainly because the HDF-EOS5 file fails to provide the dimension information in the format netCDF-4 requires. In the following sections, we will provide three solutions. Some discussions are then involved and one of the three solutions is recommended to be implemented.

4. Solutions

All solutions consist of three parts: augmenting an HDF-EOS5 file, modifying the netCDF-4 library, and modifying the HDF-EOS5 library. Augmenting an HDF-EOS5 file means that the *HDF5 Dimension Scales* should be added inside the file so that netCDF-4 APIs can access those data. We will show how each augmenting method changes the HDF-EOS5 file shown in Figure 1.

A. Solution 1

The first solution stores *HDF5 Dimension Scales* under the original `HDFEOS` group. It requires the following augmentation of an HDF-EOS5 file.

- Create an *HDF5 Dimension Scale* for each *HDF-EOS5 Dimension* (latitude, time etc.), and associate it with corresponding *HDF5 Datasets*
- Calculate longitude and latitude values using the corresponding HDF-EOS5 API
- Fill two special *HDF5 Dimension Scales*, `XDim` and `YDim`, with longitude and latitude values

The result of the augmentation is illustrated in Figure 3. The arrows imply that `XDim` and `YDim` are *HDF5 Dimension Scales* associated with *HDF5 Datasets*: `CloudFraction` and `CloudPressure`.

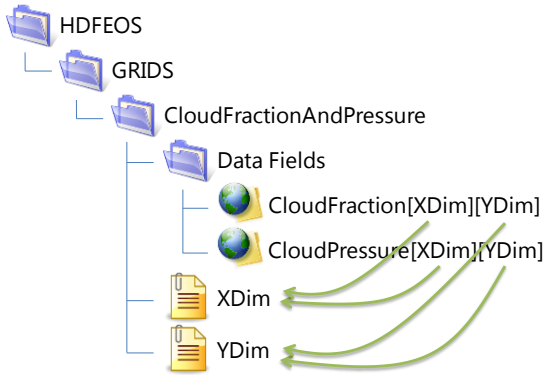

Figure 3 File structure of the augmented HDF-EOS5 file viewed as an HDF5 file according to the Solution 1

The netCDF-4 library also needs modification.

- Relax the requirement of having creation order when creating HDF5 objects

The current netCDF-4 library requires that the HDF5 objects it reads must be created with creation order, which is a new feature provided by HDF5 1.8 and later releases. However, the existing HDF-EOS5 files are not created by following creation order and these files are not desirable to be changed to follow creation order because it may break backward compatibility with tools and applications that could only read HDF-EOS5 files generated with HDF5 1.6 or earlier releases.

To solve this problem, the netCDF-4 library should be modified to allow the access of HDF5 objects without having the creation order. This may violate the netCDF programming model, but we think the requirement can be relaxed at least for read-only mode.

The HDF-EOS5 library also needs one change.

- Ignore attributes dedicated to *HDF5 Dimension Scales*

When an *HDF5 Dimension Scale* is associated with an *HDF5 Dataset*, an attribute called `DIMENSION_LIST` is attached to the *HDF5 Dataset*. The data type of this attribute is an HDF5 compound type that the HDF-EOS5 library fails to handle. We contacted the HDF-EOS5 developer regarding this issue and we concluded that the best way to avoid the failure caused by the HDF-EOS5 library is to ignore the return of this attribute when users request the list of all *HDF-EOS5 Attributes*. Since `DIMENSION_LIST` is required implicitly when using the netCDF-4 library to access *HDF5 Datasets*, HDF-EOS5 applications should not be aware of this attribute at all. So this change will be appropriate.

B. Solution 2

In this solution, the grid data accessed by the netCDF-4 applications will be ***logically*** separated from the original HDF-EOS5 grid structure so that there is a clear distinction between netCDF-4 data and HDF-EOS5 data from point of view of applications.

The HDF-EOS5 file needs to be revised as follows:

- Create an *HDF5 Group* called `NETCDF4` and copy every descendent groups of `HDFEOS` group under `NETCDF4` group
- Create an *HDF5 Link* under `NETCDF4` group for each *HDF-EOS5 Field*
- Create an *HDF5 Dimension Scale* under `NETCDF4` group for each *HDF-EOS5 Dimension* and associate it with corresponding *HDF5 Links*
- Calculate longitude and latitude values using the corresponding HDF-EOS5 API
- Fill two special *HDF5 Dimension Scales*, `XDim` and `YDim`, with longitude and latitude values

The result of the augmentation is illustrated in Figure 4. The dashed arrow indicates that an *HDF5 Link* points to another object.

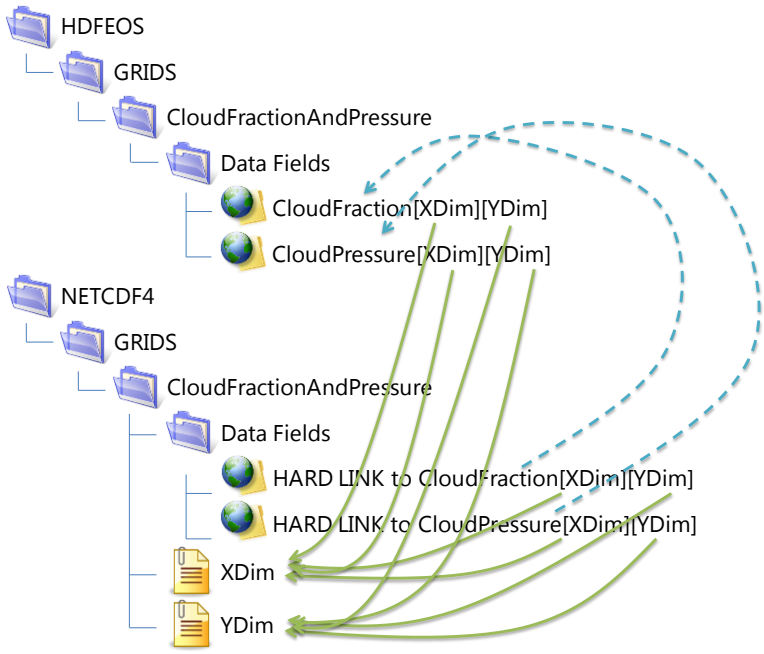

Figure 4 File structure of the augmented HDF-EOS5 file viewed as an HDF5 file according to the Solution 2

Compared with Solution 1, in addition to creating new *HDF5 Groups*, both `XDim` and `YDim` are located in the created group as shown in Figure 4.

The netCDF-4 library needs two modifications.

- Relax the requirement of having creation order when creating HDF5 objects
- Allow a *netCDF-4 Dimension* outside of variable's ancestor groups to be associated with a *netCDF-4 Variable*

The first modification is exactly the same as the modification mentioned in the Solution 1. The second modification tries to solve a problem caused by another netCDF-4 requirement. If a *netCDF-4 Dimension* is associated with a *netCDF-4 Variable*, it should be located in one of ancestor groups of the variable. Two *netCDF-4 Variables*: `CloudFraction` and `CloudPressure` in `HDFEOS` group do not meet this

requirement as Figure 4 shows. Relaxing this requirement may need a major revision of the netCDF-4 library.

Similar to the Solution 1, the HDF-EOS5 library requires one modification.

- Ignore attributes dedicated to *HDF5 Dimension Scales*

These modifications are exactly the same as the modifications made for the Solution 1. Ignoring the special `DIMENSION_LIST` attribute is still necessary because an attribute of an *HDF5 Link* is also an attribute of an H*DF5 Dataset* pointed by the *HDF5 Link*.

C. Solution 3

In this solution, the grid data accessed by the netCDF-4 applications are ***physically*** and ***logically*** separated from the original HDF-EOS5 grid structure.

The last solution clones *HDF-EOS5 Fields*.

- Create an *HDF5 Group* called `NETCDF4` and copy every descendent group of `HDFEOS` group under `NETCDF4` group
- Create an equivalent *HDF5 Dataset* under `NETCDF4` group for each *HDF-EOS5 Field*
- Create an *HDF5 Dimension Scale* for each *HDF-EOS5 Dimension* under `NETCDF4` group and associate it with created *HDF5 Datasets*
- Calculate longitude and latitude values using the corresponding HDF-EOS5 API
- Fill two special *HDF5 Dimension Scales*, `XDim` and `YDim`, with longitude and latitude values

Figure 5 shows the file structure following this solution.

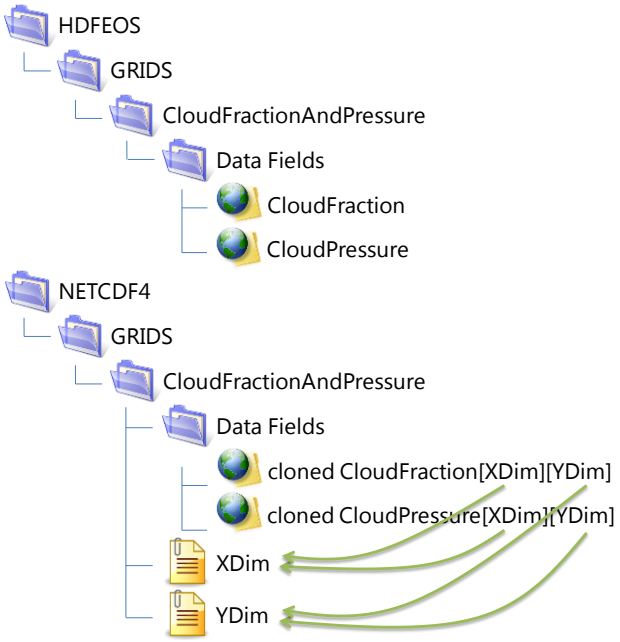

Figure 5 File structure of the augmented HDF-EOS5 file viewed as an HDF5 file according to the Solution 3

For this solution, two modifications are required for the netCDF-4 library.

- Relax the requirement of having creation order when creating HDF5 objects
- Create temporary dimensions on-the-fly

The first modification is exactly the same as the modification addressed in Solution 1 and 2. The second one is needed because two *HDF-EOS5 Fields* under `HDFEOS` group don't have associated *HDF5 Dimension Scales*, which doesn't meet one of netCDF-4 requirements. By modifying the netCDF-4 library, the temporary dimensions for these fields can be created while a file is being opened. Although the temporary dimensions don't have real dimensional data, netCDF-4 can successfully access the *HDF-EOS5 Fields*.

Compared to the Solution 2, the Solution 3 does not introduce any *HDF5 Links*. Since all objects under `HDFEOS` group are preserved and the HDF-EOS5 library is unaware of `NETCDF4` group and all its descendants, the current HDF-EOS5 library can access this file without any errors.

5. Discussion

Table 1 summarizes three solutions based on programming complexity.

Table 1 Summary of three solutions based on programming complexity

	Solution 1	Solution 2	Solution 3
Augmentation of HDF-EOS5 file	Simple	Complex	Complex

HDF-EOS5 modification	Ignore `DIMENSION_LIST` attribute	Ignore `DIMENSION_LIST` attribute	None
NetCDF-4 modification	Relax the creation order requirement	1. Relax the creation order requirement 2. Allow dimensions outside of variable's ancestor groups	1. Relax the creation order requirement 2. Create temporary dimensions
Programming Cost	Small	Big	A little bigger than solution 1

Table 2 summarizes three solutions based on space overhead and I/O time.

Table 2 Summary of three solutions based on space overhead and I/O time during augmentation

	Solution 1	Solution 2	Solution 3
File size after augmentations	Generally small	Generally small	Almost doubled
I/O time during augmentations	Relatively short	Relatively short	Relatively long

Overall, we believe that the Solution 1 is the best. Augmenting the HDF-EOS5 file is easy to implement, and the structure of the HDF-EOS5 file does not need to be changed much. Also, the augmented file size is generally small.

One may be concerned about modifications of two widely-used libraries because those modifications can cause compatibility issues. While investigating the issue of modifying the HDF-EOS5 library, we contacted the HDF-EOS5 developer. He confirmed that our approach won't cause any problems and he is willing to make the change in the next release if there is a request.

We also contacted the netCDF-4 developer regarding the modification of the netCDF-4 library. He confirmed that relaxing the requirement of the creation order and probably other modifications related to the support of accessing HDF-EOS5 grid via the netCDF-4 APIs can be done.

6. Limitations

This investigation covers only HDF-EOS5 grid data using the geographic projection. The HDF-EOS5 library supports a few types including grid, swath, point and zonal average. HDF-EOS5 library supports various projections for grid data. This investigation only addresses issues of accessing HDF-EOS5 geographic projection grid data via netCDF-4.

This investigation assumes that no HDF5 object names contain characters disallowed by the netCDF-4 library. The netCDF-4 library allows alphabets, digits and only a few symbols in object names, but

the HDF-EOS5 library allows more symbols. For this reason, a legal HDF-EOS5 object name can be rejected by the netCDF-4 library. This investigation assumes that all object names in HDF-EOS5 files are acceptable for the netCDF-4 library.

7. Acknowledgements

This work was supported by the Cooperative Agreement with the National Aeronautics and Space Administration (NASA) under NASA grant NNX06AC83A and NNX08A077A. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NASA.

We thank Dr. Abe Taaheri from Raytheon, Ed Hartnett from Unidata, and Cheryl Craig from National Center for Atmospheric Research for their support in making this investigation possible. We also thank Dr. Mike Folk from the HDF Group for providing valuable suggestions.

8. References

1. Craig, et al. *A File Format for Satellite Atmospheric Chemistry Data Based On Aura File Format Guidelines.* 2008.
2. *HDF5.* [Online] The HDF Group. http://www.hdfgroup.org/HDF5/.
3. HDF-EOS5. *HDF-EOS.* [Online] http://hdfeos.org/software.php#HDF-EOS5.
4. *NetCDF-4.* [Online] Unidata. http://www.unidata.ucar.edu/software/netcdf/netcdf-4/.
5. HDF5 Dimension Scale Specification and Design Notes. [Online] March 1, 2005. http://www.hdfgroup.org/HDF5/doc/HL/H5DS_Spec.pdf.