H5edit Atomicity Feature

(Rev: Oct 30, 2012 by Albert Cheng)

1 Purpose

This describes the requirements and design of the Atomicity feature of the H5edit tool. Section 2 shows a user case of the need of the atomicity feature. Section 3 lists the requirements of the feature. Section 4 describes the implementation design. Section 5 shows the validation requirements and implementation.

2 A User Story of the Need of Atomicity

When a user uses the H5edit tool to modify an HDF5 data file, the file is opened and modifications are made according to the edit commands specified by the user. There are two kinds of potential failures.

2.1 User command errors

The user specifies some incorrect edit commands. For example, he CREATE a new attribute and DELETE another attribute of a dataset successfully. He then attempts to CREATE a new attribute that exists already. The H5edit tool will issue an error and exit the tool with a failure status code. At that point, the original data file is already partially changed. The user may want to have his edit commands be done all or none, not partially. Unless he has made a backup copy of the data file before starting the H5edit tool, he cannot undo the partial changes. For example, he cannot recover the original attribute that has been deleted.

2.2 System failures

Sometimes computer resources fail. E.g., a disk runs out of space. If this happens while the H5edit tool is running, the tool has to exit and leaves behind an unstable data file that is not accessible by the HDF5 library any more. If the user has not made a backup copy of the original data file, all is lost.

3 Requirement Specifications

In this section, I will describe the functionality of the Atomicity option of the H5edit tool that will provide a means to restore the original data files in the failures described before.

3.1 Function Definition

The H5edit tool should create and maintain a backup copy of the original data file being edited by the tool. The Atomicity option controls the manner the backup copy is managed. In case of user commands errors or system failures, the data file can be recovered from the backup copy.

3.2 Atomicity Option

Commands are applied to the data file in an atomic manner, that is, the file will not be modified only partially if H5edit encounters error during the execution.
The Atomicity option will first make a backup copy of the data file before applying the input command. If the tool encounters any error, the user may recover the data file from the backup copy.
The Atomicity option, --atomic, supports 3 levels, *yes*, *no*, and *inc*.

- --atomic=yes applies the input commands in an all or none manner. That is, either all input commands are applied to the data file, or none is applied.
- --atomic=no will not make the backup copy and apply the input commands as much as possible. The user may use this if he is sure of the validity of the input commands or he is not concern if the data file is partially modified or corrupted.
- --atomic=inc (default) will apply the changes in an all or none manner but at a command level. If H5edit encounters an error during execution, the data file can be restored back to the last command applied successfully.

4 Implementation Design

4.1 Backup File Name Convention

A backup copy of the data file shall have a file name that is composed of the file name of the data file but preceded with a period and appended with ".bck". For example, if the data file name is "2010_10_01_data.h5", the backup file name will be ".2010_10_01_data.h5.bck".

4.2 When and How the Backup File is Generated and Removed

The backup file will be generated from the original data file before any change is applied to the data file. Upon the successful execution of the H5edit tool, the backup file will be deleted before the tool exits. If there is any error encounter during the H5edit tool execution, the backup file will not be deleted.

4.3 Implementation stage

The complete implementation of the Atomicity option will be delivered in three stages.

4.3.1 Stage 1: Minimum backup support

The backup file is generated right after the data file is opened successfully. No more modification is applied to the backup file during the execution of the H5Edit tool. Upon the successful execution of the H5edit tool, the backup file is deleted before the tool exits. If there is any error during the H5edit tool execution, the backup file is not deleted.
In another word, the *inc* level is not implemented yet and is the same as the *yes* level.

4.3.2 Stage 2: Support for Incremental Atomicity

In addition to the functionality of Stage 1, --atomicity=*inc* will be supported in the following manner:
During the execution of the H5edit tool, whenever an edit command is completed successfully, H5edit issues an H5Fflush if the atomicity level is *inc*. The H5Fflush will trigger the update of the backup file by copying the data file to the backup file before H5edit will execute the next edit command.

4.3.3 Stage 3: Performance Enhancement

In addition to the functionality of Stage 2, the update of the backup file is enhanced in the following manner:
The data file is divided into segments or regions of 1 MB each. Changes made to the data file are recorded according to segments touched. When the backup file should be updated, only segments that have been touched are copied from the data file to the backup file. After the update, the record of segments touched is reset for the next round of changes.

4.4 Implementation time estimate

The total time is estimated to be 90 hours with breakdown as the following:

4.4.1 Stage 1

For design, implementation, testing and documents: 40 hours

4.4.2 Stage 2

For design, implementation, testing and documents: 25 hours

4.4.3 Stage 3

For design, implementation, testing and documents: 25 hours

5 Validation Requirements and Implementation

This section will be completed after the stakeholders approve the function specification described above.