RFC: h5diff – Exclude Object(s) from Comparison

Jonathan Kim ([jkm@hdfgroup.org](mailto:jkm@hdfgroup.org))

The h5diff command-line utility compares objects in HDF5 files and reports differences. Currently, h5diff does either pairwise comparison of all objects in the files or comparison of two particular objects.

This RFC proposes adding an option to h5diff that allows the user to exclude object(s) from the pairwise comparison.

# Introduction

The h5diff[[1]](#footnote-1) command-line utility has ability to compare entire HDF5 files or specific objects (groups, datasets, named datatypes, symbolic links).

When comparing HDF5 files, h5diff compares objects with matching paths in the two files. This is referred to as pairwise comparison.

This RFC proposes a new option that will cause h5diff to exclude specified object(s) from the pairwise comparison when comparing HDF5 files.

# Motivation

h5diff is often run as part of a scripted batch process that relies on h5diff’s exit code to determine the results of the comparison, rather than on a visual scan of the text output. In some circumstances users may want to compare most of the objects in two HDF5 files, excluding a small number of objects that they know exist in only one file, or that they know to be different.

Two examples demonstrate when excluding an object from comparison and basing the exit code and text output on the comparison of the non-excluded objects would be useful.

## Example 1

File1.h5 has these objects: /g1, /g1/d1, /g1/d2, /d1, /g2, /g2/s1, /g2/d1

File2.h5 has these objects: /g1, /g1/d1, /g1/d2, /d1, /g2, /g2/s1

The command “h5diff File1.h5 File2.h5” will return an exit code of “1”, indicating the two files are different, because File2.h5 does not have the object /g2/d1. However, the output from h5diff, when run without the –v option, will not show any differences.

The user may be aware of the extra object (/g2/d1) in File1.h5 and want to exclude it from comparison, having h5diff compare the remaining objects in the files, and returning an exit code based only on those objects.

## Example 2

File3.h5 has these objects: /G1, /G1/D1, /G1/D2, /D1, /G2, /G2/S1, /G2/D1, /D\_timestamp

File4.h5 has these objects: /G1, /G1/D1, /G1/D2, /D1, /G2, /G2/S1, /G2/D1, /D\_timestamp

The user may not care if the D\_timestamp objects are different, but may want to compare all other objects in the files and return an exit code based on the comparison of those objects.

## Current Approach

Currently, there is no easy way to exclude object(s) from comparison.

One approach would be to run h5diff multiple times, compare a specific pair of objects with each run, and report any differences.

This approach could be implemented as follows:

1. Generate a list of absolute paths for all objects in each file.
2. Remove duplicate entries from the list.
3. Remove the path(s) of the object(s) that should not be compared.
4. Use a script to run “h5diff *file1 file2 object”* multiple times, once for each object that remains in the pruned object list.

While this approach is possible, it becomes unwieldy as the number of objects in the file increases. Furthermore, executing h5diff multiple times introduces a performance penalty.

Another approach is to run “h5diff –v *file1 file2* > diffout*”*, then use a series of post-processing commands (grep, sed, awk, etc.) to filter out the known differences from the “diffout” text file. This approach can be quite slow, and is problematic when the size of “diffout” is large.

# Proposed Solution

This RFC proposes a new option that will cause h5diff to exclude specified object(s) from the pairwise comparison when comparing HDF5 files. This solution will allow the user in the examples presented above to perform the desired comparisons without the added effort required by the current approach.

## –exclude-object Option

We propose calling the new option –*exclude-object,* with the following usage:

h5diff ***–exclude-object “****exclude\_object” file1 file2*

The argument following *--exclude-object,* denoted by “*exclude\_object”* in the sample command line, specifies the object that will be excluded from the pairwise comparison. The excluded object can be a group, dataset, named datatype, or symbolic link (soft link or external link), and must be expressed as an absolute path from the root group.

With the exclude option, all occurrences of the specified object are excluded from comparison, regardless of whether the object occurs in file1, in file2, or in both files.

If the excluded object is a group, the group and all objects in the hierarchy below the group are excluded from the pairwise comparison.

The exit code and output from h5diff will be based on pairwise comparison of the objects in file1 and file2 that have not been excluded.

## Excluding Multiple Objects

If multiple objects are to be excluded, the *--exclude-object* option must be repeated for each excluded object:

h5diff  ***--exclude-object*** “/g1/d2”   ***--exclude-object*** “/g2/d1”  … *file1 file2*

While repeating the *–exclude-object* option may seem cumbersome, it will simplify the construction of command lines for automated scripting. Furthermore, since HDF5 allows all characters to be used when naming an object (there are no reserved or special characters), separating absolute pathnames (objects) specified in a single quoted string would not be straightforward.

# Use Cases

## Case 1: Excluding an extra object

Consider the motivating Example 1 in Section 2.1, where

File1.h5 has these objects: /g1, /g1/d1, /g1/d2, /d1, /g2, /g2/s1, /g2/d1

File2.h5 has these objects: /g1, /g1/d1, /g1/d2, /d1, /g2, /g2/s1

To exclude the extra object /g2/d1 that only appears in File1.h5 from comparison, the command would be:

>> h5diff –exclude-object “/g2/d1” File1.h5 File2.h5

The h5diff exit code will be 0 if the pairwise comparison of other objects in the file found no differences, and 1 if some differences were found in those objects.

## Case2: Excluding an object that exists in both files

Consider the motivating Example 2 in Section 2.2, where

File3.h5 has these objects: /G1, /G1/D1, /G1/D2, /D1, /G2, /G2/S1, /G2/D1, /D\_timestamp

File4.h5 has these objects: /G1, /G1/D1, /G1/D2, /D1, /G2, /G2/S1, /G2/D1, /D\_timestamp

To exclude the object /D2\_timestamp (that is expected to be different) from comparison, the command would be:

>> h5diff –exclude-object “/D\_timestamp” File3.h5 File4.h5

The h5diff exit code will be 0 if the pairwise comparison of other objects in the file found no differences, and 1 if some differences were found in those objects.

## Case3: Excluding objects that exist in only one file; excluding group objects

A user has two HDF5 files, with the objects shown:

File5.h5 has these objects: /gg1, /gg1/dd1, /gg1/dd2, /dd1

File6.h5 has these objects: /dd1, /gg2, /gg2/ss1, /gg2/dd1

Based on the path names, and the fact that they have objects under them, we can infer that /gg1 and /gg2 are group objects.

The h5diff command

>> h5diff –exclude-object “/gg1” –exclude-object “/gg2” File5.h5 File6.h5

excludes the group object /gg1 (and all objects under it) and group object /gg2 (and all objects under it) from comparison. /gg1 only exists in File5.h5 and /gg2 only exists in File6.h5.

This command causes h5diff to do a comparison on the /dd1 objects in File5.h5 and File6.h5. If the /dd1 objects are the same, the h5diff exit code will be 0. If differences are found, it will be 1.

This command, with the example files shown, is equivalent to:

>> h5diff File5.h5 File6.h5 “/dd1”

# Additional Features in consideration (in progress)

As these features are in discussion, feedback for preferred argument name(s) or approach would be welcome.

## Use a file as input method

Allow user to specify <input file> which contains objects or attributes.

The <input file> can have single or multiple lines that represent various targets to be excluded. Use the same argument name as identifier for each target line, which will provide convenient way to switch between command line and <config file> by copy&paste the target list.

Example :

**--exclude-object** “/g1/g2/d2”

**--exclude-object** “/g1/g2/d4”

**--exclude-object-attribute** “/g1/g2/d8” “speed high”

…

**--exclude-unknown-future** “…” ….

…

This will be especially helpful when the object paths are too long to fit on one command line or reuse the input file as template.

We are in discussion for the final argument name.

Three candidate names are listed below:

* --**control-file-exclude** “<input file>”
* --**command-file-exclude** “<input file>”
* --**config-file-exclude** “<input file>”

## Excluding attribute(s)

### Feature

There are 4 ways to exclude attribute(s). THG tool team is in discussion for final argument names and approach.

* Case1: Exclude one specified attribute from one specified object
* Case2: Exclude all attributes from one specified object
* Case3: Exclude one specified attribute from all object
* Case4: Exclude all attributes from all objects

All of this feature may not be initially implemented; however it’s necessary to discuss at this point to prevent interface confliction from future updates.

### Considerations for user interface

For the user interface, two approaches are presented below:

#### Consideration 1

Use single argument name along with wildcard ‘\*’.

* Case 1: Exclude one specified attribute from one specified object
  + **--exclude-object-attribute** “<full path to an object>” “<attr name>”
* Case 2: Exclude all attributes from one specified object
  + **--exclude-object-attribute** “<full path to an object>” “**\***”
* Case 3: Exclude one specified attribute from all object
  + **--exclude-object-attribute**  “**\***” “<attr name>”
* Case 4: Exclude all attributes from all objects
  + **--exclude-object-attribute** “**\***” “**\***”

The wildcard ‘\*’ represent entire name of object or attribute, so it will not represent single character in object or attribute name.

Restriction: For this feature, user should not name any attribute with single ‘\*’ character, because it will be treated as wildcard.

Also wildcard can be used for representing all objects in a certain group.

For example:

* --**exclude-object-attribute** “/*g1*/\*” “<attr name>”
* --**exclude-object-attribute**  “/*g1*/\*” “\*”

Restriction: For this feature, user should not name any object with single ‘\*’ character, because it will be treated as wildcard.

An asterisk (\*) will always be interpreted as a wildcard in the context of h5diff options that understand wildcards (e.g., --exclude-object-attribute); in such circumstances, h5diff will not be able to individually access objects or attributes named \* (i.e., named by the single asterisk). Objects and attributes named \* will, however, be picked up by the wildcard along with all other eligible objects and attributes elements.

#### Consideration 2

Provide unique argument names for each case.

Below argument names are current candidates.

* Case1: Exclude one specified attribute from one specified object
  + --**exclude-object-attribute** “<full path to an object>” “<attr name>”
* Case2: Exclude all attributes from one specified object
  + --**exclude-object-all-attributes** “<full path to an object>”
* Case3:  Exclude one specified attribute from all object
  + --**exclude-attribute** “<attr name>”
* Case4: Exclude all attributes from all objects
  + --**exclude-all-attributes**

# Additional Features for Future Consideration

These feature requests came up in the process of discussing the RFC. They will not be implemented in the initial release, but are recorded for future reference and possible implementation at a later date.

* ‘*—exclude-all-paths object’*: If the specified object is accessible via multiple paths (has multiple hard links to it), exclude all of those paths from comparison. If this option is implemented, it will also be necessary to clarify how soft links to the object will be treated (i.e., will they also be excluded), and if the treatment of soft links to the object will depend on whether or not --follow-symlinks option is specified.
* Support multiple paths with one invocation of the *–exclude-object* option. Before this feature can be added, a strategy for reserving and/or escaping special characters in object paths must be developed.
* Allow wild cards in object path names. Before this feature can be added, a strategy for reserving and/or escaping special characters in object paths must be developed.
* Add an option, such as “--common-objects-only”, to specify that h5diff should only perform pairwise comparisons on objects that are common to both files. With this option, a user could more easily indicate that they don’t want the occurrence of an object in one file but not the other to trigger a “files are different” condition.

In particular, the Use Case 1 command:

>> h5diff --exclude-object “/g2/d1” File1.h5 File2.h5

could be replaced by:

>> h5diff --common-objects-only File1.h5 File2.h5

And, the Use Case 3 command:

>> h5diff --exclude-object “/gg1” --exclude-object “/gg2” File5.h5 File6.h5

could be replaced by:

>> h5diff --common-objects-only File5.h5 File6.h5

For users who frequently want to compare only objects that are common to both files, this option would allow them to do so easily, and would not require that the enumerate (or even know) the objects that are in only one of the two files.

If the implementation of the --exclude-object option makes the --common-objects-only option very straightforward, it may be included in the initial release.

# Development procedure

For the implementation procedure, excluding object in file to file comparison via command line will be implemented as initial step. After that, input file method can be added along with the feature.

For excluding attribute features, case1 (Exclude one specified attribute from one specified object) from chapter 5.2.1 will be considered as main feature to be implemented and the rest of features (case2 to 4 from chapter 5.2.1) will be considered as optional.

# Acknowledgements

This work was prompted by a request from Mike Linda (SSAI - NPP SDS Ozone PEATE), who frequently wants to use batch scripts to compare some, but not all, objects in HDF5 files.

Ruth Aydt (aydt@hdfgroup.org) provided editorial support.

# Revision History

|  |  |
| --- | --- |
| *May 13, 2010:* | Version 1 circulated for comment within The HDF Group. |
| *May 17, 2010:* | Version 1.2 updated from the internal meeting and feedbacks within the HDF Group. |
| May 24, 2010: | Version 1.3 changed option from --ignore-path to --exclude to be more descriptive and consistent with existing h5diff reference manual page.  Removed discussion of hard and soft links, which were orthogonal to the main purpose of the RFC. Made other editorial changes.  Expanded Additional Features discussion. |
| June 10, 2010: | Version 4: Incorporated feedback from Mike Linda in Motivation, Current Approach, and Acknowledgements sections.  Changed Version numbering to be consistent with other RFCs (no version 1.x, just version x). |
| *July 9,2010:* | Version 5 incorporated internal discussions within the HDF Group.  Change –exclude to –exclude-object.  Add chapter 5 and Development procedure sections. |

1. Refer to the h5diff reference manual entry for a more extensive discussion of h5diff’s behavior. [↑](#footnote-ref-1)