RFC: Supporting soft-link and external-link for h5diff

Jonathan Kim (jkm@hdfgroup.org)

This RFC is for discussing about a new feature for supporting soft-link and external-link when ‘follow link’ command option is given for h5diff as our Chicago customer requested for the feature in h5diff command line tool.

1)

Description

Currently, h5diff command only compares the name (path) of link value not the actual target object when dealing with link(s).

With the new feature, h5diff will go through the link(s) and figure out the object at the end of the link and compare its data value and attribute if exist.

Internally there are 3 kinds of links, which are hard-link, soft-link, and external-link (as part of user defined-link). The new feature will provide transparent experience to a user when comparing linked objects as the user can specify any of those links to be compared in any combination.

Test results without following link option as example

Default behavior compares type of link itself (soft link, external link as part of user-defined link) and line’s value name. So if type of specified links is same, h5diff compare the links’ value. If all matches, it’s treated as same.

Test cases were created to demonstrate current h5diff behavior. There are 3 HDF5 data files were used. One file is for soft-link tests and the other two files are for external-link tests. The output from h5dump for each file is shown below.

- Soft-link test file contents:

```
HDF5 "soft_link.h5" {
GROUP "/" {
  SOFTLINK "softlink_dset1_1" {
    LINKTARGET "target_dset1"
  }
  SOFTLINK "softlink_dset1_2" {
    LINKTARGET "target_dset1"
  }
  SOFTLINK "softlink_dset2" {
    LINKTARGET "target_dset2"
  }
  SOFTLINK "softlink_group1" {
    LINKTARGET "target_group"
  }
  SOFTLINK "softlink_group2" {
    LINKTARGET "target_group"
  }
  SOFTLINK "softlink_noexist" {
    LINKTARGET "no_obj"
  }
  DATASET "target_dset1" {
    DATATYPE  H5T_STD_I32LE
    DATASPACE  SIMPLE { ( 2, 4 ) / ( 2, 4 ) }
    DATA {
    (0,0): 0, 1, 2, 3,
    (1,0): 1, 2, 3, 4
    }
  }
  DATASET "target_dset2" {
    DATATYPE  H5T_STD_I32LE
    DATASPACE  SIMPLE { ( 2, 4 ) / ( 2, 4 ) }
    DATA {
    (0,0): 0, 0, 0, 0,
    (1,0): 0, 0, 0, 0
    }
  }
  GROUP "target_group" {
    DATASET "dset" {
      DATATYPE  H5T_STD_I32LE
      DATASPACE  SIMPLE { ( 2, 4 ) / ( 2, 4 ) }
      DATA {
      (0,0): 0, 1, 2, 3,
      (1,0): 1, 2, 3, 4
      }
    }
  }
}
}
```

- External-link source file contents:

```
HDF5 "extlink_source.h5" {
GROUP "/" {
  EXTERNAL_LINK "ext_link_dset1" {
    TARGETFILE "extlink_target.h5"
    TARGETPATH "/target_group/x_dset"
      DATASET "/target_group/x_dset" {
        DATATYPE  H5T_STD_I32LE
        DATASPACE  SIMPLE { ( 2, 4 ) / ( 2, 4 ) }
        DATA {
        (0,0): 0, 1, 2, 3,
        (1,0): 1, 2, 3, 4
        }
      }
  }
  EXTERNAL_LINK "ext_link_dset2" {
    TARGETFILE "extlink_target.h5"
    TARGETPATH "/target_group2/x_dset"
      DATASET "/target_group2/x_dset" {
        DATATYPE  H5T_STD_I32LE
        DATASPACE  SIMPLE { ( 2, 4 ) / ( 2, 4 ) }
        DATA {
        (0,0): 0, 0, 0, 0,
        (1,0): 0, 0, 0, 0
        }
      }
  }
  EXTERNAL_LINK "ext_link_grp1" {
    TARGETFILE "extlink_target.h5"
    TARGETPATH "target_group"
      GROUP "target_group" {
        DATASET "x_dset" {
          HARDLINK "/target_group/x_dset"
        }
      }
  }
  EXTERNAL_LINK "ext_link_grp2" {
    TARGETFILE "extlink_target.h5"
    TARGETPATH "target_group2"
      GROUP "target_group2" {
        DATASET "x_dset" {
          HARDLINK "/target_group2/x_dset"
        }
      }
  }
  EXTERNAL_LINK "ext_link_noexist1" {
    TARGETFILE "extlink_target.h5"
    TARGETPATH "no_obj"
  }
  EXTERNAL_LINK "ext_link_noexist2" {
    TARGETFILE "no_file.h5"
    TARGETPATH "no_obj"
  }
}
}
```

- External-link target file contents:

```
HDF5 "extlink_target.h5" {
GROUP "/" {
  DATASET "target_dset1" {
    DATATYPE  H5T_STD_I32LE
    DATASPACE  SIMPLE { ( 2, 4 ) / ( 2, 4 ) }
    DATA {
    (0,0): 0, 1, 2, 3,
    (1,0): 1, 2, 3, 4
    }
  }
  GROUP "target_group" {
    DATASET "x_dset" {
      DATATYPE  H5T_STD_I32LE
      DATASPACE  SIMPLE { ( 2, 4 ) / ( 2, 4 ) }
      DATA {
      (0,0): 0, 1, 2, 3,
      (1,0): 1, 2, 3, 4
      }
    }
  }
  GROUP "target_group2" {
    DATASET "x_dset" {
      DATATYPE  H5T_STD_I32LE
      DATASPACE  SIMPLE { ( 2, 4 ) / ( 2, 4 ) }
      DATA {
      (0,0): 0, 0, 0, 0,
      (1,0): 0, 0, 0, 0
      }
    }
  }
}
}
```

Test results

- Soft-link test results:

```
==================================================================
# Case: group (soft linked) vs group (target)
CMD> h5diff -v soft_link.h5 soft_link.h5 /softlink_group1 /target_group

</softlink_group1> is of type H5G_LINK and </target_group> is of type H5G_GROUP
--------------------------------
Some objects are not comparable
--------------------------------
Use -c for a list of objects.


==================================================================
# Case: dset (soft linked) vs dset (target)
CMD> h5diff -v soft_link.h5 soft_link.h5 /softlink_dset1_1 /target_dset1

</softlink_dset1_1> is of type H5G_LINK and </target_dset1> is of type H5G_DATASET
--------------------------------
Some objects are not comparable
```

```
--------------------------------
Use -c for a list of objects.


===============================================================
# Case: dset (soft linked) vs non-exist object
CMD> h5diff -v soft_link.h5 soft_link.h5 /softlink_dset1_1 /softlink_noexist

link   : </softlink_dset1_1> and </softlink_noexist>
1 differences found


===============================================================
# Case: group (soft linked and exist) vs group (soft linked and exist)
CMD> h5diff -v soft_link.h5 soft_link.h5 /softlink_group1 /softlink_group2

link   : </softlink_group1> and </softlink_group2>
0 differences found


===============================================================
# Case: dset (soft linked and exist) vs dset (soft linked and exist) - same values
CMD> h5diff -v soft_link.h5 soft_link.h5 /softlink_dset1_1 /softlink_dset1_2

link   : </softlink_dset1_1> and </softlink_dset1_2>
0 differences found


===============================================================
# Case: dset (soft linked and exist) vs dset (soft linked and exist) - diff values
CMD> h5diff -v soft_link.h5 soft_link.h5 /softlink_dset1_1 /softlink_dset2

link   : </softlink_dset1_1> and </softlink_dset2>
1 differences found
```

- External-link test results

```
===============================================================
# Case group (ext linked in src file) vs group (in target file)
CMD> h5diff -v extlink_source.h5 extlink_target.h5 /ext_link_grp1 /target_group

</ext_link_grp1> is of type H5G_UDLINK and </target_group> is of type H5G_GROUP
--------------------------------
Some objects are not comparable
--------------------------------
Use -c for a list of objects.


===============================================================
# Case dset (ext linked in src file) vs dset (in target file)
CMD> h5diff -v extlink_source.h5 extlink_target.h5 /ext_link_dset1 /target_group/x_dset

</ext_link_dset1> is of type H5G_UDLINK and </target_group/x_dset> is of type H5G_DATASET
```

```
--------------------------------
Some objects are not comparable
--------------------------------
Use -c for a list of objects.


================================================================
# Case  dset (ext linked) vs object (non-exist / exsiting file)
CMD> h5diff -v extlink_source.h5 extlink_target.h5 /ext_link_dset1 /ext_link_noexist1

Object </ext_link_noexist1> could not be found in <extlink_target.h5>


================================================================
# Case  dset (ext linked) vs object (non-exist / no file)
CMD> h5diff -v extlink_source.h5 extlink_target.h5 /ext_link_dset1 /ext_link_noexist2

Object </ext_link_noexist2> could not be found in <extlink_target.h5>


================================================================
# Case  group (ext linked and exist) vs group (ext linked and exist)
CMD> h5diff -v extlink_source.h5 extlink_source.h5 /ext_link_grp1 /ext_link_grp2

external link: </ext_link_grp1> and </ext_link_grp2>
1 differences found


================================================================
# Case  dset (ext linked and exist) vs dset (ext linked and exist)
CMD> h5diff -v extlink_source.h5 extlink_source.h5 /ext_link_dset1 /ext_link_dset2

external link: </ext_link_dset1> and </ext_link_dset2>
1 differences found
```

Decision from suggestions for following link behavior

We decided to go with the 'suggestion 2' at the discussion with the RFC1.0.

For the reference, all the previous suggestions recorded here.

Two simple cases of using h5diff

- For comparing the two files : h5diff   <file1> <file2>
- For comparing the two object :   h5diff   <file1> <file2> <obj1> <job2>

Suggestion 1

Act as the current h5diff behavior, which perform diff only if both <obj1> and <obj1> is same type. So follow link only if the both <obj1> and <obj2> are same type as link.

- File compare case - If 'follow option' is given, h5diff will follow links since it only compares when the both objects are same type.

- Object compare case – If 'follow option' is given, check if both are link type, if so, continue and will follow links to compare. If not, it's 'Not compatible'

Suggestion 2

Follow links like the diff command does in unix – go through link if any of given objects are either soft-link or external-link. It will perform diff either both given objects are links (soft/external/hard) or one of objects is link (soft/external/hard) as well as none of the given objects is link (only hard).

- If 'follow option' is given, h5diff will follow links to figure out the end-target object and compares when the both objects are same type. If the type is dataset, h5diff will compare all the values and attribute. If the type is group, h5diff will compare the name and attribute. If the type is named-type h5diff will compare the name and attribute.

Suggestion 3

Give user control via 'follow option' with below arguments

- both – follow link if both <obj1> and <obj2> are link , otherwise it's 'Not compatible'
- src – follow <obj1> if it's link, otherwise it's 'Not compatible'
- dst – follow <obj2> if it's link, otherwise it's 'Not compatible'

- File compare case – if 'follow option' is given,
    - both- follow link if both types are link , otherwise it's 'Not compatible'
    - src – follow links in file1 , not file2. This will always be 'Not compatible'
    - dst – follow links in file2, not file1. This will always be 'Not compatible'

- Object compare case – if 'follow options' is given,
    - both – follow links if both are link
    - src – follow obj1, if it's link , otherwise it's 'Not compatible'
    - dst – follow obj2, if it's link, otherwise it's 'Not compatible'

Command options for comparing through links

For the option, we '-f' or '-l' were considered first. After further discussion, '--follow-links' option is decided as a final candidate.

Only long option will be used, so it can be commonly used among other tools without confliction issue.

Two options were added:

--follow-links

- Follow symbolic links (soft links and external links) and compare the links' target objects.
- If symbolic link(s) with the same name exist in the files being compared, then determine whether the target of each link is an existing object (dataset, group, or named datatype) or the link is a dangling link (a soft or external link pointing to a target object that does not yet exist).
    - If both symbolic links are dangling links, they are treated as being the same; by default, h5diff returns an exit code of 0. If, however, --no-dangling-links is used with --follow-links, this situation is treated as an error and h5diff returns an exit code of 2.
    - If only one of the two links is a dangling link, they are treated as being different and h5diff returns an exit code of 1. If, however, --no-dangling-links is used with --follow-links, this situation is treated as an error and h5diff returns an exit code of 2.
    - If both symbolic links point to existing objects, h5diff compares the two objects.
- If any symbolic link specified in the call to h5diff does not exist, h5diff treats it as an error and returns an exit code of 2.

--no-dangling-links

- Must be used with --follow-links option; otherwise, h5diff shows error message and returns an exit code of 2.
- Check for any symbolic links (soft links or external links) that do not resolve to an existing object (dataset, group, or named datatype). If any dangling link is found, this situation is treated as an error and h5diff returns an exit code of 2.

Past activity records:

Below list shows options for other tools for the reference. There is no specific short options for using follow link.

- Unix ln command
    - –s for symbolic link
    - Create hard links by default, symbolic links with --symbolic.
    - -f is used for force, flag,
- H5ls
    - -f : print full path name
    - –s : print 1-byte integer
    - –l : label members of compound datasets
- H5copy
    - –f : flag with args (shallow, soft, ext, ref, noattr…)
    - –s : source obj name
    - No –l
- H5dump
    - –f D : --filedriver=D : Specify which driver to open the file with
    - -l P, --soft-link=P Print the value(s) of the specified soft link
    - No -s

Option(s) examples

- h5diff **--follow-links [--no-danglink-links]** <file1> <file1>
- h5diff **--follow-links [--no-danglink-links]** <file1> <file2>
- h5diff **–follow-links [--no-danglink-links]**   <file1> <file2> <soft link1> <soft link2>
- h5diff **–follow-links [--no-danglink-links]**   <file1> <file2> <soft link> <dataset|group|named datatype>
- h5diff **–follow-links [--no-danglink-links]**   <file1> <file2> <dataset|group|named datatype> <soft link>
- h5diff **–follow-links [--no-danglink-links]**   <file1> <file2> <external link1> < external link2>
- h5diff **–follow-links [--no-danglink-links]**   <file1> <file2> <external link> <dataset/group/named datatype>
- h5diff **–follow-links [--no-danglink-links]**   <file1> <file2> <dataset|group|named datatype> <external link>
- h5diff **–follow-links [--no-danglink-links]**   <file1> <file2> <soft link> <external link>

2)

Outputs (-v, verbose mode) and exit code on dangling link detection

[with both –follow-links and –no-dangling-links]

CASE: Both dangling links - decided to treat as Error

```
h5diff –v –follow-links –no-dangling-links <file1> <file2> /link1 /link2
(both link1 and link2 are dangling links)

Warning: </link1> is a dangling link.
>> exit code 2
```

CASE: Only one is dangling link – decided to treat as Error

```
h5diff –v –follow-links –no-dangling-links <file1> <file2> /link1 /link2
(link2 is a dangling link)

Warning: </link2> is a dangling link.
>> exit code 2
```

[with –follow-links only]

CASE: Only one is dangling link – decided to treat different

```
h5diff –v –follow-links <file1> <file2> /link1 /link2
(link2 is a dangling link)

obj2  </link2> is a dangling link
1 differences found
>> exit code 1
```

CASE: Both dangling links - decided to treat as same

```
h5diff –v –follow-links <file1> <file2> /link1 /link2
(both link1 and link2 are dangling link)

dangling link: </link1> and </link2>
0 differences found
>> exit code 0
```

According to the current output sample:

```
dset: </dset1>  </dset2>0 differences
foundgroup: </grp1> </grp2>0 differences
foundsoft link: </slink1> </slink2>external
link:  </extlink1> </extlink2>>> exit code 0
```

Acknowledgements

The Chicago customer requested for this feature in h5diff command line tool. Thanks Peter for guideline.

Revision History

December 18, 2009:	Version 1 circulated for comment within The HDF Group.
Jan 10, 2010:	Version 1.1 circulated for comment within The HDF Group
Feb 10, 2010:	Version 1.2 circulated for comment within The HDF Group