Subject: Re: Another HDF File Question
Posted by liamgumley on Thu, 02 Aug 2007 18:24:37 GMT
View Forum Message <> Reply to Message

On Aug 2, 11:12 am, David Fanning <da...@dfanning.com> wrote:
> Folks,
>
> Thanks for the answer to my last question. Now another.
> My program is failing with a message from HDF_SD_START
> that it is "Unable to start the HDF-SD Interface".
> All other 67 files in the directory have no problems,
> but this 68th file causes the error, which leads me
> to believe this must be a corrupted file of some sort.
>
> But the error message leaves me in the dark. Can you
> suggest other ways I can diagnose the problem? The
> message "Sorry Charlie!" doesn't help me much. :-(

It's not a valid HDF file. So it's either a corrupted HDF file, or not HDF to begin with.

I find the following function quite useful in cases like this. It determines whether a given filename exists, has read or write permission, and whether it is in HDF or netCDF format. It also returns the file size in bytes.

```
FUNCTION FILEINFO, FILENAME

;+
; NAME:
;    FILEINFO
;
; PURPOSE:
;    Return information about a file.
;
; CATEGORY:
;    File utilities.
;
; CALLING SEQUENCE:
;    RESULT = FILEINFO(FILE)
;
; INPUTS:
;    FILE    Name of file
;
; OPTIONAL INPUTS:
;    None.
;
; KEYWORD PARAMETERS:
```

```
;    None.
;
; OUTPUTS:
;    An anonymous structure containing information about the file.
;    The fields in the structure are as follows:
;    NAME     String containing the name of the file
;    EXIST    1 if file exists, 0 otherwise.
;    READ     1 if file can be read, 0 otherwise.
;    WRITE    1 if file can be written, 0 otherwise.
;    HDF      1 if file is HDF format, 0 otherwise
;             (0 if HDF API is not available).
;    NETCDF   1 if file is netCDF format, 0 otherwise
;             (0 if netCDF API is not available).
;    SIZE     File size in bytes
;             (-1 if file does not exist, or if file cannot be read).
;
; OPTIONAL OUTPUTS:
;    None
;
; COMMON BLOCKS:
;    None
;
; SIDE EFFECTS:
;    None.
;
; RESTRICTIONS:
;    Requires IDL 5.0 or higher (square bracket array syntax).
;
; EXAMPLE:
;
;;- Check an existing file
;file = filepath('hurric.dat', subdir='examples/data')
;help, fileinfo(file), /structure
;
;;- Check a new file
;file = 'zztest.dat'
;help, fileinfo(file), /structure
;
; MODIFICATION HISTORY:
; Liam.Gumley@ssec.wisc.edu
; http://cimss.ssec.wisc.edu/~gumley
; $Id: fileinfo.pro,v 1.1 2003/06/30 20:27:21 gumley Exp $
;
; Copyright (C) 1999, 2000 Liam E. Gumley
;
; This program is free software; you can redistribute it and/or
; modify it under the terms of the GNU General Public License
; as published by the Free Software Foundation; either version 2
```

```
; of the License, or (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program; if not, write to the Free Software
; Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA
02111-1307, USA.
;-

rcs_id = '$Id: fileinfo.pro,v 1.1 2003/06/30 20:27:21 gumley Exp $'

;- Check arguments
if (n_params() eq 0) then message, 'Usage: RESULT =
FILEINFO(FILENAME)'
if (n_elements(filename) eq 0) then message, 'Argument FILENAME is
undefined'

;- Set default return value
result = {name:'', exist:0L, read:0L, write:0L, hdf:0L, netcdf:0L,
size:-1L}

;- Get file name
result.name = filename[0]

;- Get file existence status
path = (findfile(result.name))[0]
if (path ne '') then result.exist = 1

;- Get expanded file name, file size, and file read status
if (result.exist) then begin
  openr, lun, path, /get_lun, error=error
  if (error eq 0) then begin
    finfo = fstat(lun)
    result.name = finfo.name
    result.size = finfo.size
    result.read = 1
    free_lun, lun
  endif
endif

;- Get file write status
if (demo_mode() eq 0) then begin
  case result.exist of
    1 : openu, lun, result.name, /get_lun, error=error
```

```
    0 : openw, lun, result.name, /get_lun, error=error, /delete
  endcase
  if (error eq 0) then begin
    finfo = fstat(lun)
    result.name = finfo.name
    if result.exist then result.size = finfo.size
    result.write = 1
    free_lun, lun
  endif
endif

;- Get HDF status
if (result.read and hdf_exists()) then begin
  hdfid = hdf_open(result.name, /read)
  if (hdfid ne -1) then begin
    result.hdf = 1
    hdf_close, hdfid
  endif
endif

;- Get netCDF status
if (result.read and ncdf_exists()) then begin
  catch, error_status
  if (error_status ne 0) then goto, ncdf_continue
  cdfid = ncdf_open(result.name)
  result.netcdf = 1
  ncdf_close, cdfid
  ncdf_continue:
  catch, /cancel
endif

;- Return result to caller
return, result

END
```

For example, here is what happens with a file which exists, is
readable and writable, but not valid HDF.

```
$ echo "Hello world" > test.hdf
$ idl
IDL Version 6.2 (linux x86_64 m64). (c) 2005, Research Systems, Inc.
IDL> result = fileinfo('test.hdf')
IDL> help, result, /structure
** Structure <762498>, 7 tags, length=40, data length=40, refs=1:
   NAME            STRING    'test.hdf'
   EXIST           LONG                 1
   READ            LONG                 1
```

```
WRITE           LONG                 1
HDF             LONG                 0
NETCDF          LONG                 0
SIZE            LONG                12
```

Cheers,
Liam.

Practical IDL Programming
http://www.gumley.com/
