Subject: findfile on unix bug - help
Posted by R.G.S. on Thu, 04 Oct 2001 19:43:43 GMT
View Forum Message <> Reply to Message

Hi,

I have a problem running my code on a unix machine
(idlv5.4). The command is as follows:

files = findfile(subdirectory+'*.dat',count=num)

This does not give me the files in that subdirectory, rather it gives me the
files in the current folder.

So now if I do the same command as:
files = findfile(subdirectory+'*',count=num)
I do get the entire listing. This is how I work around it,
however, it is possible that there are files other than the *.dat
and I want to filter on those. This works in win2000 nicely, so
what is the problem on the unix side?

I can use * in unix command lines. For instance
> ls *.txt
lists only the txt files.

So how do I perform this command so as to find and filter
the *.dat files?

Cheers,
bob stockwell
colorado research associates
boulder co usa

-- didn't find anything helpful on google groups

Subject: Re: findfile on unix bug - help
Posted by thompson on Thu, 04 Oct 2001 21:32:29 GMT
View Forum Message <> Reply to Message

"R.G.S." <rgs1967@hotmail.com> writes:

> Hi,

> I have a problem running my code on a unix machine
> (idlv5.4). The command is as follows:

> files = findfile(subdirectory+'*.dat',count=num)

> This does not give me the files in that subdirectory, rather it gives me the
> files in the current folder.

> So now if I do the same command as:
> files = findfile(subdirectory+'*',count=num)
> I do get the entire listing. This is how I work around it,
> however, it is possible that there are files other than the *.dat
> and I want to filter on those. This works in win2000 nicely, so
> what is the problem on the unix side?

I've never seen this behavior on any Unix platform that I've used, and can't replicate it with IDL/v5.4.1 on my Alpha/OSF workstation. Two things occur to me:

1. The simplest answer is that the variable subdirectory is ending up as a blank string. That would definitely cause the behavior you're seeing. Have you verified that it contains what you think it contains?

2. We've had problems in Unix with findfile when the directory is very large, but the symptoms of that is that the result ends up being the null string, with a count of 0. Maybe this is somehow causing your problem, although it doesn't sound like it.

My colleague Stein Vidar Haugan got around this problem by writing a program called find_file.pro, which I've appended below. There are some additional routines which can be found at

 ftp://sohoftp.nascom.nasa.gov/solarsoft/gen/idl/

Bill Thompson

```
;+
; Project     : SOHO - CDS
;
; Name        : FIND_FILE()
;
```

```
; Purpose     : Fixing builtin FINDFILE() problem
;
; Explanation : The builtin FINDFILE() function has problems on some unixes
;               whenever *a lot* of files are matching the file
;               specification. This is due to the fact that filename expansion
;               is done by the shell *before* interpreting a command. Too many
;               files cause too long commands, which are not accepted.  This
;               causes FINDFILE() to return an empty list of candidates.
;
;               FIND_FILE tries the builtin function first, and whenever the
;               returned list of files is empty, it tries to recheck through
;               spawning a "find" command.
;
;               Since FINDFILE doesn't discriminate between directories, links
;               and files, this function will not do this either.
;
;               Under unix, however, calls like FINDFILE("*") returns the
;               unfiltered output of the shell commmand "ls *", including
;               colon-terminated lines for each subdirectory matching the
;               specification and empty lines separating each subdirectory
;               listing. Such silly effects are not implemented in the "find"
;               version. Be warned, however, that these effects are present
;               when the builtin function does not "fail" due to a too long
;               file list.
;
;               It is possible (under unix) to use the "find" method as
;               default by setting the keyword /USEFIND (no effect under other
;               operating systems).
;
; Use         : files = find_file(file_specification)
;
; Inputs      : file_specification : A scalar string used to find
;                                   files. See FINDFILE()
;
; Opt. Inputs : None.
;
; Outputs     : Returns a list of files or a blank string if none found.
;
; Opt. Outputs:
;
; Keywords    : COUNT : Returns the number of files
;
;               USEFIND : Always use a spawned "find" command under unix.
;                     No effect under other operating systems.
;
;  NODOT : Apply a filter to the results from find to prevent
;   finding the directory itself in a large file expansion.
;   eg 'find_file,"foo/*"' returns ("foo/","foo/a",...)
```

```
;   but 'find_file,"foo/*",/nodot' returns
;   ("foo/a","foo/b",...) without the leading "foo/".
;   This behavior is closer to the behavior of findfile()
;   without the long-directory braindamage.  It is
;   *not* the default so as not to break heritage
;   code that uses find_file().
;
; Calls       : BREAD_FILE, FINDFILE, SPAWN
;
; Common      : None
;
; Restrictions: As for FINDFILE
;
; Side effects: None, hopefully
;
; Category    : Utilities, Operating_system
;
; Prev. Hist. : Lots of problems with FINDFILE is hopefully history.
;
; Written     : S.V.H. Haugan, UiO, 12 April 1996
;
; Modified    : Version 2, SVHH, 10 June 1996
;                     Moved the CD,curr_path command to avoid
;                     returns without resetting path.
;               Version 3, SVHH, 26 June 1996
;                     Took away the -type f argument to find, added
;                     /USEFIND keyword.
;       : Added /nodot keyword C. DeForest 9-August-1998
;
; Version     : 3, 26 June 1996
;-

FUNCTION  find_file,file_specification,count=count,usefind=usefind,nod ot=nodot
  count = 0
  use_find = KEYWORD_SET(usefind) AND os_family() EQ 'unix'

  IF NOT use_find AND N_PARAMS() EQ 0 THEN BEGIN
    result = findfile(count = count)
    RETURN,result             ; Unix doesn't have problems with this
  END

  IF N_PARAMS() EQ 0 THEN file_specification = '*'
  IF file_specification EQ '' THEN file_specification = '*'

  IF NOT use_find THEN result = findfile(file_specification,count=count) $
  ELSE count = 0

  ;; Check for problems
```

```
 IF count EQ 0 AND os_family() EQ 'unix' THEN BEGIN
   file = file_specification
   break_file,file,disk,dir,filnam,ext

   ;; Check if directory exists
   IF dir NE '' THEN BEGIN
     IF (findfile(dir))(0) eq '' THEN RETURN,''
   END

   ;; Temporary switch to that directory
   IF dir NE '' THEN cd,dir,current=curr_path

   IF filnam+ext EQ '' THEN filnam = '*'

   ;; Find all matching
   spn = ["find",".","-name",filnam+ext,"-print"]
   spawn,spn,result,/noshell

   ;; Switch back to original directory
   IF dir NE '' THEN cd,curr_path


   IF result(0) EQ '' THEN RETURN,'' ; None matching, return

   ;; Get rid of current-directory match, if necessary
   if keyword_set(nodot) and result(0) eq '.' then $
result = result(1:n_elements(result)-1)

   ;; Chop off './'
   result = STRMID(result,2,1000)

   ;; Chip out subdirectories (for some reason, the -prune option doesn't
   ;; work properly, so I have dropped using it).

   ix = WHERE(STRPOS(result,'/') EQ -1,count)
   IF count EQ 0 THEN RETURN,''

   ;; Put back the specified (not full) path
   result = dir + result(ix)
 END

 RETURN,result

END

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;;;;;
; End of 'findfile.pro'.
```

;; ;;;;;;;;;;;

Subject: Re: findfile on unix bug - help
Posted by R.G.S. on Thu, 04 Oct 2001 21:48:23 GMT
View Forum Message <> Reply to Message

William Thompson <thompson@orpheus.nascom.nasa.gov> wrote in message
news:9pikhdpkb1@skates.gsfc.nasa.gov...
> "R.G.S." <rgs1967@hotmail.com> writes:
>
>> Hi,
>
>
>
>> I have a problem running my code on a unix machine
>> (idlv5.4). The command is as follows:
>
>
>> files = findfile(subdirectory+'*.dat',count=num)
>
>> This does not give me the files in that subdirectory, rather it gives me
the
>> files in the current folder.
>
>> So now if I do the same command as:
>> files = findfile(subdirectory+'*',count=num)
>> I do get the entire listing. This is how I work around it,
>> however, it is possible that there are files other than the *.dat
>> and I want to filter on those. This works in win2000 nicely, so
>> what is the problem on the unix side?
>
> I've never seen this behavior on any Unix platform that I've used, and
can't
> replicate it with IDL/v5.4.1 on my Alpha/OSF workstation. Two things
occur to
> me:
>
> 1. The simplest answer is that the variable subdirectory is ending up as
a
> blank string. That would definitely cause the behavior you're seeing.
Have
> you verified that it contains what you think it contains?

It has been verified.
In fact, I can change the code to
cd,subdirectory
files = findfiles('*.dat')

and this works (of course, I shudder at the thought of using code that changes the current directory).

The problem is with the path somehow. (And the results can be pretty weird, although I haven't nailed them down just yet. But I swear this code worked last week.).

Thanks for the code. I'll check it out.
I am using this function on both window2000 machines as well as unix (and soon a linux box). I wanted the cross platform compatibily so I do not have different versions, hence the desire to use IDLs findfile.

Cheers,
bob

Subject: Re: findfile on unix bug - help
Posted by Paul Manusiu on Mon, 08 Oct 2001 14:00:43 GMT
View Forum Message <> Reply to Message

> and this works (of course, I shudder at the thought of using code that
> changes the current directory).
>

Under Unix, CD does not affect the working directory of the process that started IDL.

>
> The problem is with the path somehow. (And the results can be pretty
> weird, although I haven't nailed them down just yet. But I swear this
> code worked last week.).
>

Check that there isn't a blank character in your subdirectory string. For example use
strtrim(subdirectory,1)

Also if its a subdirectory of the current directory (ie relative path) check that you have not place " / "
at the beginning of your path.

>
> Thanks for the code. I'll check it out.
> I am using this function on both window2000 machines as well as unix
> (and soon a linux box). I wanted the cross platform compatibily so I
> do not have different versions, hence the desire to use IDLs findfile.

>
> Cheers,
> bob

Subject: Re: findfile on unix bug - help
Posted by R.G.S. on Mon, 08 Oct 2001 14:35:13 GMT
View Forum Message <> Reply to Message

Paul Manusiu <phpvm@alinga.newcastle.edu.au> wrote in message
news:3BC1B18B.B708E9C7@alinga.newcastle.edu.au...
>> and this works (of course, I shudder at the thought of using code that
>> changes the current directory).
>>
>
> Under Unix, CD does not affect the working directory of the process that
> started IDL.

I did not know that. If that is info that is kept by IDL, does it extend to
the following question: If I have two IDL sessions running, they won't
cross talk the current directory? That is my fear with this solution, that
process1 would change directory to /data, then process2 would change
it to /output, then process1 would try to read data files and fail.
If this won't happen, then the cd,subdirectory kludge isn't so bad.

> Check that there isn't a blank character in your subdirectory string. For
> example use
> strtrim(subdirectory,1)

No blank characters, I checked that. I have used
strcompress(subdir+'*."+extension,/remove_all).
Same results.

> Also if its a subdirectory of the current directory (ie relative path)
check
> that you have not place " / "
> at the beginning of your path.

I use absolute paths (since the data is kept in its own tree, separate
from my user tree, or the IDL tree).

Cheers,

bob

Subject: Re: findfile on unix bug - help
Posted by Paul Manusiu on Tue, 09 Oct 2001 05:54:24 GMT
View Forum Message <> Reply to Message

"R.G.S." wrote:

> Paul Manusiu <phpvm@alinga.newcastle.edu.au> wrote in message
> news:3BC1B18B.B708E9C7@alinga.newcastle.edu.au...
>>> and this works (of course, I shudder at the thought of using code that
>>> changes the current directory).
>>>
>>
>> Under Unix, CD does not affect the working directory of the process that
>> started IDL.
>
> I did not know that. If that is info that is kept by IDL, does it extend to
> the following question: If I have two IDL sessions running, they won't
> cross talk the current directory? That is my fear with this solution, that
> process1 would change directory to /data, then process2 would change
> it to /output, then process1 would try to read data files and fail.
> If this won't happen, then the cd,subdirectory kludge isn't so bad.
>

Well if you run the first IDL session with the & option under unix than the
second IDL session should run under a different shell. The shells would not know
what the other is doing so there should be no problems. Although I could be
wrong !

Cheers
Paul M.