Subject: findfile gives 'Array has a corrupted descriptor' error
Posted by b_gom on Fri, 26 Jul 2013 00:44:55 GMT

View Forum Message <> Reply to Message

I running IDL 8.2.3 on Win7 64bit. I have an older program that uses findfile() to recursively find a set of filenames with a wildcard. I realize findfile is obsolete, but it runs *much* faster than file_search. When the findfile returns more than ~5000 files, however, I get the following error:

found=file_search(uval.path+'*',count=count,/mark_dir)
% Array has a corrupted descriptor: FOUND.

Any ideas what is causing the error?
Assuming that this is a bug that will not be fixed, does anyone have a fast alternative to file_search?

Thanks

Subject: Re: findfile gives 'Array has a corrupted descriptor' error
Posted by David Fanning on Fri, 26 Jul 2013 01:05:31 GMT

View Forum Message <> Reply to Message

b_gom@hotmail.com writes:

> I running IDL 8.2.3 on Win7 64bit. I have an older program that uses findfile() to recursively find a set of filenames with a wildcard. I realize findfile is obsolete, but it runs *much* faster than file_search. When the findfile returns more than ~5000 files, however, I get the following error:
>
> found=file_search(uval.path+'*',count=count,/mark_dir)
> % Array has a corrupted descriptor: FOUND.
>
> Any ideas what is causing the error?

A bug in FindFile that occurs at about this number of files.

> Assuming that this is a bug that will not be fixed, does anyone have a fast alternative to file_search?

No, sorry. :-)

Cheers,

David

P.S. Do you know the old joke about computers making very fast, very
accurate mistakes?

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: findfile gives 'Array has a corrupted descriptor' error
Posted by Phillip Bitzer on Fri, 26 Jul 2013 12:54:32 GMT

View Forum Message <> Reply to Message

On Thursday, July 25, 2013 7:44:55 PM UTC-5, b_...@hotmail.com wrote:
> I running IDL 8.2.3 on Win7 64bit. I have an older program that uses findfile() to recursively find a set of filenames with a wildcard. I realize findfile is obsolete, but it runs *much* faster than file_search. When the findfile returns more than ~5000 files, however, I get the following error:
>
>
> found=file_search(uval.path+'*',count=count,/mark_dir)
>

Are you looking for all files in recursive directories? If so, try this on for size:

```
found = file_search(uval.path, '*', count=count,/mark_dir)
```

Subject: Re: findfile gives 'Array has a corrupted descriptor' error
Posted by wlandsman on Fri, 26 Jul 2013 15:29:41 GMT

View Forum Message <> Reply to Message

On Thursday, July 25, 2013 8:44:55 PM UTC-4, b_...@hotmail.com wrote:

> Assuming that this is a bug that will not be fixed, does anyone have a fast alternative to file_search?

Not a direct answer but I do notice on the Mac that file_search() is slow only on the first call:

```
IDL> tic & a = file_search('.','*.pro',/nosort) & toc
% Time elapsed: 41.371398 seconds.

IDL> tic & a = file_search('.','*.pro',/nosort) & toc
% Time elapsed: 0.45945001 seconds.
```

So file_search() was of order 100 times faster on the second call. This is similar to the Unix find command which stores the information of a search to speed up the processing on subsequent calls.

(I included /nosort because that is supposed to speed things up somewhat but it seemed to make little difference on the Mac).

If your recursive search includes a lot of unnecessary directories, then it might be quicker to use a vector of plausible directories in you file_search() call, rather than searching every directory below the specified one. --Wayne

Subject: Re: findfile gives 'Array has a corrupted descriptor' error
Posted by David Fanning on Fri, 26 Jul 2013 15:34:19 GMT

View Forum Message <> Reply to Message

wlandsman writes:

>
> On Thursday, July 25, 2013 8:44:55 PM UTC-4, b_...@hotmail.com wrote:
>
>> Assuming that this is a bug that will not be fixed, does anyone have a fast alternative to file_search?
>
> Not a direct answer but I do notice on the Mac that file_search() is slow only on the first call:
>
> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
> % Time elapsed: 41.371398 seconds.
>
> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
> % Time elapsed: 0.45945001 seconds.
>
>
> So file_search() was of order 100 times faster on the second call.   This is similar to the Unix find command which stores the information of a search to speed up the processing on subsequent calls.
> (I included /nosort because that is supposed to speed things up somewhat but it seemed to make little difference on the Mac).
>
> If your recursive search includes a lot of unnecessary directories, then it might be quicker to use a vector of plausible directories in you file_search() call, rather than searching every directory below the specified one. --Wayne

The speed up doesn't seem to be so pronounced on Windows:

```
IDL> tic & a = file_search('.','*.pro',/nosort) & toc
 Elapsed Time:  9.873000
IDL> tic & a = file_search('.','*.pro',/nosort) & toc
 Elapsed Time:  5.622000
IDL> tic & a = file_search('.','*.pro',/nosort) & toc
 Elapsed Time:  5.600000
```

This command found 5128 files.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: findfile gives 'Array has a corrupted descriptor' error
Posted by b_gom on Fri, 26 Jul 2013 17:34:24 GMT

View Forum Message <> Reply to Message

The program in question is an old compound widget that has been working happily up until the last IDL release. I've noticed more IDL crashes with the last release, but this is the only one that has an obvious cause.

The issue with this widget is that it traverses a directory tree and builds a tree widget with the directories and any files matching a search string. This is being done with a recursive function that builds the tree nodes as it goes, which means *many* calls to findfile(). The only way file_search() would work is if I use it to return the entire directory structure, and parse the result to build the tree, which would mean a major rewrite. Being lazy, I was hoping there was a working equivalent to findfile(). Sigh.

On Friday, July 26, 2013 9:34:19 AM UTC-6, David Fanning wrote:
> wlandsman writes:
>
>
>
>>
>
>> On Thursday, July 25, 2013 8:44:55 PM UTC-4, b_...@hotmail.com wrote:
>
>>
>
>>> Assuming that this is a bug that will not be fixed, does anyone have a fast alternative to file_search?
>
>>
>
>> Not a direct answer but I do notice on the Mac that file_search() is slow only on the first call:
>

>>
>
>> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>> % Time elapsed: 41.371398 seconds.
>
>>
>
>> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>> % Time elapsed: 0.45945001 seconds.
>
>>
>
>>
>
>> So file_search() was of order 100 times faster on the second call.   This is similar to the Unix find command which stores the information of a search to speed up the processing on subsequent calls.
>
>> (I included /nosort because that is supposed to speed things up somewhat but it seemed to make little difference on the Mac).
>
>>
>
>> If your recursive search includes a lot of unnecessary directories, then it might be quicker to use a vector of plausible directories in you file_search() call, rather than searching every directory below the specified one. --Wayne
>
>
>
> The speed up doesn't seem to be so pronounced on Windows:
>
>
>
> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>   Elapsed Time: 9.873000
>
> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>   Elapsed Time: 5.622000
>
> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>   Elapsed Time: 5.600000
>
>

>
> This command found 5128 files.
>
>
>
> Cheers,
>
>
>
> David
>
>
>
> --
>
> David Fanning, Ph.D.
>
> Fanning Software Consulting, Inc.
>
> Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
>
> Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: findfile gives 'Array has a corrupted descriptor' error
Posted by b_gom on Fri, 26 Jul 2013 19:54:15 GMT
View Forum Message <> Reply to Message

Welll, I've optimized the widget code and managed to reduce the number of calls to file_search to the bare minimum, but the show-stopping issue is that file_search is basically unusable on network shares (CIFS/SMB).

For example, file_search takes 26 seconds (!!) to list a folder with ~7000 files:

```
IDL> tic & found=file_search('U:\somenetworkshare\*',count=count) & toc
% Time elapsed: 26.115000 seconds.
IDL> tic & found=file_search('U:\somenetworkshare\*',count=count) & toc
% Time elapsed: 26.052000 seconds.
IDL> tic & found=file_search('\\server\pathtoshare\*',count=count) & toc
% Time elapsed: 26.110000 seconds.
```

Whereas findfile does the same job in no time (except that at random times it crashes with a 'array has corrupted descriptor' fault):

```
IDL> tic & found=findfile('U:\somenetworkshare\*',count=count) & toc
% Time elapsed: 0.63899994 seconds.
```

What in the world is file_search doing?

On Friday, July 26, 2013 11:34:24 AM UTC-6, b_...@hotmail.com wrote:
> The program in question is an old compound widget that has been working happily up until the last IDL release. I've noticed more IDL crashes with the last release, but this is the only one that has an obvious cause.
>
>
>
> The issue with this widget is that it traverses a directory tree and builds a tree widget with the directories and any files matching a search string. This is being done with a recursive function that builds the tree nodes as it goes, which means *many* calls to findfile(). The only way file_search() would work is if I use it to return the entire directory structure, and parse the result to build the tree, which would mean a major rewrite. Being lazy, I was hoping there was a working equivalent to findfile(). Sigh.
>
>
>
>
>
>
>
> On Friday, July 26, 2013 9:34:19 AM UTC-6, David Fanning wrote:
>
>> wlandsman writes:
>
>>
>
>>
>
>>
>
>>>
>
>>
>
>>> On Thursday, July 25, 2013 8:44:55 PM UTC-4, b_...@hotmail.com wrote:
>
>>
>
>>>
>
>>
>
>>>> Assuming that this is a bug that will not be fixed, does anyone have a fast alternative to file_search?

>
>>
>
>>>
>
>>
>
>>> Not a direct answer but I do notice on the Mac that file_search() is slow only on the first call:
>
>>
>
>>>
>
>>
>
>>> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>>
>
>>> % Time elapsed: 41.371398 seconds.
>
>>
>
>>>
>
>>
>
>>> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>>
>
>>> % Time elapsed: 0.45945001 seconds.
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> So file_search() was of order 100 times faster on the second call.   This is similar to the Unix find command which stores the information of a search to speed up the processing on subsequent calls.
>
>>

>
>>> (I included /nosort because that is supposed to speed things up somewhat but it seemed to make little difference on the Mac).
>
>>
>
>>>
>
>>
>
>>> If your recursive search includes a lot of unnecessary directories, then it might be quicker to use a vector of plausible directories in you file_search() call, rather than searching every directory below the specified one. --Wayne
>
>>
>
>>
>
>>
>
>> The speed up doesn't seem to be so pronounced on Windows:
>
>>
>
>>
>
>>
>
>> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>>
>
>>   Elapsed Time: 9.873000
>
>>
>
>> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>>
>
>>   Elapsed Time: 5.622000
>
>>
>
>> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>>
>

>> Elapsed Time: 5.600000
>
>>
>
>>
>
>>
>
>> This command found 5128 files.
>
>>
>
>>
>
>>
>
>> Cheers,
>
>>
>
>>
>
>>
>
>> David
>
>>
>
>>
>
>>
>
>> --
>
>>
>
>> David Fanning, Ph.D.
>
>>
>
>> Fanning Software Consulting, Inc.
>
>>
>
>> Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
>
>>
>

>> Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: findfile gives 'Array has a corrupted descriptor' error
Posted by b_gom on Fri, 26 Jul 2013 23:51:47 GMT

View Forum Message <> Reply to Message

Some further information:

when testing on a Linux system, accessing the same CIFS share, I get the following:

```
IDL> tic & found=file_search('/somenetworkshare/*',count=count) & toc
% Time elapsed: 0.25748897 seconds.
IDL> tic & found=file_search('/somenetworkshare/*',count=count) & toc
% Time elapsed: 0.26749086 seconds.
```

For Linux, it seems that findfile is slower than file_search, but still consistent with the Windows results:

```
IDL> tic & found=findfile('/somenetworkshare/*',count=count) & toc
% Time elapsed: 0.54775500 seconds.
```

On Friday, July 26, 2013 1:54:15 PM UTC-6, b_...@hotmail.com wrote:
> Welll, I've optimized the widget code and managed to reduce the number of calls to file_search to the bare minimum, but the show-stopping issue is that file_search is basically unusable on network shares (CIFS/SMB).
>
>
>
> For example, file_search takes 26 seconds (!!) to list a folder with ~7000 files:
>
>
>
> IDL> tic & found=file_search('U:\somenetworkshare\*',count=count) & toc
>
> % Time elapsed: 26.115000 seconds.
>
> IDL> tic & found=file_search('U:\somenetworkshare\*',count=count) & toc
>
> % Time elapsed: 26.052000 seconds.
>
> IDL> tic & found=file_search('\\server\pathtoshare\*',count=count) & toc
>
> % Time elapsed: 26.110000 seconds.
>
>
>
> Whereas findfile does the same job in no time (except that at random times it crashes with a

'array has corrupted descriptor' fault):
>
>
>
> IDL> tic & found=findfile('U:\somenetworkshare\*',count=count) & toc
>
> % Time elapsed: 0.63899994 seconds.
>
>
>
> What in the world is file_search doing?
>
>
>
>
>
>
>
>
>
> On Friday, July 26, 2013 11:34:24 AM UTC-6, b_...@hotmail.com wrote:
>
>> The program in question is an old compound widget that has been working happily up until the last IDL release. I've noticed more IDL crashes with the last release, but this is the only one that has an obvious cause.
>
>>
>
>>
>
>>
>
>> The issue with this widget is that it traverses a directory tree and builds a tree widget with the directories and any files matching a search string. This is being done with a recursive function that builds the tree nodes as it goes, which means *many* calls to findfile(). The only way file_search() would work is if I use it to return the entire directory structure, and parse the result to build the tree, which would mean a major rewrite. Being lazy, I was hoping there was a working equivalent to findfile(). Sigh.
>
>>
>
>>
>
>>
>
>>
>
>>

>
>>
>
>>
>
>> On Friday, July 26, 2013 9:34:19 AM UTC-6, David Fanning wrote:
>
>>
>
>>> wlandsman writes:
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>>
>
>>
>
>>>
>
>>
>
>>>> On Thursday, July 25, 2013 8:44:55 PM UTC-4, b_...@hotmail.com wrote:
>
>>
>
>>>
>
>>
>
>>>>
>
>>
>
>>>
>
>>

>
>>>> > Assuming that this is a bug that will not be fixed, does anyone have a fast alternative to file_search?
>
>>
>
>>>
>
>>
>
>>>>
>
>>
>
>>>
>
>>
>
>>>> Not a direct answer but I do notice on the Mac that file_search() is slow only on the first call:
>
>>
>
>>>
>
>>
>
>>>>
>
>>
>
>>>
>
>>
>
>>>> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>>
>
>>>
>
>>
>
>>>> % Time elapsed: 41.371398 seconds.
>
>>
>
>>>
>

>>
>
>>>>
>
>>
>
>>>
>
>>
>
>>>> IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>>
>
>>>
>
>>
>
>>>> % Time elapsed: 0.45945001 seconds.
>
>>
>
>>>
>
>>
>
>>>>
>
>>
>
>>>
>
>>
>
>>>>
>
>>
>
>>>
>
>>
>
>>>> So file_search() was of order 100 times faster on the second call.   This is similar to the Unix find command which stores the information of a search to speed up the processing on subsequent calls.
>
>>
>

>>>
>
>>
>
>>>> (I included /nosort because that is supposed to speed things up somewhat but it seemed to make little difference on the Mac).
>
>>
>
>>>
>
>>
>
>>>>
>
>>
>
>>>
>
>>
>
>>>> If your recursive search includes a lot of unnecessary directories, then it might be quicker to use a vector of plausible directories in you file_search() call, rather than searching every directory below the specified one. --Wayne
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> The speed up doesn't seem to be so pronounced on Windows:
>
>>
>
>>>
>
>>
>
>>>

```
>
>>
>
>>>
>
>>
>
>>>  IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>>
>
>>>
>
>>
>
>>>   Elapsed Time:  9.873000
>
>>
>
>>>
>
>>
>
>>>  IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>>
>
>>>
>
>>
>
>>>   Elapsed Time:  5.622000
>
>>
>
>>>
>
>>
>
>>>  IDL> tic & a = file_search('.','*.pro',/nosort) & toc
>
>>
>
>>>
>
>>
>
>>>   Elapsed Time:  5.600000
```

>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> This command found 5128 files.
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> Cheers,
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> David

>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> --
>
>>
>
>>>
>
>>
>
>>> David Fanning, Ph.D.
>
>>
>
>>>
>
>>
>
>>> Fanning Software Consulting, Inc.
>
>>
>
>>>
>
>>
>
>>> Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
>
>>
>
>>>
>
>>
>
>>> Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: findfile gives 'Array has a corrupted descriptor' error
Posted by b_gom on Tue, 30 Jul 2013 17:16:30 GMT
View Forum Message <> Reply to Message

Forgive the sin of continuously replying to my own post, but here is the workaround I've used. Spawning the 'dir' command takes much less time than file_search on network shares with many files:

```
IDL> tic & spawn, 'dir U:\somenetworkshare\* /b /aD',found,/hide & toc
% Time elapsed: 0.32800007 seconds.
IDL> tic &  found=file_search('U:\somenetworkshare\*',count=count,/nosor t) & toc
% Time elapsed: 26.066000 seconds.
```

So, I wrote a wrapper for the file_search function that determines if the Windows OS is in use, and if the VM mode is not in use, and then does the following:

```
function listfiles,path,pattern,count=count,_extra=e
  if n_elements(pattern) eq 0 then pattern='*'
  if LMGR(/VM) then begin
     return,file_search(path+pattern,/test_regular,count=count,_e xtra=e)   ;forced to use slow
version in VM mode.
  endif
  case strupcase(!version.os_family) of
    'WINDOWS':begin
      spawn, 'dir '+path+pattern+' /b /a-D /ON',result,/hide
      count = (result[0] eq '') ? 0 : n_elements(result)
      return,file_dirname(path+pattern,/mark)+result
      end
    'UNIX':begin
       return,file_search(path+pattern,/test_regular,count=count,_e xtra=e)
      end
  endcase
end
```

P.S., I've also found that file_search is slow to return a large list of file matches (>~5000) from a given directory, but not when returning a short list of matches from the same directory. For example, in a directory of ~7000 files, file_search(path+'*') takes around 26 seconds, but file_search(path+'*.txt') returns in 0.3 seconds if there are only a few .txt files.
So, the above workaround actually costs a bit more time for cases where only small file lists are expected.

On Thursday, July 25, 2013 6:44:55 PM UTC-6, b_...@hotmail.com wrote:
> I running IDL 8.2.3 on Win7 64bit. I have an older program that uses findfile() to recursively find a set of filenames with a wildcard. I realize findfile is obsolete, but it runs *much* faster than file_search. When the findfile returns more than ~5000 files, however, I get the following error:
>
>

>
> found=file_search(uval.path+'*',count=count,/mark_dir)
>
> % Array has a corrupted descriptor: FOUND.
>
>
>
> Any ideas what is causing the error?
>
> Assuming that this is a bug that will not be fixed, does anyone have a fast alternative to file_search?
>
>
>
> Thanks