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.")