
Subject: Re: allocating memory

Posted by [pook41](#) on Sun, 21 Oct 2007 02:25:06 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Oct 21, 6:25 am, "R.G. Stockwell" <noem...@please.com> wrote:

> "Andrew Cool" <andrew.c...@dsto.defence.gov.au> wrote in message

>

> news:1192868061.603580.289300@e9g2000prf.googlegroups.com...

>

>

>

>> On Oct 19, 12:37 am, "R.G. Stockwell" <noem...@please.com> wrote:

>>> <jtmcah...@gmail.com> wrote in message

>

>>> news:1192673434.544923.139390@i38g2000prf.googlegroups.com.. .

>

>>>> So, I have this large model file that I need to open. Although it is

>>>> 650MB I should be able to open it in IDL on my pc computer which has

>>>> ~4GB of Memory. However, it keeps telling me insufficient memory.

>>>> However, if I try to open it in IDL on a linux machine with ~2GB

>>>> memory I can open it no problem. Is there a way to make my windows

>>>> based pc cooperate and allow me to open this file that should be no

>>>> problem to open?

>

>>>> Thanks,

>>>> Hawaiianite

>

>>> I've attached a short program memtest.pro below. I grabbed this

>>> off the newsgroup.

>

>>> It shows you the memory sizes you can allocate.

>

>>> The problem may be fragmentation of your ram by the many

>>> dlls loaded by windows and other programs. I'd remove everything

>>> you can from the startup (and other automatically loading programs)

>>> and reboot (remove spyware, antivirus, firewalls, mail programs, but

>>> be careful not to forget to turn them back on). That may help.

>

>>> pro memtest

>>> compile_opt idl2 ; set default integers to 32-bit and enforce [] for

>>> indexing

>

>>> MB = long64(2)^20

>>> currentBlockSize = MB * 2047 ; 2 GB

>

>>> print,'current block size = ',currentblocksize

>>> maxIterations = 10 ; Max loop iterations

>>> memPtrs = ptrarr(maxIterations)

```

>>> memBlockSizes = ulongarr(maxIterations)
>
>>> for i=0, maxIterations-1 do begin
>>>   ; Error handler
>>>   catch, err
>
>>>   ; Sepcifically designed for "Failure to allocate memory..." error
>>>   if (err ne 0) then begin
>>>     currentBlockSize = currentBlockSize - MB ; ...try 1 MB smaller
>>> allocation
>>>   if (currentBlockSize lt MB) then break ; Give up, if
>>> currentBlockSize
>>> < 1 MB
>>>   endif
>
>>> ; This 'wait' enables Ctrl-Break to interrupt if necessary (Windows).
>>>   wait, 0.0001
>
>>> ; Allocate memory (if possible)
>>>   memPtrs[i] = ptr_new(bytearr(currentBlockSize, /nozero), /no_copy)
>>>   memBlockSizes[i] = currentBlockSize ; Store the latest successful
>>> allocation size
>
>>> ; Print the current allocated block size and the running total, in Mb
>>>   print, format=('%"Memory block #"%2d: %6d Mb (total: %4d Mb)"', $
>>>     i + 1, ishft(currentBlockSize, -20),
>>> ishft(ulong(total(memBlockSizes)), -20)
>>>   endfor
>
>>>   ptr_free, memPtrs
>>> end
>
>> Hmm, Here's what I get on my 4GB of RAM Quad core system running 64
>> bit IDL (v6.4) under 64 bit Vista :-
>
>> IDL> memtest
>> current block size =      2146435072
>> Memory block # 1:  2047 Mb (total: 2047 Mb)
>> Memory block # 2:  2047 Mb (total: 4094 Mb)
>> Memory block # 3:  2047 Mb (total: 2045 Mb)
>> Memory block # 4:  2045 Mb (total: 4090 Mb)
>> Memory block # 5:  2043 Mb (total: 2037 Mb)
>> Memory block # 6:  2041 Mb (total: 4078 Mb)
>> Memory block # 7:  1803 Mb (total: 1785 Mb)
>
>> Can't say I know how to interpret that at all !!
>
> YOWZA! I'd interpret that as: you have scads of memory available

```

> (yes scads).
>
> and you should be able to allocate 650mb (since you allocate 2gb twice)
> very easily.
> Is the problem somewhere else? (check the actual size you are trying
> to allocate for instance)
>
> The best I can do (on my system) is slightly less than 1gb on a 4gb system,
> so I ended up running my image processing on a 64 bit machine
> with 16gb of ram.
>
> Cheers,
> bob

Hi Bob,

I think you've confused me with the OP, "Hawaiianite".

I just ran memtest out of interest. After a little thought, I may have the reason why my PC appears to have scads of memory. Vista has a functionality called ReadyBoost, whereby you can utilise the memory in a fast USB stick as extra RAM (effectively). My PC has 4GB of RAM, and I'm running a 4GB USB stick dedicated to ReadyBoost.

My HDD's seldom need to spin up, and I reckon that memtest is allocating 2GB on the USB stick, and 2GB in actual RAM.

Maybe Karl Schultz can comment on whether this is indeed the case?

I should test that idea by removing the ReadyBoost USB stick, and running memtest again!

Cheers,

Andrew
