Subject: memory allocation on Macs
Posted by pgrigis on Thu, 01 May 2008 16:18:31 GMT
View Forum Message <> Reply to Message

Hi folks,

we have pretty much exhausted the topic of memory
allocation on Windows and Linux, but I don't remember
any discussion abut this on Mac OS.

So, I am using IDL 6.3 on Mac OS X 10.4.11.

I tried allocating as many 1GB array as possible,
and it failed after 3 successful allocations.
Now, the "Activity Monitor" indicates that at this point
I have 3.6 GB of memory used and 3.4 GB free.
So I am wondering why cant'I allocate a couple more
of 1GB arrays?

Cheers,
Paolo

```
IDL> a=bytarr(1000,1000,1000)
IDL> b=bytarr(1000,1000,1000)
IDL> c=bytarr(1000,1000,1000)
IDL> d=bytarr(1000,1000,1000)
idl(318,0xa000d000) malloc: *** vm_allocate(size=1000001536) failed
(error code=3)
idl(318,0xa000d000) malloc: *** error: can't allocate region
idl(318,0xa000d000) malloc: *** set a breakpoint in szone_error to
debug
% Unable to allocate memory: to make array.
  Cannot allocate memory
% Execution halted at: $MAIN$
I
```

Subject: Re: memory allocation on Macs
Posted by henrygroe on Sat, 03 May 2008 14:20:48 GMT
View Forum Message <> Reply to Message

On May 2, 12:39 pm, "Kenneth P. Bowman" <k-bow...@null.edu> wrote:
> In article
> < c76c7777-b3ab-4645-b22a-c90be59a1...@t54g2000hsg.googlegroup s.com >,
> Brian Larsen <balar...@gmail.com> wrote:
>
>> Rick,
>

>> On this point
>
>>> In any case, if you need a single process size to exceed 4GB, use a 64-
>>> bit OS.
>
>> at the risk of being a little ignorant here. I ran this test on
>> Leopard (10.5.2) which is reported to be a 64-bit OS...
>> http://www.apple.com/macosx/technology/64bit.html
>> Making this an issue of having 32-bit IDL not a 32-bit OS. Right?
>
> Yes, I'm sure I'm not the only one waiting for ITTVIS
> to make a 64-bit version for Mac OS X.
>
> Ken

I'm waiting too...
-henry

Subject: Re: memory allocation on Macs
Posted by dcleon@gmail.com on Mon, 05 May 2008 13:49:28 GMT
View Forum Message <> Reply to Message

On May 3, 8:20 am, henryg...@gmail.com wrote:
> On May 2, 12:39 pm, "Kenneth P. Bowman" <k-bow...@null.edu> wrote:
>
>
>
>> In article
>> < c76c7777-b3ab-4645-b22a-c90be59a1...@t54g2000hsg.googlegroup s.com >,
>>   Brian Larsen <balar...@gmail.com> wrote:
>
>>> Rick,
>
>>> On this point
>
>>>> In any case, if you need a single process size to exceed 4GB, use a 64-
>>>> bit OS.
>
>>> at the risk of being a little ignorant here. I ran this test on
>>> Leopard (10.5.2) which is reported to be a 64-bit OS...
>>> http://www.apple.com/macosx/technology/64bit.html
>>> Making this an issue of having 32-bit IDL not a 32-bit OS. Right?
>
>> Yes, I'm sure I'm not the only one waiting for ITTVIS
>> to make a 64-bit version for Mac OS X.
>
>> Ken

>
> I'm waiting too...
> -henry

I asked about the timeframe for the 64bit verison of IDL on OS X when
I renewed my maintenance contract a few months ago. Apparently they
are aiming for May. We'll see how that works out.

dave

Subject: Re: memory allocation on Macs
Posted by pgrigis on Mon, 05 May 2008 20:49:40 GMT
View Forum Message <> Reply to Message

Karl wrote:
> On May 2, 1:13 pm, pgri...@gmail.com wrote:
>> Yes, you're right that I can allocate all the 7 GB (and more) in
>> different IDL
>> sessions. So there seems to be a limit indeed on how much memory one
>> single
>> IDL session (or in general , one process) can use up, but there isn't
>> a limit for
>> total usage (which, though I am sure there are a number of technical
>> reason
>> for it, seems a bit silly, after all if the system as a whole can
>> access more
>> than 4 GB, why shouldn't parts of the system be allowed to do the
>> same?)
>
> Because it is a 32-bit application. One key difference between 32-bit
> and 64-bit applications is that the pointers maintained by a 32-bit
> application are 32 bits in size, and the pointers maintained by a 64-
> bit application are 64 bits in size. This happens at compile time.
> So, your 32-bit application simply cannot address more than 4GB at a
> time due to its fundamental pointer size. Note that a 64-bit
> application will have a larger storage requirement due to the larger
> pointers.
>
> The memory management unit on the 32-bit CPU, something that you
> cannot directly access

OK, I guess I see the logic here: since the application cannot access
this,
the 4GB stands as a hard limit, and it makes more sense for the
vendors
to just move the application to 64 bits than implement fancy
techniques.

Thanks for your explanations,
Paolo

> , can address more than 4GB worth of RAM since
> it can map more than 4GB among several processes. Here, it is
> probably mapping larger chunks of memory, or pages, rather than
> individual bytes, so it isn't as hard as it sounds. But it is the MMU
> that locates the memory pages assigned to a 4GB process among the 7GB
> of installed memory and translates their physical addresses to 32-bit
> virtual addresses for the 4GB process.
>
> While there are lots of ways to emulate bigger address spaces and ways
> to fit bigger problems onto small machines, it may often be much
> easier to move to a 64-bit address space.
>
> Karl
>
>
>
>>
>> FYI, this is a Xeon machine in Mac OS X 10.4, so it is a 64 bit
>> processor
>> in a 32 bit OS running a 32 bit application.
>>
>> Anyway, thanks to all. I can cope with reading a few arrays off the
>> disk
>> from time to time.
>>
>> Ciao,
>> Paolo

Subject: Re: memory allocation on Macs
Posted by Keflavich on Thu, 26 Jun 2008 19:14:01 GMT
View Forum Message <> Reply to Message

I have some follow-up questions on this discussion....

I'm running into the errors described above, I believe:
idl(42359,0xa0281fa0) malloc: *** mmap(size=112533504) failed (error
code=12)
*** error: can't allocate region
*** set a breakpoint in malloc_error_break to debug
% Unable to allocate memory: to make array.
  Cannot allocate memory

so this appears to be a limit imposed by the OS. What I'm not

entirely clear on is, first, does that 4GB limit include virtual
memory?

Then... second... any tips on getting around gigantic memory issues?
I'm running into them using the Goddard astron library for coordinate
transformations. The big problem is (at least partly) that my very
large float arrays get converted into doubles because all of the
astron packages use doubles. There's no way to force the arrays to
stay in the smaller version, right?

Would it be possible to do something like write the necessary files to
hard disk, spawn a new IDL, have it process the data (assuming that it
doesn't independently exceed 4gb...) and write that back? Seems
complicated to me, so I don't really want to attempt it until I've
ruled out other possiblities.

Thanks,
Adam

Subject: Re: memory allocation on Macs
Posted by wlandsman on Thu, 26 Jun 2008 20:54:22 GMT
View Forum Message <> Reply to Message

> Then... second... any tips on getting around gigantic memory issues?
> I'm running into them using the Goddard astron library for coordinate
> transformations. The big problem is (at least partly) that my very
> large float arrays get converted into doubles because all of the
> astron packages use doubles. There's no way to force the arrays to
> stay in the smaller version, right?

It's somewhat odd to be carrying all your coordinates in a big
array. Usually one has a world coordinate system (e.g. in a FITS
header) from which one can compute the coordinate of every pixel.
One can then precess, rotate, or otherwise transform the coordinate
system without applying the transformation to each individual
pxiel.

But presuming you need to work with arrays of coordinates, you can
always transform the result back to float. For example, if you have
big celestial coordinates arrays, ra and dec, that you need to
transform to Galactic then use euler.pro

```
IDL> euler,1,ra,dec,glong,glat     ;Glong and glat are always output
double precision
IDL> glong=float(glong) & glat = float(glat)   ;so convert back to
float
```

Also think about whether you need to keep old variables.   For
example,

IDL> euler,1,ra,dec   ;Convert ra,dec to Galactic
IDL> glong = float(temporary(ra)) & glat = float(temporary(dec))

--Wayne

Subject: Re: memory allocation on Macs
Posted by Keflavich on Thu, 26 Jun 2008 23:43:36 GMT
View Forum Message <> Reply to Message

On Jun 26, 2:54 pm, wlandsman <wlands...@gmail.com> wrote:
>> Then... second... any tips on getting around gigantic memory issues?
>> I'm running into them using the Goddard astron library for coordinate
>> transformations. The big problem is (at least partly) that my very
>> large float arrays get converted into doubles because all of the
>> astron packages use doubles. There's no way to force the arrays to
>> stay in the smaller version, right?
>
> It's somewhat odd to be carrying all your coordinates in a big
> array.   Usually one has a world coordinate system (e.g. in a FITS
> header) from which one can compute the coordinate of every pixel.
> One can then precess, rotate, or otherwise transform the coordinate
> system without applying the transformation to each individual
> pxiel.
>
> But presuming you need to work with arrays of coordinates, you can
> always transform the result back to float.   For example, if you have
> big celestial coordinates arrays, ra and dec, that you need to
> transform to Galactic then use euler.pro
>
> IDL> euler,1,ra,dec,glong,glat   ;Glong and glat are always output
> double precision
> IDL> glong=float(glong) & glat = float(glat)   ;so convert back to
> float
>
> Also think about whether you need to keep old variables.   For
> example,
>
> IDL> euler,1,ra,dec   ;Convert ra,dec to Galactic
> IDL> glong = float(temporary(ra)) & glat = float(temporary(dec))
>
> --Wayne

Sorry, I wasn't clear: my ra/dec arrays are timestream arrays that are

used to map each data point to an image pixel. My code is crashing
WITHIN the astron routines, so converting to float before/after
doesn't help any. I think a big part of the problem is that the
astron routines copy a lot of the arrays. Euler is one place it
crashes, one of the WCS rotation programs is another.

also, isn't euler's syntax 'euler,ra,dec,glon,glat,1' ?

Thanks,
Adam

Subject: Re: memory allocation on Macs
Posted by wlandsman on Fri, 27 Jun 2008 13:40:37 GMT

View Forum Message <> Reply to Message

On Jun 26, 7:43 pm, Keflavich <keflav...@gmail.com> wrote:
  I think a big part of the problem is that the
> astron routines copy a lot of the arrays. Euler is one place it
> crashes, one of the WCS rotation programs is another.
>

Yeah, Euler is not at all concerned about memory issues.   Besides
performing all calculations in double precision, it makes sure that
the original arrays are not modified, and precomputes arrays of
trigonometric quantities (e.g. sin(ra))

I've put an udpated version of euler.pro on
http://idlastro.gsfc.nasa.gov/ftp/pro/astro/euler.pro
which is more careful about memory issues.   Let it update the input
arrays ,e.g.

```
IDL> euler, ra,dec,select=1
```

But considering that you have problems with other routines too, I'd
probably chunk your processing, so, for example, you only process 50
million coordinates at a time.

> also, isn't euler's syntax 'euler,ra,dec,glon,glat,1' ?

Yep, my fingers were typing faster than my brain was thinking...

--Wayne