Subject: Re: Memory question by a newbie
Posted by Pete[1] on Sun, 04 Aug 2002 23:33:09 GMT
View Forum Message <> Reply to Message

Hi Pat,

Now *here's* a good candidate for memory-mapping.

If you are using a Windows platform, read my recent post under "Memory headaches" about the availability of memory-mapped-file DLMs. If you are using Unix, there's Eric Korpela's VARRAY:
http://albert.ssl.berkeley.edu/~korpela/mmap/

Although you are already past the 1GB mark with your inputs alone, you have ten input arrays and that's much easier to cope with than one single, huge array. Nonetheless you are still going to have to be careful about how you go about it if you're working with Windows NT or 2000 - you could run out of suitably large chunks of address space if you aren't lucky. (I don't know how big your output is - hopefully 115MB like the others? If several hundred MB, you will probably have to do this job in stages on Windows.) Probably the best way to go about it is to run your program in as fresh and lean an IDL session as possible, and allocate or memory-map the largest arrays first (i.e., your output array and your ten inputs, before any other largish arrays that you might be using). You will also have to be very careful in the way you do your calculations. Learn the ways of IDL's TEMPORARY function and efficient array insertion (e.g., using expressions like A[0,I]=B[*,J] rather than A[*,I]=B[*,J]) to minimise temporary-variable overheads in array arithmetic. These overheads are usually impossible to avoid entirely in IDL, but they can be kept down. They can occasionally be show-stoppers when working with huge arrays if they aren't dealt with properly. (Note that TEMPORARY(MMF_VAR) will essentially invalidate a memory-mapped-file variable MMF_VAR for subsequent access, though, and you will have to re-map it to use it later. So you might want to mix and match memory mapping with normal arrays or IDL's ASSOC() and / or carefully consider the order in which you do things.)

Good luck,
Cheers
Peter Mason

"Pat" <patt@cnr.usu.edu> wrote in message
news:4928c7c9.0208021254.50d3f519@posting.google.com...
> I am a newbie to IDL and am hoping for some advice on memory issues.
> My programm reads 10 files and places the information (floating point)
> into 10 different 2D arrays. The arrays are about 5000 columns by
> 6000 rows. Each array is subject to simple mathematical manipulation

> that results in a new 2D array. So, I read 10 files and create 10 2D
> arrays. These 10 arrays are intermediate arrays that are used to
> create a final 2D array. Well, you can probably figure out that I ran
> out of memory. I read up on memory problems with IDL and it seems
> like this is the way life is in IDL. The way I solved the memory
> problem was to break up the files into sections and work with those
> sections, then read the file again only moving the file pointer to the
> end of previous section. I used the update parameter of the read
> function to create the final array, so each section is added to the
> final array file. I am wondering if there is a better work around? I
> found the IDL_MAKETEMPARRAY function but I don't understand what it
> does and am not sure if that is the answer. Any other ideas? Pat

---