Subject: Re: memory allocation for structure arrays
Posted by Peter Mason on Thu, 30 Apr 1998 07:00:00 GMT
View Forum Message <> Reply to Message

On Wed, 29 Apr 1998, Ian Sprod wrote:

> I am trying to read a pretty large data file (40Mb) into IDL. The file
> is 925,801 records, each 44 bytes long. I can describe the 44 bytes as a
> data structure and then replicate this to make a structure array (albeit
> a very large one). Then, in theory, reading in the data is a breeze.
>
> The problem is that IDL runs out of memory trying to read in the file.
> It seems that each line of the structure array is somehow requiring MORE
> than 44 bytes of memory. Poking around with top and free shows that it
> seems to be using ~312 bytes for each line instead. At this rate I can
> only read in the first ~225,000 lines of the file.
>
> Does anyone know exactly how IDL allocates memory for structures?
>
> Should I be using an associative array to do this?
>
> I am running IDL 5.0.2 on a Linux box with 128Mb of physical RAM and
> twice that of swap space.

Are you doing an ASCII (READF) or binary (READU) read? If ASCII then it might be some sort of buffer problem. Try using OPEN,BUFSIZE=0, ...

(Really assuming a binary read from here on...)

Do you get the error during the READU call? (i.e., Not during the REPLICATE call?) If so then there's something very odd going on here.
Can you read the first few (say 10) records correctly? If so, then I'd say that there might be a bug in IDL on Linux.

Type: HELP,my_struc,/STRUC to see how much actual memory is used by each instance of your struc. (My_struc being either your array or your original struc def.) Note that this includes IDL's padding bytes, and will most probably be a little more than 44 bytes (this is OK). If it's a *lot* more than 44 bytes then you have a problem in your struc def.

If your structure contains no strings then I would have expected the I/O to work with READU. With READU and WRITEU, padding bytes are handled transparently by IDL, and not stored on disk.
If it contains strings then (assuming the file was not generated by IDL) they will have a fixed amount of space in the original structure. The easiest way to accomodate this in IDL is to use BYTARRs of the required size instead of strings in your structure definition. (You can get the strings later when

you need them - STRING(a_bytarr) will find and use any NULL-termination in a_bytarr.) String structure-members in IDL are dynamically sized and it is a nightmare to do binary I/O with them.

Do not use ASSOC to try to read the file unless you know that it was written by an IDL program using ASSOC. I believe that ASSOC with structures does not have the intelligence of READU/WRITEU - it reads and writes the padding bytes from/to disk and so is incompatible with READU and WRITEU (and essentially any external program that does not know exactly how IDL has padded the structure on your particular platform).

Not very conclusive, but I hope it helped a bit.
Peter Mason
