Subject: Re: array concatenation
Posted by Karl[1] on Sat, 04 Oct 2008 01:11:19 GMT
View Forum Message <> Reply to Message

On Oct 3, 3:16 pm, lecacheux.al...@wanadoo.fr wrote:
> On 3 oct, 17:15, Joost Aan de Brugh <joost...@gmail.com> wrote:
>
>
>
>> On Oct 3, 1:19 pm, lecacheux.al...@wanadoo.fr wrote:
>
>> Hello,
>
>> Maybe it has something to do with the array descriptor. Anyway, in a
>> large group concatenation is not the most elegant way. In Matlab (a
>> language similar to IDL), you get a warning if you use such a
>> construction. It has to do with the fact that if your array grows, you
>> ask your system for more space. A safer way is to ask for enough space
>> at once.
>
>> afh = a few 100
>
>> b = BytArr(afh*1000) ; Here is where you ask for a lot of space.
>> for i=0,999 do begin
>> ... compute a = array of bytes (a few 100) ...
>> b[i*afh:(i+1)*afh-1] = a ; Now b does not grow in the loop
>> end
>
>> Or use a 2D array
>
>> b = BytArr(afh,1000) ; Here, you ask for the space again.
>> for i=0,999 do begin
>> ... compute a = array of bytes (a few 100) ...
>> b[*,i] = a ; Now b does not grow in the loop
>> end
>
>> Cheers,
>> Joost
>
>> b = Reform(b,afh*1000) ; Or b = Reform(b,N_Elements(b))
>
>> It is a bit harder if you have different 'a few 100's for each
>> iteration
>
> Thanks for your reply. I agree with you that such a programming style
> is far from ideal.
> My point is that it likely can produce some not obvious array boundary
> error (and subsequent IDL crash),

> while largest used array sizes remain far below the maximum authorized
> one.
> Or I missed something ?
> alx.

I dunno, I think that there's something else going on. IDL should fail gracefully if it runs out of memory allocating that array over and over, even if you are chewing up space and causing a lot of fragmentation. How big is the array actually getting when you crash? I just tried it on linux with an array size of 500 and looping 25,000 times with no issue at all. Are you pushing up against the max virt mem in your machine? If you are indeed paging, Windows can get a little unstable under excessive paging.

The COMPILE_OPT IDL2 note is interesting too. Recall that if you don't specify this, integers are 16-bit, and 32-bit if you do. What does your code specified by "... compute a = array of bytes (a few 100) ... " do? Does it call a custom DLM? Is there something that would break or overflow if ints are 16-bit? Still, IDL checks array accesses at run time, so a bad array index shouldn't cause a crash. It may be worth taking a closer look at why IDL2 makes a difference.
