Subject: Re: Dealing with Large data arrays, reducing memory and ASSOC
Posted by Ambrosia_Everlovely on Fri, 15 Jun 2007 07:26:09 GMT
View Forum Message <> Reply to Message

On Jun 14, 8:52 pm, Kenneth Bowman <k-bow...@tamu.edu> wrote:
> In article <1181828486.257277.182...@q19g2000prn.googlegroups.com>,
>
>
>
> bill.d...@gmail.com wrote:
>> On Jun 14, 8:33 am, Ambrosia_Everlovely
>> <ambrosia_everlov...@hotmail.com> wrote:
>>> Hi,
>>> I have a fairly large datacube, DC(x,y,t)=DC(512,512,2048) and I want
>>> to perform an FFT in the t direction. Now I can do,
>>> FFTDC=fft(DC,-1,dim=3) which takes an excessive amount of memory (19 G
>>> + 50 G virtual) and slows the whole system down.
>>> Since this must be a fairly common practice amongst astronomers, can
>>> anyone provide - or link to - a small IDL algorithm which will allow
>>> me to use ASSOC or reduce the memory in some way? I have also tried
>>> TEMPORARY, but this doesn't seem to help at all.
>
>>> Thankyou!!!!
>
>> Assuming you are using single precision, you can limit memory needed
>> to about 6GB with
>
>> fftdc = complexarr(512,512,2048)
>> for i=0,511 do for j=0,511 do fftdc[i,j,0] = fft(dc[i,j,*],-1)
>
>> this should help if your machine has more than 6GB for you to use.
>
> I don't think this will work as written. The trick of zero-subscripting
> on the LHS of an assignment works for the leading dimensions only.
>
> IDL> x = findgen(4,4)
> IDL> print, x
> 0.00000 1.00000 2.00000 3.00000
> 4.00000 5.00000 6.00000 7.00000
> 8.00000 9.00000 10.0000 11.0000
> 12.0000 13.0000 14.0000 15.0000
> IDL> x[0,2] = replicate(99.0, 4)
> IDL> print, x
> 0.00000 1.00000 2.00000 3.00000
> 4.00000 5.00000 6.00000 7.00000
> 99.0000 99.0000 99.0000 99.0000
> 12.0000 13.0000 14.0000 15.0000
>

> If you try this with a trailing dimension you get this
>
> IDL> x = findgen(4,4)
> IDL> x[2,0] = replicate(99.0, 4)
> % Out of range subscript encountered: X.
> % Execution halted at: $MAIN$
>
> To make your expression work, you would have to write
>
> fftdc[i,j,*] = fft(dc[i,j,*],-1)
>
> which results in some performance penalty.
>
> Ken Bowman

Yes, this is what I have done combined with the TEMPORARY function, it has reduced the memory use a fraction. I was not so concerned with the time it was taking, but the memory it was using (according to sys admin, swapping memory in and out), and thus slowing everyone else down. I have also taken this out of a loop and am calculating it individually - but this is clearly not ideal. It appears that the memory "builds up" with each loop. This may be the real root of my problems - is there a way to "clear" all the variables after each loop? DELVAR only works at the MAINS level.....

You guys have been great, thanks.
