Subject: Re: idl parallel processing
Posted by wlandsman on Mon, 22 May 2017 19:06:41 GMT
View Forum Message <> Reply to Message

As I mentioned, you can only use the .hasvalue static method if you have IDL 8.4 or later. If you have an earlier version of IDL you can still get some speed improvement using the following function.

```
function hasvalue,array, value
; Determine if scalar value is present in array
; Emulates the .hasvalue static method introduced in IDL 8.4
return,~array_equal( array NE value, 1b)
end
```

and use it like this.

```
pro computation,data=data,siteN=siteN,result
N = N_elements(siteN)
result=fltarr(N,/nozero)
FOR i= 0,N-1  do result[i] = hasvalue(data, siteN[i])
END
```

I find hasvalue() is a factor of two faster than using WHERE(), but a factor of five slower than using the .hasvalue method. The actual speed ratios depend on the size of the arrays, and the fraction of time that .hasvalue is true.

On Sunday, May 21, 2017 at 10:59:52 PM UTC-4, Sium T wrote:
> On Friday, May 19, 2017 at 10:59:07 AM UTC-4, wlandsman wrote:
>> Yes, you can use the IDL Bridge for this. But if you have IDL 8.4 or later, then more valuable would be using the .HASVALUE() static method. Your code would then be
>>
>> result= bytarr(n_elements(siteN))
>> FOR i= 0,n_elements(siteN)-1 do result[i] = data.hasvalue(siteN[i])
>>
>> The reasons this is much faster are (1) you don't need to compute the output vector of WHERE(). All you care about is whether the siteN[i] value is present in the data array-- you don't care where it is. And (2) the .hasvalue() method will return as soon as it finds a single case where the siteN[i] value is present, so you skip having to search the entire data array
>>
>> --Wayne
>>
>> On Thursday, May 18, 2017 at 6:05:51 PM UTC-4, Sium T wrote:
>>> Hello,
>>>
>>> I have a procedure below. It want to call my procedure in my main program and do parallel processing on the do loop.
>>>
>>> How can use the IDL_Bridge . Any suggestion

```
>>>
>>>  pro computation,data=data,siteN=siteN,result
>>>
>>>  result=fltarr(n_elements(siteN))
>>>
>>>    FOR i= 0,n_elements(siteN)-1  do begin
>>>      y=where(data eq siteN(i))
>>>      if y(0) ge 0 then begin
>>>        result(i)=1
>>>      endif else begin
>>>        result(i)=0
>>>      endelse
>>>    ENDFOR
>>>
>>>  end
>
>  Thanks Wayne
>
>  I tried your method
>  result= bytarr(n_elements(siteN))
>  FOR i= 0,n_elements(siteN)-1  do result[i] = data.hasvalue(siteN[i])
>
>  However, I got this error message.
>
>  Object reference type required in this context:
```