Subject: idl parallel processing
Posted by siumtesfai on Thu, 18 May 2017 22:05:47 GMT
View Forum Message <> Reply to Message

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
```

Subject: Re: idl parallel processing
Posted by wlandsman on Fri, 19 May 2017 14:59:05 GMT
View Forum Message <> Reply to Message

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

Subject: Re: idl parallel processing
Posted by siumtesfai on Mon, 22 May 2017 02:59:50 GMT
View Forum Message <> Reply to Message

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:

Subject: Re: idl parallel processing
Posted by Helder Marchetto on Mon, 22 May 2017 09:35:47 GMT
View Forum Message <> Reply to Message

On Monday, May 22, 2017 at 4:59:52 AM UTC+2, 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:

Hi,
what do you get if you type at the command line:
help, !version

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

On Monday, May 22, 2017 at 5:35:49 AM UTC-4, Helder wrote:
> On Monday, May 22, 2017 at 4:59:52 AM UTC+2, 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:
>
> Hi,
> what do you get if you type at the command line:
> help, !version

I have IDL version 8.2.3 . HasValue works with version 8.4 or above.

So I need to use idl_bridge. But it becomes challenging to me

Here is my trial code .
First I have this procedure. It takes for ever to compute Shourly result.

Can you help with how to call this procedure in idl_idlbridge ?

```
;=============================================
pro program1,Rdata,edata,Shourly


  StateN=reform(edata(0,*))
  CountyN=reform(edata(1,*))
  siteN=reform(edata(2,*))
;=======================
scode=reform(Rdata(0,*))
ccode=reform(Rdata(1,*))
snum=reform(Rdata(2,*))
year=reform(Rdata(3,*))
month=reform(Rdata(4,*))
day=reform(Rdata(5,*))
hour=reform(Rdata(6,*))
lats=reform(Rdata(7,*))
lons=reform(Rdata(8,*))
;==============================
Shourly=fltarr(n_elements(siteN),12,31,24)

for s=0,n_elements(stateN)-1 do begin

  z=where(scode eq fix(StateN(s)) and ccode eq fix(CountyN(s)) and snum eq fix(siteN(s)))

  if z(0) ge 0 then begin
    data2=Rdata(*,z)

    FOR mn=1,12 do begin
      FOR dy=1,31 do begin
        FOR hr=0,23 do begin

          b=where(month eq mn and day eq dy and hour eq hr)

          if b(0) ge 0 then begin
            value=data2(9,b)
            Shourly(s,mn-1,dy-1,hr)=value(0)
          endif else begin
            Shourly(s,mn-1,dy-1,hr)=-9999.0
          endelse

        ENDFOR
      ENDFOR
    ENDFOR
```

```
  endif else begin
    Shourly(s,*,*,*)=-9999.0
  endelse

endfor

end

;=======================================
```

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:

Subject: Re: idl parallel processing
Posted by siumtesfai on Mon, 22 May 2017 19:34:05 GMT
View Forum Message <> Reply to Message

On Monday, May 22, 2017 at 3:06:45 PM UTC-4, wlandsman wrote:
> 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:

Thank you for being patient with me

I tried but it says :
Variable is undefined: HASVALUE

It seems there is no Hasvalue in IDL 8.2.3 . correct me if I am wrong

Best regards

Subject: Re: idl parallel processing
Posted by siumtesfai on Mon, 22 May 2017 19:40:04 GMT
View Forum Message <> Reply to Message

On Monday, May 22, 2017 at 3:06:45 PM UTC-4, wlandsman wrote:
> 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:

Oh I saw your function Hasvalue

Thanks

Let me know if you can help me do the calculation using IDL_IDLbridge.

Best wishes

Subject: Re: idl parallel processing
Posted by natha on Tue, 23 May 2017 12:29:51 GMT
View Forum Message <> Reply to Message

Hi guys,

Please remember that some time ago I developed a library to manage parallel processing under IDL. It is very easy to use and I strongly encourage the community to take a look at it and use it.

https://github.com/bernatp3rs/idl_cpu_pm/wiki

Best >:D

Subject: Re: idl parallel processing
Posted by Markus Schmassmann on Tue, 23 May 2017 12:59:27 GMT
View Forum Message <> Reply to Message

On 05/22/2017 09:06 PM, Sium T wrote:
> On Monday, May 22, 2017 at 5:35:49 AM UTC-4, Helder wrote:
>> On Monday, May 22, 2017 at 4:59:52 AM UTC+2, 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:
>>>>
>>>> Hi, what do you get if you type at the command line: help,
>>>> !version
>>>
>>> I have IDL version 8.2.3 . HasValue works with version 8.4 or
>>> above.
>>>
>>> So I need to use idl_bridge. But it becomes challenging to me

```
>
>  Here is my trial code . First I have this procedure. It takes for
>  ever to compute Shourly result.
>
>  Can you help with how to call this procedure in idl_idlbridge ?
>
>
>
>  ;=============================================
>  pro program1,Rdata,edata,Shourly
>
>
>  StateN=reform(edata(0,*))
> CountyN=reform(edata(1,*))
>  siteN=reform(edata(2,*))
> ;=======================
>  scode=reform(Rdata(0,*))
> ccode=reform(Rdata(1,*))
>  snum=reform(Rdata(2,*))
> year=reform(Rdata(3,*))
>  month=reform(Rdata(4,*))
> day=reform(Rdata(5,*))
>  hour=reform(Rdata(6,*))
> lats=reform(Rdata(7,*))
>  lons=reform(Rdata(8,*))
> ;================================
>  Shourly=fltarr(n_elements(siteN),12,31,24)
>
>  for s=0,n_elements(stateN)-1 do begin
>
>  z=where(scode eq fix(StateN(s)) and ccode eq fix(CountyN(s)) and snum eq fix(siteN(s)))
>
>  if z(0) ge 0 then begin
>   data2=Rdata(*,z)
>
>     FOR mn=1,12 do begin
>   FOR dy=1,31 do begin
>   FOR hr=0,23 do begin
>
>           b=where(month eq mn and day eq dy and hour eq hr)
>
>           if b(0) ge 0 then begin
>   value=data2(9,b)
>             Shourly(s,mn-1,dy-1,hr)=value(0)
>   endif else begin
>             Shourly(s,mn-1,dy-1,hr)=-9999.0
>   endelse
>
```

```
>         ENDFOR
>   ENDFOR
>     ENDFOR
>
>  endif else begin
>   Shourly(s,*,*,*)=-9999.0
>  endelse
>
>  endfor
>
>  end
>
>  ;=======================================
```

I have no experience with the IDL_IDLbridge, but maybe you can speed up your processing using HISTOGRAM and avoid most of the looping.
Below I did it (without testing) for your trial code, for more see:
http://www.idlcoyote.com/tips/histogram_tutorial.html

```
;================================
shourly=fltarr(24,31,12,n_elements(siteN))
for s=0,n_elements(stateN)-1 do begin

   z=where( scode eq fix(StateN(s)) and ccode eq fix(CountyN(s)) and $
            snum eq fix(siteN(s)),cnt)

   if cnt ne 0 then begin
     data2=Rdata[*,z]

     h=histogram(((month-1)*31+(day-1))*24+hour, $
        min=0,max=12l*31*24-1,bin=1,reverse_indices=ri)
      Shourly[*,*,*,s]=reform(data2[9,ri[ri[0:(12l*31*24-1)]]],[24 ,31,12])
     w=where(h eq 0,cnt2)
     if cnt2 ne 0 then Shourly[12l*31*24*s+w]=-9999.0

   endif else begin
     Shourly[*,*,*,s]=-9999.0
   endelse

endfor
Shourly=transpose(Shourly,[3,2,1,0])
;=======================================
```

You probably could eliminate the outer loop as well, but that would be a bit more complicated.

good luck, I hope this helps, Markus

Subject: Re: idl parallel processing
Posted by siumtesfai on Wed, 24 May 2017 22:20:20 GMT
View Forum Message <> Reply to Message

On Tuesday, May 23, 2017 at 8:59:31 AM UTC-4, Markus Schmassmann wrote:
> On 05/22/2017 09:06 PM, Sium T wrote:
>> On Monday, May 22, 2017 at 5:35:49 AM UTC-4, Helder wrote:
>>> On Monday, May 22, 2017 at 4:59:52 AM UTC+2, 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:
>>>
>>> Hi, what do you get if you type at the command line: help,
>>> !version
>>
>> I have IDL version 8.2.3 . HasValue works with version 8.4 or
>> above.
>>
>> So I need to use idl_bridge. But it becomes challenging to me
>>
>> Here is my trial code . First I have this procedure. It takes for
>> ever to compute Shourly result.
>>
>> Can you help with how to call this procedure in idl_idlbridge ?
>>
>>
>>
>> ;===
>> pro program1,Rdata,edata,Shourly
>>
>>
>> StateN=reform(edata(0,*))
>> CountyN=reform(edata(1,*))
>> siteN=reform(edata(2,*))
>> ;=======================
>> scode=reform(Rdata(0,*))
>> ccode=reform(Rdata(1,*))
>> snum=reform(Rdata(2,*))
>> year=reform(Rdata(3,*))
>> month=reform(Rdata(4,*))
>> day=reform(Rdata(5,*))
>> hour=reform(Rdata(6,*))
>> lats=reform(Rdata(7,*))
>> lons=reform(Rdata(8,*))
>> ;==============================
>> Shourly=fltarr(n_elements(siteN),12,31,24)
>>
>> for s=0,n_elements(stateN)-1 do begin
>>
>> z=where(scode eq fix(StateN(s)) and ccode eq fix(CountyN(s)) and snum eq fix(siteN(s)))

```
>>
>>  if z(0) ge 0 then begin
>>   data2=Rdata(*,z)
>>
>>     FOR mn=1,12 do begin
>>   FOR dy=1,31 do begin
>>   FOR hr=0,23 do begin
>>
>>           b=where(month eq mn and day eq dy and hour eq hr)
>>
>>           if b(0) ge 0 then begin
>>   value=data2(9,b)
>>             Shourly(s,mn-1,dy-1,hr)=value(0)
>>   endif else begin
>>             Shourly(s,mn-1,dy-1,hr)=-9999.0
>>   endelse
>>
>>         ENDFOR
>>   ENDFOR
>>     ENDFOR
>>
>>  endif else begin
>>   Shourly(s,*,*,*)=-9999.0
>>  endelse
>>
>>  endfor
>>
>>  end
>>
>>  ;=======================================
>  I have no experience with the IDL_IDLbridge, but maybe you can speed up
>  your processing using HISTOGRAM and avoid most of the looping.
>  Below I did it (without testing) for your trial code, for more see:
>  http://www.idlcoyote.com/tips/histogram_tutorial.html
>
>  ;================================
>  shourly=fltarr(24,31,12,n_elements(siteN))
>  for s=0,n_elements(stateN)-1 do begin
>
>     z=where( scode eq fix(StateN(s)) and ccode eq fix(CountyN(s)) and $
>              snum eq fix(siteN(s)),cnt)
>
>     if cnt ne 0 then begin
>       data2=Rdata[*,z]
>
>       h=histogram(((month-1)*31+(day-1))*24+hour, $
>          min=0,max=12l*31*24-1,bin=1,reverse_indices=ri)
>        Shourly[*,*,*,s]=reform(data2[9,ri[ri[0:(12l*31*24-1)]]],[24 ,31,12])
```

```
>       w=where(h eq 0,cnt2)
>       if cnt2 ne 0 then Shourly[12l*31*24*s+w]=-9999.0
>
>     endif else begin
>       Shourly[*,*,*,s]=-9999.0
>     endelse
>
>  endfor
>  Shourly=transpose(Shourly,[3,2,1,0])
>  ;=======================================
>
>  You probably could eliminate the outer loop as well, but that would be a
>  bit more complicated.
>
>  good luck, I hope this helps,     Markus
```

thanks Markus
