Subject: Re: Threads in IDL 7.0
Posted by Vince Hradil on Tue, 28 Oct 2008 14:31:21 GMT
View Forum Message <> Reply to Message

```
On Oct 27, 12:26 pm, Heinz Stege <public.215....@arcor.de> wrote:
> On Mon, 27 Oct 2008 16:44:29 +0100, Bernhard Reinhardt wrote:
>> Well, I wasn't really precise. I'm not doing this on a 1d-array but on a
>> 4d-array, where 2 dimensions are time and 2 dimensions are space. I try
>> to filter special events in time and count those on a 2d-map. Here's the
>> code:
>
       for i = 0, N ELEMENTS(STRUC.data[*,0,0,0])-1 do begin
>>
        for j = 0, N_ELEMENTS(STRUC.data[0,*,0,0])-1 do begin
>>
         indices = Where(STRUC.data[i,j,*,*] GE 150., count)
>>
         freq [i,j]=count
>>
        endfor
>>
       endfor
>>
>> Although the array data is quite big "where" only get's a small portion
>> to see of it. So thread-pool isn't invoked. => CPU-Usage still 50%
>> I also asked some more IDL-experienced colleagues about generating
>> threads manually but they also didn't know about anything like that :(
>> BUT using your method still brought me a gain of 3.6 times faster
>> execution :)
>> regards
>
>> Bernhard
  Please try the following command:
>
    freq=total(total(STRUC.data ge 150.,4,/integer),3,/integer)
>
>
> The IDL manual says, that TOTAL makes use of IDL's thread pool. And
 there is no for-loop needed anymore...
> HTH, Heinz
At the risk of repeating myself (I tried to post this earlier, but it
hasn't shown up?):
Which is faster - (1)TRANSPOSE and TOTAL over leading dims or (2)
TOTAL over trailing dims
Here's my quick (un-scientific) test:
```