Subject: Re: Threads in IDL 7.0
Posted by Allan Whiteford on Mon, 27 Oct 2008 17:23:23 GMT
View Forum Message <> Reply to Message

>
> On Mon, 27 Oct 2008, Bernhard Reinhardt wrote:
>
>> David Fanning wrote:
>>
>>> Bernhard Reinhardt writes:
>>>
>>>> I have have read the Chapter "Multithreading in IDL" in the
>>> IDL-Help. As > far as I can see some IDL functions use threads.
>>>> > Basically I want to count elements of a big array that exceed a
>>> given > number.
>>>> > count=0
>>>> for i = 0, 200000 do begin
>>>> if array[i] ge 10. then count++
>>>> endfor
>>>
>>> How about something like this:
>>>
>>> indices = Where(array GE 10, count)
>>
>>

>> 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.

>>
>> 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)
>
>
> You are accessing elements in the wrong order, resulting in cache misses
> if struc.data is greater than the CPU data cache.
>
>> freq [i,j]=count
>> endfor
>> endfor
>>
>
> Try this:
>
> d=transpose(struc.data, [2,3,0,1])
> dims=size(d, /dim)

```
>  for i = 0, dims[2]-1 do begin
>    for j = 0, dims[3]-1 do begin
>      indices = Where(d[*,*,i,j] GE 150., count)
>      freq [i,j]=count
>    endfor
>  endfor
>
>
>  regards,
>  lajos
```

Or this:

```
freq = fix(total(total(transpose(struc.data,[2,3,0,1]) ge 150.,1),1))
```

which should go faster because of the lack of loops.

Thanks,

Allan
