Subject: Re: Shortcut needed
Posted by davidf on Sun, 16 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

E. Scott Claflin (sclaflin@netcom.com) writes in response
to a Daniel Williams conundrum:

```
>    for i=0,n_max do begin
>      if (r[i+1] gt r[i]) then begin
>        y[i] = total(a[r[r[i]:r[i+1]-1]].x)
>      endif
>    endfor
```

There are days when I feel really old. :-(

I'm sticking with object graphics. They may be
incomprehensible, but at least I can read my code. :-)

Cheers,

David

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Shortcut needed
Posted by sclaflin on Sun, 16 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

Daniel Williams (williams@idunn.srl.caltech.edu) wrote:

: Hi fellow IDL-users,

: I have a program which runs slower than I like, and wonder if someone
: could help me speed it up a bit.

: My data is an array of structures, call it 'a'. One element of the
: structure, call it 'n' is an integer, another is a quantity which I
: want to add up, call it 'x'.

: There are about 100000 elements in this array. The integer n, marks
: data as being of a certain type. I want to make a new array, call it
: 'y', which has as its nth element the sum of all the x's which have a

: n-value of n. I currently do this as follows,

: for n=0, n_max do begin
:   y[n] = total(a[where(a.n eq n)].x)
: endfor

: The trouble is that the loop is slow, as for loops always are. Does
: anyone out there know a better way to do this kind of sorting?

Some weeks ago the virtues of the histogram function for sorting
problems were discussed in this news group. I haven't tested the
the following solution, but if you have a really slow process,
it might be worth a try. There is still a "for" loop, but the
"where" function does not have to be called each time through the loop.

```
  h = histogram(a.n, Min=0, Max=n_max, Reverse_Indices=r)
  h = 0b
  y = intarr(n_max + 1)

  for i=0,n_max do begin
    if (r[i+1] gt r[i]) then begin
      y[i] = total(a[r[r[i]:r[i+1]-1]].x)
    endif
  endfor
```

Subject: Re: Shortcut needed
Posted by Martin Schultz on Mon, 17 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

Daniel Williams wrote:
>
> Hi fellow IDL-users,
>
> I have a program which runs slower than I like, and wonder if someone
> could help me speed it up a bit.
>
> My data is an array of structures, call it 'a'. One element of the
> structure, call it 'n' is an integer, another is a quantity which I
> want to add up, call it 'x'.
>
> There are about 100000 elements in this array. The integer n, marks
> data as being of a certain type. I want to make a new array, call it
> 'y', which has as its nth element the sum of all the x's which have a
> n-value of n. I currently do this as follows,
>
> for n=0, n_max do begin

> y[n] = total(a[where(a.n eq n)].x)
> endfor
>
> The trouble is that the loop is slow, as for loops always are. Does
> anyone out there know a better way to do this kind of sorting?
>
> Thanks,
> Daniel Williams
> --
> +---+
> | Daniel L. Williams | Email: williams@srl.caltech.edu |
> | Space Radiation Lab | Tel: 626/395-6634 |
> | Caltech 220-47 | Fax: 626/449-8676 |
> | Pasadena, Ca 91125 | |
> | WWW: www.srl.caltech.edu/personnel/williams.html |
> +---+

another option may be to sort the data and use the uniq function:
```
a=a(sort(a.n))
ua=uniq(a.n)
```
This would give you an array of indices of the first occurence of
all n values and you could then loop like
```
for i=0,n_elements(ua)-2 do tot(i)=total(a.value(ua(i):ua(i+1)))
```

Haven't tried it, but should work - don't know how fast though.

Martin.

--
-- -------
Dr. Martin Schultz
Department for Earth&Planetary Sciences, Harvard University
186 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu
IDL-homepage: http://www-as.harvard.edu/people/staff/mgs/idl/
-- -------