Subject: Re: Need help to optimize for speed
Posted by John-David T. Smith on Tue, 16 May 2000 07:00:00 GMT
View Forum Message <> Reply to Message

From way of Hamid wrote:

```
>
>  Hi,
>
>  I have this for loop that searches for a particular pattern (roi).
>  This search takes for ever.
>  Does anyone knows how to optimize this code for speed.
>
>  Any suggestions ?
>
>  Thanks.
>
>  Hamid
>
>  size1 = size(bscl) ;;; is  1000 x 1000
>  size2 = size(roi) ;;;;   is about 50 x 50
>
>  print, 'Size of bscl is ', size1
>  print, 'Size of roi is ', size2
>
>  a = reform(bscl,size1(4))
>  b = reform(roi,size2(4))
>  nmatch = 0
>
>  print, 'searching'
>  for column = 0, size1(1) -1 do begin
>   for row = 0, size1(2) -1 do begin  ; num
>       print,'row and col', row, col
>    sample = reform(extrac(bscl,column,row,size2(1),size2(2)),size2(4))
>    IF (array_match(sample,b)) THEN begin
>            nmatch = nmatch +1 ;;; length has to match
>            print, '------------------------'
>            print,'We have a match at row of', row, ' and colof', column
>            print, 'The centriod is at ', row + size2(1)/2 , column +
>  size2(2)/2
>            print, extrac(bscl,column,row,size2(1),size2(2))
>            print, '------------------------'
>            print, b
>           end
>   end
>  end
>  print, 'Number of matches', nmatch
>  print, ' success '
```

We would really have to know what all those functions like extrac and array_match do. If your intention is to look for a given subarray "match" anywhere within a larger array, one suggestion would be:

```
IDL> wh=where(abs(convol(array,match)-total(match^2)) le eps,cnt)
```

This will find the center positions of places in array where match also occurs (if match's size isn't odd, you'll have to offset down and right by one). I use eps, a small number (like .001), depending on your data to protect against roundoff. You could also do things in double precision and just use an equality test like:

```
IDL> wh=where(convol(array,match) eq total(match^2),cnt)
```

e.g.

```
IDL> a=randomu(sd,1000,1000)
IDL> m=randomu(sd,50,50)
IDL> a[randomu(sd)*1000,randomu(sd)*1000]=m
IDL> print,where(abs(convol(a,m) - total(m^2)) le .001)
     166122
```

which takes about 30 seconds on my machine (don't know what your definition of forever is). This obviously supposes that the convolution value is a unique signature of the match's presence, which isn't strictly true, but it would be an unusual conspiracy otherwise, unless your data is highly degenerate (0's 1's and 2's say) (and you can always check the presumably small list of results to find the actual matches). A real "boolean" convolve would do the trick but alas we have no builtin for that... though it would be trivial to implement in C, and wickedly faster than what you have.

JD

--

J.D. Smith |*| WORK: (607) 255-5842
Cornell University Dept. of Astronomy |*| (607) 255-6263
304 Space Sciences Bldg. |*| FAX: (607) 255-5875
Ithaca, NY 14853 |*|