Subject: Re: Extract Array positions for a set of Values
Posted by wlandsman on Wed, 09 Mar 2011 11:08:05 GMT
View Forum Message <> Reply to Message

You might try match2.pro from http://idlastro.gsfc.nasa.gov/ftp/pro/misc/match2.pro though it is a bit of overkill.

(You first have to REFORM to a 1-d array)

```
IDL> match2,reform(a,21),b,suba,subb

IDL> print,reform(suba,7,3)
       0       0       0       0      -1      -1      -1
       0       0       0       0      -1      -1       1
      -1      -1      -1      -1       2      -1       1
```

The output is set to -1 where there is no match in the vector B.

```
IDL> a[where(suba ge 0)] = 99
IDL> print,a
      99      99      99      99       0       0       0
      99      99      99      99       0       0      99
       0       0       0       0      99       0      99
```

Subject: Re: Extract Array positions for a set of Values
Posted by Jeremy Bailin on Wed, 09 Mar 2011 14:37:33 GMT
View Forum Message <> Reply to Message

On Wednesday, March 9, 2011 5:48:42 AM UTC-5, Paul Magdon wrote:
> Hi,
> have a quite simple problem for which I can find a fas solution:
>
> 1.) I have an IntArray A (e.g a result from LABEL_REGION)
>
> 1 1 1 1 0 0 0
> 1 1 1 1 0 0 2
> 0 0 0 0 9 0 2
>
> 2.) I have a vector B with Integers (e.g. 1,2,9)
>
> Now I want to extract the positions of B in A and set all values in A which are included in B to let's say 99. How can I do this without a loop?
> I tested HISTOGRAM(,REVERSE_INDICES) but as B is not consecutive (e.g 1,2,3,4) I can't find a solution.
>
> Cheers Paul

I'd try remapping the values so that B is consecutive, and then use value_locate (come on, you *knew* it was coming...) to figure out if it's in B or not.

```
b_sorted = b[SORT(b)]
locations = WHERE(b_sorted[VALUE_LOCATE(b_sorted, a)] EQ a, nlocations)
IF nlocations GT 0 THEN a[locations]=99
```

-Jeremy.

Subject: Re: Extract Array positions for a set of Values
Posted by Gray on Wed, 09 Mar 2011 21:29:50 GMT
View Forum Message <> Reply to Message

On Mar 9, 5:48 am, Paul Magdon <paulmag...@yahoo.de> wrote:
> Hi,
> have a quite simple problem for which I can find a fas solution:
>
> 1.) I have an IntArray A (e.g a result from LABEL_REGION)
>
> 1 1 1 1 0 0 0
> 1 1 1 1 0 0 2
> 0 0 0 0 9 0 2
>
> 2.) I have a vector B with Integers (e.g. 1,2,9)
>
> Now I want to extract the positions of B in A and set all values in A which are included in B to let's say 99. How can I do this without a loop?
> I tested HISTOGRAM(,REVERSE_INDICES) but as B is not consecutive (e.g 1,2,3,4) I can't find a solution.
>
> Cheers Paul

Here's a solution that uses a FOR-loop and histogram:

```
H = histogram(A,min=0,max=max(B),reverse_indices=ri)
for i=0,n_elements(B)-1 do begin
  if H[B[i]] eq 0 then continue
  A[ri[ri[B[i]]:ri[B[i]+1]-1]] = 99
endfor
```

Who cares if B isn't consecutive? Just use it to index the histogram (and the reverse_indices array), so you only have to loop over B. I would remove duplicate values, if any, from B beforehand to save redundant iterations.

Subject: Re: Extract Array positions for a set of Values
Posted by Jeremy Bailin on Thu, 10 Mar 2011 13:49:41 GMT
View Forum Message <> Reply to Message

So I tried a little time test of these 3 solutions. Here are the results for one particular set - they vary slightly depending on how big A and B are relative to each other, but the general pattern holds. Code follows at the end:

```
IDL> .run tester
% Compiled module: $MAIN$.
VALUE_LOCATE:       0.43740702
VALUE_LOCATE pre-sorted:       0.43772793
HISTOGRAM:       0.10194707
MATCH2:       2.5133259
```

(I pre-compiled match2 before the test)

So the conclusion, not surprisingly, is that histogram kicks ass. ;-) Note also that the time in the value_locate solution is essentially all in the value_locate part, not in the sorting step.

-Jeremy.

```
stride=5
blength = 1000
adimen = [2000,2000]
seed=1L

b = stride*sort(randomu(seed,blength))
a = floor(stride*blength*randomu(seed, adimen))

; case 1: VALUE_LOCATE
t1=systime(/sec)
b_sorted = b[sort(b)]
locations = where(b_sorted[value_locate(b_sorted, a)] eq a, nlocations)
if nlocations gt 0 then a[locations]=99
t2=systime(/sec)

b = stride*sort(randomu(seed,blength))
a = floor(stride*blength*randomu(seed, adimen))

; case 2: HISTOGRAM
t3=systime(/sec)
H = histogram(A,min=0,max=max(B),reverse_indices=ri)
for i=0,n_elements(B)-1 do begin
  if H[B[i]] eq 0 then continue
  A[ri[ri[B[i]]:ri[B[i]+1]-1]] = 99
endfor
```

```
t4=systime(/sec)

b = stride*sort(randomu(seed,blength))
a = floor(stride*blength*randomu(seed, adimen))

; case 3: MATCH2
t5=systime(/sec)
match2, reform(a, n_elements(a)), b, suba, subb
locations = where(suba ge 0, nlocations)
if nlocations gt 0 then a[locations]=99
t6=systime(/sec)

b = stride*lindgen(blength)
a = floor(stride*blength*randomu(seed, adimen))

; case 4: VALUE_LOCATE pre-sorted
t7=systime(/sec)
locations = where(b[value_locate(b, a)] eq a, nlocations)
if nlocations gt 0 then a[locations]=99
t8=systime(/sec)


print, 'VALUE_LOCATE: ',t2-t1
print, 'VALUE_LOCATE pre-sorted: ',t8-t7
print, 'HISTOGRAM: ',t4-t3
print, 'MATCH2: ',t6-t5

end
```

Subject: Re: Extract Array positions for a set of Values
Posted by wlandsman on Thu, 10 Mar 2011 14:54:57 GMT
View Forum Message <> Reply to Message

On Thursday, March 10, 2011 8:49:41 AM UTC-5, Jeremy Bailin wrote:

> So the conclusion, not surprisingly, is that histogram kicks ass. ;-) Note also that the time in the value_locate solution is essentially all in the value_locate part, not in the sorting step.

Yes, histogram is will suited for this problem. HISTOGRAM is not so good when the data values cover a large range say 0 - 1000000.

MATCH2 is overkill because it doesn't assume the values are integers, doesn't assume the values in the B vector are unique, and supplies matching indices for all elements in both A and B.

--Wayne
