Subject: Re: fast array comparison
Posted by David Fanning on Sun, 08 Dec 2002 17:45:00 GMT
View Forum Message <> Reply to Message

Sean Raffuse (sean@me.wustl.edu) writes:

> Let's say I have two arrays.
>
> requested_array = [5,6,7,8,9,10]
> available_array = [3,7,8,9,12,13,16]
>
> What is the absolute fastest way to determine the indices of available_array
> that contain values in requested_array? The indices need not match. i.e.,
> if the two arrays above were used, I would like to return index=[1,2,3]
> because the requested values 7, 8 and 9 are in the available array.

The absolute fastest way MUST involve histograms, so
I maintain with a great deal of confidence (say, in the 40-50
percent range) that this is the fastest possible algorithm:

```
 ********************************************************** ************
FUNCTION SetIntersection, a, b, Indices=indices
minab = Min(a, Max=maxa) > Min(b, Max=maxb) ;Only need intersection of
ranges
maxab = maxa < maxb

  ; If either set is empty, or ranges don't intersect: result = NULL.

IF maxab LT minab OR maxab LT 0 THEN RETURN, -1
r = Where((Histogram(a, Min=minab, Max=maxab) NE 0) AND  $
        (Histogram(b, Min=minab, Max=maxab) NE 0), count)
IF Arg_Present(indices) THEN $
   indices = Where((Histogram(a, Min=minab, Max=maxab) NE 0))
IF count EQ 0 THEN RETURN, -1 ELSE RETURN, r + minab
END
 ********************************************************** ************
```

Use it like this:

```
  IDL> request_array = [5,6,7,8,9,10]
  IDL> avail_array = [3,7,8,9,12,13,16]
  IDL> int = setintersection(avail_array, request_array, Indices=i)
  IDL> print, int, i
        7       8       9
        2       3       4
```

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: fast array comparison
Posted by Sean Raffuse on Sun, 08 Dec 2002 20:32:07 GMT
View Forum Message <> Reply to Message

David,

Thanks for this. It works almost perfectly. I am a little confused though.
It seems that the indices keyword returns the indices of the requested array
and not the available array.

"David Fanning" <david@dfanning.com> wrote in message
news:MPG.185d3307aad96bee989a51@news.frii.com...
> Sean Raffuse (sean@me.wustl.edu) writes:
>
>> Let's say I have two arrays.
>>
>> requested_array = [5,6,7,8,9,10]
>> available_array = [3,7,8,9,12,13,16]
>>
>> What is the absolute fastest way to determine the indices of
available_array
>> that contain values in requested_array? The indices need not match.
i.e.,
>> if the two arrays above were used, I would like to return index=[1,2,3]
>> because the requested values 7, 8 and 9 are in the available array.
>
> The absolute fastest way MUST involve histograms, so
> I maintain with a great deal of confidence (say, in the 40-50
> percent range) that this is the fastest possible algorithm:
>
> ********** **********
> FUNCTION SetIntersection, a, b, Indices=indices
> minab = Min(a, Max=maxa) > Min(b, Max=maxb) ;Only need intersection of
> ranges
> maxab = maxa < maxb
>

```
>    ; If either set is empty, or ranges don't intersect: result = NULL.
>
>  IF maxab LT minab OR maxab LT 0 THEN RETURN, -1
>  r = Where((Histogram(a, Min=minab, Max=maxab) NE 0) AND  $
>           (Histogram(b, Min=minab, Max=maxab) NE 0), count)
>  IF Arg_Present(indices) THEN $
>     indices = Where((Histogram(a, Min=minab, Max=maxab) NE 0))
>  IF count EQ 0 THEN RETURN, -1 ELSE RETURN, r + minab
>  END
>   ************************************************************ ************
>
>  Use it like this:
>
>
>    IDL> request_array = [5,6,7,8,9,10]
>    IDL> avail_array = [3,7,8,9,12,13,16]
>    IDL> int = setintersection(avail_array, request_array, Indices=i)
>    IDL> print, int, i
>             7       8       9
>             2       3       4
>
>  Cheers,
>
>  David
>
>  --
>  David W. Fanning, Ph.D.
>  Fanning Software Consulting, Inc.
>  Phone: 970-221-0438, E-mail: david@dfanning.com
>  Coyote's Guide to IDL Programming: http://www.dfanning.com/
>  Toll-Free IDL Book Orders: 1-888-461-0155
```

Subject: Re: fast array comparison
Posted by Sean Raffuse on Sun, 08 Dec 2002 20:35:57 GMT
View Forum Message <> Reply to Message

Sorry, sent that last post before I wanted to.

```
>  David,
>
>  Thanks for this.  It works almost perfectly.  I am a little confused
though.
>  It seems that the indices keyword returns the indices of the requested
array
>  and not the available array.
```

>   IDL> request_array = [5,6,7,8,9,10]
>   IDL> avail_array = [3,7,8,9,12,13,16]
>   IDL> int = setintersection(avail_array, request_array, Indices=i)
>   IDL> print, int, i
>   7   8   9
>   2   3   4
>

How do I switch it so that indices returns the indices of the available
array. I am apparently not understanding the code.

Thanks yet again,

Sean

>
>
> "David Fanning" <david@dfanning.com> wrote in message
> news:MPG.185d3307aad96bee989a51@news.frii.com...
>> Sean Raffuse (sean@me.wustl.edu) writes:
>>
>>> Let's say I have two arrays.
>>>
>>> requested_array = [5,6,7,8,9,10]
>>> available_array = [3,7,8,9,12,13,16]
>>>
>>> What is the absolute fastest way to determine the indices of
> available_array
>>> that contain values in requested_array? The indices need not match.
> i.e.,
>>> if the two arrays above were used, I would like to return
index=[1,2,3]
>>> because the requested values 7, 8 and 9 are in the available array.
>>
>> The absolute fastest way MUST involve histograms, so
>> I maintain with a great deal of confidence (say, in the 40-50
>> percent range) that this is the fastest possible algorithm:
>>
>>   ** ************
>> FUNCTION SetIntersection, a, b, Indices=indices
>> minab = Min(a, Max=maxa) > Min(b, Max=maxb) ;Only need intersection of
>> ranges
>> maxab = maxa < maxb
>>
>>   ; If either set is empty, or ranges don't intersect: result = NULL.
>>
>> IF maxab LT minab OR maxab LT 0 THEN RETURN, -1

```
>>  r = Where((Histogram(a, Min=minab, Max=maxab) NE 0) AND  $
>>           (Histogram(b, Min=minab, Max=maxab) NE 0), count)
>>  IF Arg_Present(indices) THEN $
>>    indices = Where((Histogram(a, Min=minab, Max=maxab) NE 0))
>>  IF count EQ 0 THEN RETURN, -1 ELSE RETURN, r + minab
>>  END
>>  ********************************************************** ************
>>
>>  Use it like this:
>>
>>
>>    IDL> request_array = [5,6,7,8,9,10]
>>    IDL> avail_array = [3,7,8,9,12,13,16]
>>    IDL> int = setintersection(avail_array, request_array, Indices=i)
>>    IDL> print, int, i
>>             7       8       9
>>             2       3       4
>>
>>  Cheers,
>>
>>  David
>>
>>  --
>>  David W. Fanning, Ph.D.
>>  Fanning Software Consulting, Inc.
>>  Phone: 970-221-0438, E-mail: david@dfanning.com
>>  Coyote's Guide to IDL Programming: http://www.dfanning.com/
>>  Toll-Free IDL Book Orders: 1-888-461-0155
>
>
```

Subject: Re: fast array comparison
Posted by David Fanning on Sun, 08 Dec 2002 20:40:42 GMT
View Forum Message <> Reply to Message

Sean Raffuse (sean@me.wustl.edu) writes:

> Thanks for this. It works almost perfectly. I am a little confused though.
> It seems that the indices keyword returns the indices of the requested array
> and not the available array.

Oh, I just threw that together in about 30 seconds by
adding an INDICES keyword to the SetIntersection program
described here:

http://www.dfanning.com/tips/set_operations.html

I probably got it wrong. While you are fixing it,
you probably want to add some error handling, too,
so you can figure out when there are no matches in
the intersection. :-)

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: fast array comparison
Posted by David Fanning on Sun, 08 Dec 2002 20:47:37 GMT
View Forum Message <> Reply to Message

Sean,

Whoops, I see the problem. But I'm just going to
a basketball game. I'll have a look when I come
back. But you will probably have it solved by then. :-)

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: fast array comparison
Posted by Craig Markwardt on Mon, 09 Dec 2002 01:39:23 GMT
View Forum Message <> Reply to Message

"Sean Raffuse" <sean@me.wustl.edu> writes:
> David,
>
> Thanks for this. It works almost perfectly. I am a little confused though.
> It seems that the indices keyword returns the indices of the requested array
> and not the available array.

Greetings Sean--

Another option is CMSET_OP from my web page. It uses the HISTOGRAM technique, if the dynamic range of values is small, but graduates to a different, sort/uniq-based method, if the dynamic range is large. Indices can be returned using the /INDEX keyword. [In case it isn't obvious, you want to use the 'AND' operation.]

CMSET_OP also returns indices from the *first* array. This is not a problem for your application, since you can simply swap the positions of the first and second arrays in the function call.

Yours,
Craig

http://cow.physics.wisc.edu/~craigm/idl/idl.html (under Array/Set Ops)

Subject: Re: fast array comparison
Posted by Sean Raffuse on Mon, 09 Dec 2002 15:24:29 GMT
View Forum Message <> Reply to Message

Well, I stared at it for quite some time yesterday, and I still can't figure out how to return the index properly. Can you point me in the right direction?

thanks,

Sean

"David Fanning" <david@dfanning.com> wrote in message news:MPG.185d5dd8a0ec97ca989a53@news.frii.com...
> Sean,
>
> Whoops, I see the problem. But I'm just going to
> a basketball game. I'll have a look when I come
> back. But you will probably have it solved by then. :-)
>
> David
>
> --
> David W. Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Phone: 970-221-0438, E-mail: david@dfanning.com
> Coyote's Guide to IDL Programming: http://www.dfanning.com/
> Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: fast array comparison
Posted by JD Smith on Mon, 09 Dec 2002 15:25:07 GMT
View Forum Message <> Reply to Message

On Sun, 08 Dec 2002 18:39:23 -0700, Craig Markwardt wrote:

> "Sean Raffuse" <sean@me.wustl.edu> writes:
>> David,
>>
>> Thanks for this. It works almost perfectly. I am a little confused
>> though. It seems that the indices keyword returns the indices of the
>> requested array and not the available array.
>
> Greetings Sean--
>
> Another option is CMSET_OP from my web page. It uses the HISTOGRAM
> technique, if the dynamic range of values is small, but graduates to a
> different, sort/uniq-based method, if the dynamic range is large.
> Indices can be returned using the /INDEX keyword. [In case it isn't
> obvious, you want to use the 'AND' operation.]
>
> CMSET_OP also returns indices from the *first* array. This is not a
> problem for your application, since you can simply swap the positions of
> the first and second arrays in the function call.
>
> Yours,
> Craig
>
> http://cow.physics.wisc.edu/~craigm/idl/idl.html (under Array/Set Ops)

This topic was discussed ad naseum over the past couple of years, with the critical differences between looking for the values and looking for the indices of intersection pointed out. Several different methods were compared using HISTOGRAM, SORT, and direct array inflation. Depending on your problem size, one of these will be fastest. Usually. ;).

Give it a search on Gusenet.

JD

Subject: Re: fast array comparison
Posted by Sean Raffuse on Mon, 09 Dec 2002 15:48:00 GMT
View Forum Message <> Reply to Message

I apologize for my inexperience. Gusenet?

"JD Smith" <jdsmith@as.arizona.edu> wrote in message

news:pan.2002.12.09.15.25.07.822280.6387@as.arizona.edu...
> On Sun, 08 Dec 2002 18:39:23 -0700, Craig Markwardt wrote:
>
>> "Sean Raffuse" <sean@me.wustl.edu> writes:
>>> David,
>>>
>>> Thanks for this. It works almost perfectly. I am a little confused
>>> though. It seems that the indices keyword returns the indices of the
>>> requested array and not the available array.
>>
>> Greetings Sean--
>>
>> Another option is CMSET_OP from my web page. It uses the HISTOGRAM
>> technique, if the dynamic range of values is small, but graduates to a
>> different, sort/uniq-based method, if the dynamic range is large.
>> Indices can be returned using the /INDEX keyword. [In case it isn't
>> obvious, you want to use the 'AND' operation.]
>>
>> CMSET_OP also returns indices from the *first* array. This is not a
>> problem for your application, since you can simply swap the positions of
>> the first and second arrays in the function call.
>>
>> Yours,
>> Craig
>>
>> http://cow.physics.wisc.edu/~craigm/idl/idl.html (under Array/Set Ops)
>
> This topic was discussed ad naseum over the past couple of years, with
> the critical differences between looking for the values and looking for
> the indices of intersection pointed out. Several different methods were
> compared using HISTOGRAM, SORT, and direct array inflation. Depending on
> your problem size, one of these will be fastest. Usually. ;).
>
> Give it a search on Gusenet.
>
> JD

Subject: Re: fast array comparison
Posted by David Fanning on Mon, 09 Dec 2002 16:00:38 GMT
View Forum Message <> Reply to Message

JD Smith (jdsmith@as.arizona.edu) writes:

> This topic was discussed ad naseum over the past couple of years, with
> the critical differences between looking for the values and looking for
> the indices of intersection pointed out. Several different methods were
> compared using HISTOGRAM, SORT, and direct array inflation. Depending on

> your problem size, one of these will be fastest. Usually. ;).
>
> Give it a search on Gusenet.

JD hasn't had his coffee yet this morning. :-)

Search the IDL newsgroup archives on Goggle for "Matching Lists"
by Mark Fardal for the ad nauseam discussion JD mentions. You
may learn more about list searching than you ever wanted to know. :-)

Or, if you just want an answer (no sense reading this
newsgroup if you fall into this category), here you go:

```
;************************************************************ ******
FUNCTION SetIntersection, a, b, Indices=indices, Count=count
minab = Min(a, Max=maxa) > Min(b, Max=maxb) ; Intersection of ranges
maxab = maxa < maxb

  ; If either set is empty, or ranges don't intersect: result = NULL.

IF maxab LT minab OR maxab LT 0 THEN RETURN, -1
hist_a = Histogram(a, Min=minab, Max=maxab, Reverse_Indices=rev_a)
hist_b = Histogram(b, Min=minab, Max=maxab)
r = Where((hist_a NE 0) AND (hist_b NE 0), count)
IF count EQ 0 THEN BEGIN
   RETURN, -1
ENDIF ELSE BEGIN
   IF Arg_Present(indices) THEN indices = rev_a[rev_a[r]]
   RETURN, r + minab
ENDELSE
END
;************************************************************ ******

  IDL> request_array = [5,6,7,8,9,10]
  IDL> avail_array = [3,7,8,9,12,13,16]
  IDL> int = setintersection(avail_array, request_array, Indices=I)
  IDL> print, int, I
         7       8       9
         1       2       3
```

Remember, this is one of three possible algorithms discussed in
that series of articles. Choose wisely.

Cheers,

David

--

David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: fast array comparison
Posted by David Fanning on Mon, 09 Dec 2002 16:07:44 GMT
View Forum Message <> Reply to Message

David Fanning (david@dfanning.com) writes:

> Search the IDL newsgroup archives on Goggle for "Matching Lists"

Whoops, Sean, that's "Google". "Goggle" is JD's home page. :-)

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: fast array comparison
Posted by JD Smith on Mon, 09 Dec 2002 17:59:37 GMT
View Forum Message <> Reply to Message

On Mon, 09 Dec 2002 08:48:00 -0700, Sean Raffuse wrote:

> I apologize for my inexperience. Gusenet?
>

Oh sorry, that's what I call Google Groups: groups.google.com. E.g. try
this:

http://groups.google.com/groups?as_q=array%20comparison& as_ugroup=*idl-pvwave*

JD
