Subject: Re: fast array comparison
Posted by JD Smith on Mon, 09 Dec 2002 15:25:07 GMT

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