Subject: Using ASSOC function
Posted by Bernard Puc on Thu, 02 Jul 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Hi all,
Using ASSOC, I can access an image array stored in a datafile. Is it possible to instead select a subimage of that array? I'm looking to optimize the file access speed by not having to read the entire file or array into a variable. Thanks in advance.

-Bernard Puc
bpuc@va.aetc.com

Subject: Re: Using ASSOC function
Posted by menakkis on Fri, 03 Jul 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Bernard Puc (puc@gsfc.nasa.gov) wrote:
> Using ASSOC, I can access an image array stored in a datafile. Is it
> possible to instead select a subimage of that array? I'm looking to
> optimize the file access speed by not having to read the entire file or
> array into a variable. Thanks in advance.

If your image array is stored simply then you can do this (to some extent) by setting up your ASSOC with a "tile" instead of a complete image. My meanings here are that "simply" means that there's no header information in between the images and the images are rectangular, and that a "tile" is a physically-contiguous subset of a complete image.

If you do this, you will of course have to keep track of tiles rather than images when working with your ASSOC "variable".

e.g., Suppose that you had a band-sequential array of 100 byte images, each 640 pixels wide and 512 deep. No header info - nothing but the images. ("Band sequential" means that image 0 is stored first, completely, then image 1, then 2, etc, and that each image is stored "across" first, then "down".) Then: OPENU,lun,image_filename,/get_lun image_width=640 ;we *have to* match this in our tile tile_lines=32 ;our tile will be a chunk 640 pixels wide, 32 deep tile=assoc(lun,bytarr(image_width,tile_lines)) This will let you get at your images 32 (complete) lines at a time. There will be 16 tiles to each image (16*32 == 512); e.g., tile(16) will give you the first 32 lines of the second image.

You can make other tiling arrangements, of course. Just remember that the tile must match the way that the data is stored. e.g., a tile of BYTARR(320,32) will NOT give you coherent half-lines in the above example. Also, the tile size should divide evenly into the size of one image.

Cheers

Peter Mason

Subject: Re: Using ASSOC
Posted by Peter Mason on Wed, 26 May 2004 00:50:36 GMT
View Forum Message <> Reply to Message

Jonathan Greenberg wrote:
> I was hoping to get some help/feedback on the use of ASSOC. I'm
> working with a large image file and, while I can extract subsections
> of the image (say 3 consequtive lines), I'd prefer to work with the
> image as an array, but not have it loaded into memory. Is this what
> ASSOC allows me to do? I'm having a problem with the following
> situation:
>
> Nl and ns = the dimensions of the byte image "someimage.dat".
> Openr,unit,'someimage.dat',/get_lun
> Testassoc=assoc(unit,bytarr(nl,ns))
>
> I'd like to be able to extract the value from position x,y from
> testassoc -- how do I do this? Testassoc[0] seems to be the entire
> image. But testassoc[0,1] gives me an end of file error, as does
> testassoc[0,1,1] (which I would think would give me the value at that
> position).
>
> Using the /packed keyword doesn't seem to do anything. Help!

What you have done is to set up ASSOC with a chunk size that's the same size as the whole image. (You don't want to do that as it's the same thing as reading the entire image into memory.) As such, your ASSOC variable can only give you one chunk, chunk number 0.

While ASSOC appears to implement memory mapping, it should really be thought of as "random access made simple". Here's the gist of it:

```
Testassoc=assoc(unit,bytarr(ns,nlchunk))
```

This means: "I want to access this file in chunks, where the size of one chunk is BYTARR(ns,nlchunk)."
(BTW, from your BYTARR(nl,ns), it would appear that your images is stored rotated?)

```
tile=testassoc[17]
```

This means: "Give me the eighteenth chunk." (And TILE ends up being a BYTARR(ns,nlchunk)'s worth of data.)

It's handy because you can whiz around your image, accessing chunks backwards and forwards as you please.

These days you can actually do *proper* memory mapping in IDL, using SHMMAP and SHMVAR (and ultimately SHMUNMAP). Using these functions, you can get your entire image mapped to memory... maybe. If it works, it'll look and function the same as an IDL array into which you have read the image, only it won't necessarily consume as much of your computer's RAM + pagefile as a regular IDL array. I said "maybe" earlier because it might not work if your image is too big. Memory mapping consumes address space in the same way that regular memory allocation does, and on a 32-bit platform (certainly on a win32 platform) you might have even less address space than RAM + pagefile.

So, if your image isn't too large you might want to give proper memory-mapping a go, otherwise experiment with ASSOC using a smallish tile size. For instance, say you have a 1-band image that's NS samples wide and NL lines long, and it's stored "across then down". The simplest way to get pixels in it would be to set up a line-sized chunk:

```
linebuf=ASSOC( unit, BYTARR(ns) )
```

Then to get the pixel at, say sample X and line Y (counting from 0), you could do:

```
pixel_xy=(linebuf[y])[x]
```

...The simplest way but probably not the most efficient. To improve efficiency, try using a larger chunk size (more than 1 line) and caching it, updating whenever Y is "out of range".

Subject: Re: Using ASSOC
Posted by JD Smith on Wed, 26 May 2004 01:28:32 GMT
View Forum Message <> Reply to Message

On Tue, 25 May 2004 23:03:57 +0000, Jonathan Greenberg wrote:

> I was hoping to get some help/feedback on the use of ASSOC. I'm working
> with a large image file and, while I can extract subsections of the image
> (say 3 consequtive lines), I'd prefer to work with the image as an array,
> but not have it loaded into memory. Is this what ASSOC allows me to do?
> I'm having a problem with the following situation:
>
> Nl and ns = the dimensions of the byte image "someimage.dat".
> Openr,unit,'someimage.dat',/get_lun
> Testassoc=assoc(unit,bytarr(nl,ns))
>
> I'd like to be able to extract the value from position x,y from testassoc --

> how do I do this? Testassoc[0] seems to be the entire image. But
> testassoc[0,1] gives me an end of file error, as does testassoc[0,1,1]
> (which I would think would give me the value at that position).
>
> Using the /packed keyword doesn't seem to do anything. Help!

ASSOC is designed to work with sequential chunks of data, e.g. a 3D image array of dimension n1xn2xn3 could be indexed with with a=ASSOC(unit,bytarr(n1,n2)) & a[0..n3-1]. What is not mentioned in the manual (which is confusing on this topic), is that this only works for sequential pieces of data. If you have a large image, and you'd like to work with 2D subsections of the image, ASSOC will not do what you want. Let's say an image is 2048x2048, and you'd like to work with a 256x256 subsection somewhere in the middle. This data is not written contiguously in memory; instead you have to skip over most of each row to get to the portion of the next row belonging to the subimage.

A few ways to deal with this problem are:

1. Re-format your image on disk to consist of a series (in memory) of sub-array tiles written out serially. Then ASSOC should work fine with the fixed tile size you give it.

2. Use the new shared memory mapping in IDL>5.6 to use your systems virtual memory subsystem to access non-contiguous tiles of an array. The file will only be read from and written to memory as needed. You'll need to keep up with the global indexing yourself. Keep in mind that you will make the virtual memory system work hard as it skips around in memory finding the pieces you need.

JD

Subject: Re: Using ASSOC
Posted by David Fanning on Wed, 26 May 2004 02:33:03 GMT
View Forum Message <> Reply to Message

Peter Mason writes:

> These days you can actually do *proper* memory mapping in IDL, using SHMMAP
> and SHMVAR (and ultimately SHMUNMAP).

Peter, where have you been!? We could use more of these informative answers! :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
