Subject: Today's IDL Lesson
Posted by David Fanning on Thu, 14 Aug 2008 19:05:13 GMT
View Forum Message <> Reply to Message

Folks,

A couple of weeks ago I was give some IDL code to, essentially, find blobs in a large image, do some processing on them, and return a result image. The images we want to work on are on the order of 8800 by 6800, pretty big.

Naturally, the code contained FOR loops coming out the wazzoo and it was VERY slow. I don't know *how* slow because I turned the computer off when I noticed smoke coming out the back and I couldn't immediately locate the damn fire extinguisher.

Because lousy code doesn't necessarily prevent you from writing scientific papers about your algorithm, I read the paper to get the gist and decided their algorithm was overly complicated and that I could do it a lot more simply (and in WAY fewer loops).

So I code it up, using a much smaller image subset, and it was quicker than snot. When I thought I had all the kinks worked out of it, I decided to give'er a try on the larger images. It took about 48 minutes to run. Humm. Odd.

But I couldn't immediately see where the problem was. I decided to play tennis instead of work on it anymore. :-)

This morning I ran the PROFILER on the code, and saw it was spending most of its time in the WHERE function. I was using the WHERE to locate the indices of blobs I wanted to work with from the LABEL_REGION image. A couple of quick tests showed that as that as that image gets larger, the search time of WHERE goes up exponentially.

"Humm, I wish LABEL_REGION just returned the indices like the HISTOGRAM function does," I thought.

What!? After 20 years of working with IDL it finally sunk in. What I need is a HISTOGRAM!!!!

Here is the bottom line. My program, which took 48 minutes to run yesterday, takes 10 *seconds* to run today.

Sometimes you just gotta love IDL! :-)

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming (www.dfanning.com)
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Today's IDL Lesson
Posted by Kenneth P. Bowman on Thu, 14 Aug 2008 19:41:17 GMT
View Forum Message <> Reply to Message

In article <MPG.230e2fd1f36aa30a98972c@news.giganews.com>,
David Fanning <news@dfanning.com> wrote:

> This morning I ran the PROFILER on the code, and saw it
> was spending most of its time in the WHERE function.
> I was using the WHERE to locate the indices of blobs I
> wanted to work with from the LABEL_REGION image. A couple
> of quick tests showed that as that as that image gets larger,
> the search time of WHERE goes up exponentially.

Shouldn't that be "increases linearly with the image size and
number of invocations of WHERE"?

Ken

Subject: Re: Today's IDL Lesson
Posted by David Fanning on Fri, 15 Aug 2008 02:12:10 GMT
View Forum Message <> Reply to Message

Kenneth P. Bowman writes:

> Shouldn't that be "increases linearly with the image size and
> number of invocations of WHERE"?

I don't think so. A 1500x1500 array took about 3.2 sec.
a 2500x2500 array took about 35 sec. Just eyeballing it,
it doesn't look linear to me. :-)

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Today's IDL Lesson
Posted by Allan Whiteford on Fri, 15 Aug 2008 09:05:50 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> Kenneth P. Bowman writes:
>
>
>> Shouldn't that be "increases linearly with the image size and
>> number of invocations of WHERE"?
>
>
> I don't think so. A 1500x1500 array took about 3.2 sec.
> a 2500x2500 array took about 35 sec. Just eyeballing it,
> it doesn't look linear to me. :-)
>
> Cheers,
>
> David

This scared me!

So, I wrote the following:

```
pro profile_where

    max=26
    time=fltarr(max)
    sizes=(1+2*(findgen(max) mod 3))*10l^float(indgen(max)/3)

    for i=0l,max-1 do begin
 array=findgen(sizes[i])
        time[i]=call_external('libidl.so','clock')
 idx=where(array lt 3)
        time[i]=call_external('libidl.so','clock') -time[i]
    end

    plot,sizes,time/1e6,xtitle='Number of elements',ytitle='Time in s'
    oplot,sizes,time/1e6,psym=2
```

```
    ; or...

    plot,alog10(sizes),alog10(time/1e6),xrange=[5,10]
    oplot,alog10(sizes),alog10(time/1e6),xrange=[5,10],psym=2

end
```

and I get something which looks basically linear. The maximum array size were chosen so that I wasn't hitting swap space and the where() criterion so that each iteration returned the same number of results (hence allocated the same amount of memory).

I have no doubt that in real world applications we get non-linear scaling due to hitting swap or various other reasons but I think the above shows that the basic usage of where() has linear scaling.

I hope this is as much a relief to others as it is to me.

Thanks,

Allan

Subject: Re: Today's IDL Lesson
Posted by David Fanning on Fri, 15 Aug 2008 12:11:49 GMT
View Forum Message <> Reply to Message

Allan Whiteford writes:

> I have no doubt that in real world applications we get non-linear
> scaling due to hitting swap or various other reasons but I think the
> above shows that the basic usage of where() has linear scaling.
>
> I hope this is as much a relief to others as it is to me.

Well, there you go. Yesterday's lesson was really about
not using two data points, hastily collected, to draw conclusions. ;-)

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Today's IDL Lesson
Posted by MarioIncandenza on Wed, 20 Aug 2008 17:01:54 GMT
View Forum Message <> Reply to Message

http://www.dfanning.com/ip_tips/blobanalysis.html