Subject: vectorising versus loops
Posted by nasalmon on Sun, 22 Feb 2004 18:16:41 GMT
View Forum Message <> Reply to Message

Does anyone know what the speed increase factor is in IDL programmes when going from "do loops" to full vectorisation of arrays? I know all programmes are different and not every process lends itself to vectorisation. However, there must be some rule of thumb, ie speed going as a linear function of the number of array elements times some factor.

Also, are there any tricks to play if you want to vectorise loops that have IF statement decision in them, or any general rules for neat vectorisation of looped programmes?

Many thanks,
Neil

Subject: Re: vectorising versus loops
Posted by Craig Markwardt on Tue, 24 Feb 2004 00:04:09 GMT
View Forum Message <> Reply to Message

nasalmon@onetel.net.uk (Neil) writes:
> Craig,
>
> yes many thanks for this valuable information. One of the problems i
> have is that the condition in the WHERE statement has to contain the
> counter in the "do loop", that is basically why i put it in the "do
> loop" to start with. Is there any way i can make the condition depend
> on the counter, ie the vector index.

Yes, you can make a vector of index values with LINDGEN,

```
;; Make a list of indices
ii = lindgen(n_elements(x))

;; Select on X and the index variable
wh = where(x GE 0 AND ii LT 200)
```

Good luck,
Craig

Subject: Re: vectorising versus loops
Posted by Norbert Hahn on Tue, 24 Feb 2004 10:13:03 GMT
View Forum Message <> Reply to Message

Craig Markwardt <craigmnet@REMOVEcow.physics.wisc.edu> wrote:

> ;; Make a list of indices
> ii = lindgen(n_elements(x))
>
> ;; Select on X and the index variable
> wh = where(x GE 0 AND ii LT 200)

While this is an example of using the index variable,
this statement by itself can be shortened to

wh = where(x[0:199] ge 0)

Norbert

Subject: Re: vectorising versus loops
Posted by Craig Markwardt on Sun, 07 Mar 2004 17:24:43 GMT
View Forum Message <> Reply to Message

nasalmon@onetel.net.uk (Neil) writes:
...
>
> However, there is one small outstanding problem in the vectorisation,
> and this involves vectorisation of a routine that uses the cross or
> vector product, the IDL routine being CROSSP, generating a vector from
> the cross product of two vectors.
>
> Currently i am forced to use this in a loop, as my vectorising
> attempts (see below) have failed. The loop statement below gives the
> correct result (i being the counter of one of the outer loops):
...

Neil, how about using CROSSPN, a vectorized cross product function for
just such purposes?

Craig

P.S. http://cow.physics.wisc.edu/~craigm/idl/idl.html (under Math)

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@REMOVEcow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: vectorising versus loops
Posted by nasalmon on Fri, 26 Mar 2004 19:34:51 GMT
View Forum Message <> Reply to Message

Craig Markwardt <craigmnet@REMOVEcow.physics.wisc.edu> wrote in message news:<ond67ozi1w.fsf@cow.physics.wisc.edu>...
> nasalmon@onetel.net.uk (Neil) writes:
> ...
>>
>> However, there is one small outstanding problem in the vectorisation,
>> and this involves vectorisation of a routine that uses the cross or
>> vector product, the IDL routine being CROSSP, generating a vector from
>> the cross product of two vectors.
>>
>> Currently i am forced to use this in a loop, as my vectorising
>> attempts (see below) have failed. The loop statement below gives the
>> correct result (i being the counter of one of the outer loops):
> ...
>
> Neil, how about using CROSSPN, a vectorized cross product function for
> just such purposes?
>
> Craig
>
> P.S. http://cow.physics.wisc.edu/~craigm/idl/idl.html (under Math)

Well thanks very much for CROSSSP, it worked like a dream, fitted in and sailed through perfectly.

In the main programme structure there were loops nested 4 deep, so vectorising the inner most loop, now leaves only 3 nested loops. Vectorisation used lots of WHERE statements. As the inner loop dealt with vectors, inner products of these could be conveniently replaced with matrix multiplications and transpositions.

However, the improvement factor in speed is only between of one and five, which is small, but certainly worth having. One of the reasons why it is this low is because there is lots of stuff in the other loops and sometimes the inner most loop is very short, may be only 3 elements, while some of the outer loops need to munch thousands of times.

So i was wondering whether there were any other tricks to vectorisation that i had missed, that i could use to improve speed? I can imagine that trying to vectorise the second to inner most loop may be possible. Are there any strategies or special routines that could be used in the vectorisation of loops higher in a hierarchy of loops?

many thanks,

Neil

Subject: Re: vectorising versus loops
Posted by David Fanning on Fri, 26 Mar 2004 19:47:52 GMT
View Forum Message <> Reply to Message

Neil writes:

> So i was wondering whether there were any other tricks to
> vectorisation that i had missed, that i could use to improve speed?

I presume you have read the "Embodiment of Pure Evil" article:

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

And the follow-up secular advice:

http://www.dfanning.com/tips/forloops2.html

Cheers,

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

Subject: Re: vectorising versus loops
Posted by stimmins on Mon, 12 Apr 2004 20:14:49 GMT
View Forum Message <> Reply to Message

Neil:
You have a common problem with your "busy" inner or outer loops. To save more time you need to move subsets like x = y(0:1000) and IF statements out of the inner loops because these can really dominate the time. If you send me your code (and an idea of how to construct some dummy data) I will make it run faster!

Sidey Timmins (my first name rhymes with Heidi!)
