Subject: Re: Cyclic array interfaces
Posted by David Fanning on Tue, 13 Jul 2004 00:50:25 GMT
View Forum Message <> Reply to Message

Jonathan Greenberg writes:

> A few months back I was asking about efficient ways of interacting with
> images where I can perform spatial transforms (e.g. Applying a 5 x 5 filter
> of some sort to the image), and I got lots of good information back, and one
> idea in particular intrigued me: working with an array in a cyclic fashion.
> My question is: is replacing a single "line" of an N x M arrau with a 1 x M
> vector using something like:
>
> A=[[1,2,3],[4,5,6]]
> B=[7,8,9]
> A[*,1]=B
>
> ... An efficient method of replacing data in an array? Or does A get
> completely rewritten and takes as much time as:
>
> C=[[10,11,12],[13,14,15]]
> A=C
>
> (e.g. Does IDL actually rewrite the entire array, regardless of how many
> elements are being changed, or does it only change the particular elements
> and hence the first example should be about twice as fast as the second)?
>
> If example #1 is faster than #2, then I can implement a cyclical array
> approach, where if I want to work with 5 lines of an image at a time, the
> "line index" the first iteration would be:
> 0,1,2,3,4
> And the second iteration (as I shift down one line):
> 5,1,2,3,4 (so I'm overwriting only one line of data at a time).
>
> Thoughts?

Others will write with more explicit examples, probably, but
here is an article that will get you started in the right
direction. I'm not sure about time, but some ways of subscripting
arrays can certainly take a lot of *memory*, which I presume means
time, too.

http://www.dfanning.com/misc_tips/submemory.html

Cheers,

David

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

Subject: Re: Cyclic array interfaces
Posted by cedricl on Tue, 13 Jul 2004 12:59:04 GMT
View Forum Message <> Reply to Message

Since you're aiming for optimization, I might as well point out an obvious one:

> A=[[1,2,3],[4,5,6]]
> B=[7,8,9]
> A[*,1]=B

Use A[0, 1] = B instead, since the elements are contiguous in the first dimension. You might also want to take a loot at "temporary" if you haven't already.

> (e.g. Does IDL actually rewrite the entire array, regardless of how many
> elements are being changed,

No.

> or does it only change the particular elements
> and hence the first example should be about twice as fast as the second)?

It doesn't work that way, like the previous poster has shown. Your performance mostly depends on the type of access that you're doing. If you're accessing memory in a continuous fashion (the first dimension varies), it's almost sure IMO that it's going to be faster.
Furthermore, consider the next example:

```
a = fltarr(5)
; Option 1: Create a new array
a = [3, 4, 5, 6, 7]
; Option 2: Rewrite the array
a[0] = [3, 4, 5, 6, 7]
```

Option 1 and 2 are equivalent, code-wise, but I would argue (and of course, profile, were I not lazy) that option 2 is faster, because it doesn't have to deallocate the array that was already contained in a, and allocate a new one. Of course, there are many other optimizations that IDL could do that would make 1 faster than 2, but considering RSI's approach to compile-time optimizations, I'm confident in my analysis.

Cedric

Subject: Re: Cyclic array interfaces
Posted by Mark Hadfield on Tue, 13 Jul 2004 22:29:23 GMT
View Forum Message <> Reply to Message

Cedric wrote:
> Furthermore, consider the next example:
>
> a = fltarr(5)
> ; Option 1: Create a new array
> a = [3, 4, 5, 6, 7]
> ; Option 2: Rewrite the array
> a[0] = [3, 4, 5, 6, 7]
>
> Option 1 and 2 are equivalent, code-wise, but I would argue (and of
> course, profile, were I not lazy) that option 2 is faster, because it
> doesn't have to deallocate the array that was already contained in a,
> and allocate a new one. Of course, there are many other optimizations
> that IDL could do that would make 1 faster than 2, but considering
> RSI's approach to compile-time optimizations, I'm confident in my
> analysis.

Analysis be dammed, I counted them! And got a result that disagrees with your analysis:

```
IDL> n = 10000000 & a = fltarr(n) & t0 = systime(1) & a = findgen(n) &
t1 = systime(1) & a[0] = findgen(n) & t2 = systime(1) & print, t1-t0, t2-t1
      0.14000010      0.25000000
IDL> print, !version
{ x86 Win32 Windows Microsoft Windows 6.0 Jun 27 2003      32      64}
```

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

Subject: Re: Cyclic array interfaces
Posted by cedricl on Wed, 14 Jul 2004 01:48:04 GMT
View Forum Message <> Reply to Message

Mark Hadfield <m.hadfield@niwa.co.nz> wrote in message news:<cd1no6$q54$1@newsreader.mailgate.org>...
> Analysis be dammed, I counted them! And got a result that disagrees with

> your analysis:

:) I should have remembered that programmers are always wrong when
they don't use a profiler!

I'm still surprised by your result, though. Perhaps IDL knows that
findgen(n) is a temporary in the first case (thus assigning its
address directly to a) and not in the second case. I don't have access
to IDL now, but I would try

n = 10000000 & a = fltarr(n) & b = findgen(n) & t0 = systime(1) & a =
b & t1 = systime(1) & a[0] = b & t2 = systime(1) & print, t1-t0, t2-t1

I'll see if I can do it tomorrow (or if anybody feels like doing it
now...)

Thanks for the heads up!

Cedric

Subject: Re: Cyclic array interfaces
Posted by Mark Hadfield on Wed, 14 Jul 2004 02:53:06 GMT
View Forum Message <> Reply to Message

Cedric wrote:
> Mark Hadfield <m.hadfield@niwa.co.nz> wrote in message
news:<cd1no6$q54$1@newsreader.mailgate.org>...
>
>> Analysis be dammed, I counted them! And got a result that disagrees with
>> your analysis:
>
>
> :) I should have remembered that programmers are always wrong when
> they don't use a profiler!
>
> I'm still surprised by your result, though. Perhaps IDL knows that
> findgen(n) is a temporary in the first case (thus assigning its
> address directly to a) and not in the second case.

The result below supports this.

> I don't have access
> to IDL now, but I would try
>
> n = 10000000 & a = fltarr(n) & b = findgen(n) & t0 = systime(1) & a =
> b & t1 = systime(1) & a[0] = b & t2 = systime(1) & print, t1-t0, t2-t1

0.14100003 0.093999863

The results are pretty consistent, by the way.

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

*** More included text than new text. Blah blah blah ***
*** More included text than new text. Blah blah blah ***
*** More included text than new text. Blah blah blah ***
*** More included text than new text. Blah blah blah ***
*** More included text than new text. Blah blah blah ***
*** More included text than new text. Blah blah blah ***
*** More included text than new text. Blah blah blah ***
*** More included text than new text. Blah blah blah ***