Subject: Re: large matrix operations
Posted by Vince Hradil on Mon, 13 Oct 2008 21:17:10 GMT
View Forum Message <> Reply to Message

On Oct 13, 3:51 pm, "mgal...@gmail.com" <mgal...@gmail.com> wrote:
> On Oct 13, 11:07 am, Vince Hradil <vincehra...@gmail.com> wrote:
>
>
>
>> So much for that theory...
>
>> Hardware:
>> IDL> print, !version
>> { x86 Win32 Windows Microsoft Windows 7.0 Oct 25 2007 32 64}
>
>> Results:
>> % TEST: Allocating A array
>> % TEST: took 0.14000010 sec
>> % TEST: Allocating B array
>> % TEST: took 0.21899986 sec
>> % TEST: A#B
>> % TEST: took 40.878000 sec
>> % TEST: matrix_multipy(A,B)
>> % TEST: took 42.284000 sec
>> % TEST: transpose(A)#B
>> % TEST: took 42.645000 sec
>> % TEST: matrix_multipy(A,B,/atranspose)
>> % TEST: took 43.449000 sec
>> % TEST: transpose(temporary(A))#B
>> % TEST: took 43.387000 sec
>> % TEST: matrix_multipy(temporary(A),B,/atranspose)
>> % TEST: took 50.029000 sec
>
> I think there are problems in the testing mechanism:
>
> * you're only timing one execution of the operation
> * you're timing MESSAGE which has an I/O component
> * allocating B was 50% longer than allocating A? They are the same
> size, only 12 MB.
>
> I ran the same tests 10 times and averaged the results, also removing
> the MESSAGE statements from the timed portions.
>
> Here are my results:
>
> IDL> print, !version
> { i386 darwin unix Mac OS X 7.0 Oct 25 2007 32 64}
>

```
>  % TEST: Allocating A: 0.0767788
>  % TEST: Allocating B: 0.0775957
>  % TEST: A # B: 7.68955
>  % TEST: MATRIX_MULTIPLY(A, B): 7.62423
>  % TEST: TRANSPOSE(A) # B: 7.64414
>  % TEST: MATRIX_MULTIPLY(A, B, /ATRANSPOSE): 6.66077
>  % TEST: TRANSPOSE(TEMPORARY(A)) # B: 7.51523
>  % TEST: MATRIX_MULTIPLY(TEMPORARY(A), B, /ATRANSPOSE): 6.50667
>
>  Here is my code:
>
>  pro test
>    nel = 2000L
>    ntests = 10L
>    times = fltarr(8, ntests)
>
>    for i = 0L, ntests - 1L do begin
>      print, 'Running test ' + strtrim(i, 2) + '...'
>      t0 = systime(1)
>      a = randomu(seed,[nel,nel])
>      times[0, i] = systime(1)-t0
>
>      t0 = systime(1)
>      b = randomu(seed,[nel,nel])
>      times[1, i] = systime(1)-t0
>
>      t0 = systime(1)
>      c = a#b
>      times[2, i] = systime(1)-t0
>
>      t0 = systime(1)
>      c = matrix_multiply(a,b)
>      times[3, i] = systime(1)-t0
>
>      t0 = systime(1)
>      c = transpose(a)#b
>      times[4, i] = systime(1)-t0
>
>      t0 = systime(1)
>      c = matrix_multiply(a,b,/atranspose)
>      times[5, i] = systime(1)-t0
>
>      ahold = a
>
>      t0 = systime(1)
>      c = transpose(temporary(a))#b
>      times[6, i] = systime(1)-t0
>
```

```
>      a = ahold
>
>      t0 = systime(1)
>      c = matrix_multiply(temporary(a),b,/atranspose)
>      times[7, i] = systime(1)-t0
>    endfor
>
>    message, 'Allocating A: ' + strtrim(mean(times[0, *]), 2), /info
>    message, 'Allocating B: ' + strtrim(mean(times[1, *]), 2), /info
>    message, 'A # B: ' + strtrim(mean(times[2, *]), 2), /info
>    message, 'MATRIX_MULTIPLY(A, B): ' + strtrim(mean(times[3, *]), 2), /
> info
>    message, 'TRANSPOSE(A) # B: ' + strtrim(mean(times[4, *]), 2), /info
>    message, 'MATRIX_MULTIPLY(A, B, /ATRANSPOSE): ' +
> strtrim(mean(times[5, *]), 2), /info
>    message, 'TRANSPOSE(TEMPORARY(A)) # B: ' + strtrim(mean(times[6,
> *]), 2), /info
>    message, 'MATRIX_MULTIPLY(TEMPORARY(A), B, /ATRANSPOSE): ' +
> strtrim(mean(times[7, *]), 2), /info
> end
>
> Mike
> --www.michaelgalloy.com
> Tech-X Corporation
> Software Developer II
```

Mike,
Thanks for cleaning that up. I think the conclusions are the same - matrix_multiply does not speed things up. I wonder what happens if we really push the envelope wrt RAM and matrix size. Unfortunately, I don't have a lot of time to play with this.

Vince
