Subject: Re: Fast matrix filling in IDL
Posted by weitkamp on Wed, 06 Jan 1999 08:00:00 GMT
View Forum Message <> Reply to Message

In article <36798167.7F63C1CE@io.harvard.edu>,
  Martin Schultz <mgs@io.harvard.edu> wrote:
> David Ritscher wrote:
>
>>> Hopefully everyone is running this a few times and noting when
>>> the times have become "stable". Here are three sequential runs
>>> on a Sun Sparc 2 (IDL 5.1):
>>>
>>> IDL> test
>>> Time for Loop: 2.7740721
>>> Time for Matrix Operations: 5.2337180 ; OUT OF WHACK!
>>> Time for Rebin Operations: 0.16969705
>>> IDL> test
>> [...]
>
>> Perhaps the first one is not out of whack, but rather, the accurate
>> time.
>> When the test is being performed a second time, the operating system has
>> information that is cached, thus avoiding doing part of the work
>> associated with the task. Particularly in big tests, I often notice
>> speed-ups each time, for the first few times I test something. I like
>> to test by completely exiting my environment, doing other things, and
>> coming back and repeating the test. If I'm feeling hard-core about it,
>> I do a reboot in between.
>>
>> David Ritscher
>>
>
> Although I am definitively not a system expert, this sounds very reasonable
> and corresponds to my experience. E.g., when I read a binary data file, it
> usually takes much longer the first time than at subsequent times - often
> it is not even too much faster than sequential reading of the same file in
> ASCII(!) (but on subsequent attempts, the binary read is 1-2 orders of
> magnitude faster). I would assume that this behaviour is somehow associated
> with memory allocation/swapping, together with caching as David noted. The
> speed difference should therefore depend on the array size. It would be
> nice to devise some "objective" tests on this matter, except if some C
> malloc guru out there already knows the exact answer.
>
> Here another little piece of info on this: on my SGI, I get different
> behaviour for two slightly different routines:
> #1 is the same as Stein Vidar listed it - on the first call, the matrix
> operations take longer than on subsequent calls
> #2 I changed array to array1, array2, and array3, "declaring" all three of

> them in the beginning: This time I get more pronounced changes in the rebin
> version! (both tests done after exiting the shell and new login)
>
> Interestingly, when I call test#1 first, then #2 (they are both in one pro
> file), the times reported for #2 are stable from the beginning and vice
> versa. There must be some overhead processing going on when IDL executes a
> routine for the first time.
>
> [...]

This means that the "correct" way of determining computing time for this piece of code strongly depends its later use within a program. My original question, "what is the fastest way of filling a matrix with identical column vectors", may therefore not have a definite answer even for a given machine, except "it depends on how many times this will be done and what happens in between".

Anyway, some interesting questions have been raised here, and David's and Stein Vidar's solutions to my problem have helped me speed up my program enormously.

Thanks,

Timm

P.S. Even on the different HP9000's I use, the relative speeds for the different operations depend strongly on slight differences in architecture:

HP-UX B.10.20 9000/871 (IDL 5.1)
 Time for Loop: 0.72376597
 Time for Matrix Operations: 0.095106006
 Time for Rebin Operations: 0.15038002

HP-UX B.10.20 9000/755 (IDL 5.1)
 Time for Loop: 0.79779303
 Time for Matrix Operations: 0.083228946
 Time for Rebin Operations: 0.57194102

--
Timm Weitkamp
European Synchrotron Radiation Facility (ESRF)
B.P. 220
F-38043 Grenoble Cedex
France

E-mail: weitkamp@esrf.fr