Subject: Re: efficient use of call_external
Posted by David Foster on Wed, 22 May 1996 07:00:00 GMT
View Forum Message <> Reply to Message

nelson@star.net (Nelson Tarr) wrote:
>
> I am interested in any information on the speed improvement that can
> be expected in translating parts of an IDL procedure to C. I plan on
> using "call_external" to invoke the functions within my C module.
>
> I am also interested in suggestions about when it is most beneficial
> to use C instead of IDL.

If the operation you need can be performed using IDL's built-in functions and/or array operations, you're better off sticking with that. IDL functions are optimized very well.

If, in your code, you find yourself having to perform iterative operations explicitly, then you are better off using C or Fortran, as performance will improve DRAMATICALLY. Even though IDL code is "compiled", the pseudo-code that is generated is interpreted, so iterative operations take a big cut in performance.

Some of the examples of cases where we use CALL_EXTERNAL are:

1. Needed a "personalized" type of trilinear interpolation that didn't use background (0) pixels in the interpolation (to avoid aliasing an image with sharp edges).

2. A three-dimensional neighbor-averaging filter applied to a 3D data-set.

3. Needed a "personalized" run-length-encoding algorithm.

Hope this is useful.

Dave Foster
Brain Image Analysis Lab, UCSD
foster@bial1.ucsd.edu

Subject: Re: efficient use of call_external
Posted by peter on Wed, 22 May 1996 07:00:00 GMT
View Forum Message <> Reply to Message

Nelson Tarr (nelson@star.net) wrote:
: I am interested in any information on the speed improvement that can
: be expected in translating parts of an IDL procedure to C. I plan on

: using "call_external" to invoke the functions within my C module.

: I am also interested in suggestions about when it is most beneficial
: to use C instead of IDL.

One rule of thumb is "Use call_external to replace the inner loop of nested loops". That is, when you cannot avoid writing loops in IDL, you will see tremendous improvement by replacing the inner loop(s), but much less payback for replacing the outer loops. If you can vectorize the inner loop (i.e. rewrite it to use IDL array operations) you will see almost as much improvement.

If you are able to use IDL primitives and built-ins (e.g. array addition, FFT, transposes), don't expect any gain from call_external.

Peter

Peter Webb, HP Labs Medical Dept
E-Mail: peter_webb@hpl.hp.com
Phone: (415) 813-3756

Subject: Re: efficient use of call_external
Posted by rivers on Wed, 22 May 1996 07:00:00 GMT

View Forum Message <> Reply to Message

In article <4ntrfg$697@acs1.star.net>, nelson@star.net (Nelson Tarr) writes:
> I am interested in any information on the speed improvement that can
> be expected in translating parts of an IDL procedure to C. I plan on
> using "call_external" to invoke the functions within my C module.

The answer depends entirely upon what your IDL code does. If you were to try to rewrite the IDL FFT routine in C, and use CALL_EXTERNAL to call it, you would probably have to work hard to make it any faster than using IDL. On the other hand, if you have to do some special processing on large arrays which cannot be coded in IDL to use array primitives and/or built-in IDL functions which handle arrays, then you can get 10-100 times speed improvement using CALL_EXTERNAL.

> I am also interested in suggestions about when it is most beneficial
> to use C instead of IDL.

See above, and also Chapter 12 (Efficient Programming) in the IDL User's Guide.

Mark Rivers (312) 702-2279 (office)
CARS (312) 702-9951 (secretary)

Univ. of Chicago (312) 702-5454 (FAX)
5640 S. Ellis Ave. (708) 922-0499 (home)
Chicago, IL 60637 rivers@cars3.uchicago.edu (Internet)