Subject: Re: Memory fragmentation, passing and common blocks
Posted by webb on Sun, 23 May 1993 22:31:36 GMT

View Forum Message <> Reply to Message

mayor@vaxine.larc.nasa.gov writes:

> 2) I've written quite a large IDL application and have stuck with
> the convention of passing variables among all the modules. With the
> exception of a few very small common blocks for a few widget event
> handlers, I've avoided common blocks. Now after all this development,
> I'm realizing that I'm passing huge arrays back and forth and wondering
> if it would have been better to put these in common blocks. So this is
> a two part question:
>
> a) Can passing variables cause memory fragmentation?
>
> b) What exactly do I have to gain or loose if I start putting variables
> in common blocks instead of passing them?

Passing variables does *not* cause memory problems, because IDL passes variables by reference. In the user's guide, the description of parameter passing implies that copies are made of actual parameters, then they are processed, then they are copied back (see section 10-5). However, this does not really seem to be true. I've tested this by filling up memory with large arrays till IDL cannot allocate any more space, then attempting to call a procedure that does not itself need to allocate data -- IDL successfully calls the routine, implying parameter passing does not cause data to be copied.

e.g.

declare the routine mtest.pro

```
pro mtest, arg1
print, arg1(0)
end
```

then (on a Sun with 64M of virtual memory)

```
IDL> a = fltarr(2000,2000,/nozero)
IDL> b = a
IDL> c = a
IDL> d = a
% Unable to allocate memory: to make array.
  Not enough memory
% Execution halted at $MAIN$  .
IDL> mtest, a
```

```
% Compiled module: MTEST.
  0.000668437
```

This section of the User's guide could use some rewriting!

Peter