Subject: Re: IDL -> FORTRAN translator?
Posted by grunes on Wed, 23 Aug 1995 07:00:00 GMT
View Forum Message <> Reply to Message

In article <41ctj7$oph@news.bu.edu> dlmatt@bu.edu (David Matthews) writes:
> We are writing a lot of IDL code to analyze large data sets. A colleague at
> another institution doesn't want to use IDL, wants us to provide Fortran
> source to do the same jobs. Has anyone written a program to run on unix or VMS
> that will translate IDL source (*.pro) into Fortran? I recognize that the IDL
> approach is sufficiently different from Fortran that it might be very hard to
> do a complete job automatically, but Fortran equivalents to IDL are sometimes
> given in the IDL manuals.

If you get any useful responses, please share them with the news group.
I, for one would find any translater between IDL or PV-WAVE and any
compiled language, or the reverse, to be quite useful.

The problem is inherently dificult, in fact almost insoluable, since
compiled languages like Fortran require compile-time information about
symbols that is often dificult to obtain in interpreted languages like
IDL.

The obvious way of dealing with that--changing each symbol
reference to a subroutine call, which interprets stack contents, or some
such structure--is not only inefficient, but it would be unreadable.

(In a sense that was once done--the initial implementation of
IDL was written in Fortran.)

Some time ago (13 Jul 1995) I posted the reverse question:

> Do converters exist to translate Fortran or C into IDL or PV-WAVE?
>
> Obviously, a lot of posts have asked the reverse, and the
> answer always has to be that an Interpreter like IDL or PV-WAVE
> allows symbols to change their meanings (one moment it is a scalar integer,
> the next it is a character array, the next it is a procedure...) in such a
> way that it would be very hard to generate efficient code in a
> compiled language like Fortran or C.
>
> But this direction should mostly be possible. And some of us would
> find it quite useful. (But I don't want to write it.) It would be
> especially useful if it had the option of taking some (perhaps user
> labeled) Fortran 77 loops and convert them into array operations.

I got a response from knipp@ipi.uni-hannover.de. Unfortunately
it was fairly primitive, made some strong assumptions about the
code and the way (e.g., case) in which it was coded, and would

only work on a certain classof quite simple programs. E.G.,
no dimensioned variables, etc. But it did some of the work, by
changing "program" to "pro", ** to ^, .eq. to eq, etc.

---------- ----------
Mitchell R Grunes. Opinions are mine alone.

Subject: Re: IDL -> FORTRAN translator?
Posted by Mike Mathews on Wed, 23 Aug 1995 07:00:00 GMT
View Forum Message <> Reply to Message

dlmatt@bu.edu (David Matthews) wrote:
> We are writing a lot of IDL code to analyze large data sets. A colleague at
> another institution doesn't want to use IDL, wants us to provide Fortran
> source to do the same jobs. Has anyone written a program to run on unix or VMS
> that will translate IDL source (*.pro) into Fortran? I recognize that >
>

What is IMSL? Isn't it related to IDL, but for Fortran?

Mike
---------- mailto:fskmjm@pukfsk.puk.ac.za ----------
Antarctic Workgroup Potchefstroom University
Physics Department South Africa
---------- http://www.puk.ac.za/fskdocs/ ----------

Subject: Re: IDL -> FORTRAN translator?
Posted by sjt on Thu, 24 Aug 1995 07:00:00 GMT
View Forum Message <> Reply to Message

David Matthews (dlmatt@bu.edu) wrote:
: We are writing a lot of IDL code to analyze large data sets. A colleague at
: another institution doesn't want to use IDL, wants us to provide Fortran
: source to do the same jobs. Has anyone written a program to run on unix or VMS
: that will translate IDL source (*.pro) into Fortran? I recognize that the IDL
: approach is sufficiently different from Fortran that it might be very hard to
: do a complete job automatically, but Fortran equivalents to IDL are sometimes
: given in the IDL manuals.

: Something analogous to f2c.

: Email or post, will post summary if anything useful appears.

: Dr. David Matthews, Center for Space Physics, Boston U.

Whether it can be done at all (even by hand) depends on several factors,
but essentially if your IDL code contains either of:
1) Arrays whose size is not known until some inputs have been given or
calculations made
2) variables used as more than one type
then there is little chance of being able to generate a fortran code to
do the same thing without MAJOR restructuring -- unless F90 has things to
do this.

Programs which violate (1) but not (2) could probably be re-cast into C
code relatively automatically -- although if it's efficient IDL it would
need a program of compiler standard to avoid turning array operations
into lots of little loops where one big one would be cleared and faster.

The other big stumbling block would be graphics, although if you had a
suitable library you could probably make an IDL plot commands emulator
library.

--
 +-----------------------+---------------------------------- --+---------+
James Tappin,	School of Physics & Space Research	O__
sjt@star.sr.bham.ac.uk	University of Birmingham	-- \/`
Ph: 0121-414-6462. Fax: 0121-414-3722	
 +-- --+---------+

Subject: Re: IDL -> FORTRAN translator?
Posted by Prof. Loren Meissner on Fri, 25 Aug 1995 07:00:00 GMT
View Forum Message <> Reply to Message

sjt@xuna.sr.bham.ac.uk (James Tappin) wrote:
  ...
> Whether it can be done at all (even by hand) depends on several factors,
> but essentially if your IDL code contains either of:

> 1) Arrays whose size is not known until some inputs have been given or
> calculations made

> 2) variables used as more than one type
> then there is little chance of being able to generate a fortran code to
> do the same thing without MAJOR restructuring -- unless F90 has things to
> do this.
>

F90 has allocatable arrays which SHOULD make #1 relatively easy.

F90 can handle "generic" variables ONLY AS PROCEDURE ARGUMENTS: you
can write a "user-defined generic procedure" in a module with a

separate specific procedure for each argument type: see "Programmer's Guide to F90", second edition, p. 241 -- OR -- my "Fortran 90", p. 438 -- OR -- Ellis etal "F90 Programming", p.370.

- Loren Meissner