Subject: Line Profiling in IDL
Posted by Eric Tittley on Wed, 08 Dec 2004 16:29:03 GMT
View Forum Message <> Reply to Message

Hi,

Is there a tool or set of tools for line profiling of IDL code?

Line profiling reports the execution time spent on each line of source code. IDL has a profiling tool, but it only resolves the time spend in functions/procedures.

I had been unable to find anything online so wrote a script to perform line profiling. I should have checked the newsgroup first, since I'm undoubtedly re-inventing the wheel.

Cheers,
Eric

--
Eric Tittley www.hmet.net

Subject: Re: Line Profiling in IDL
Posted by Ben Panter on Fri, 10 Dec 2004 10:04:20 GMT
View Forum Message <> Reply to Message

Hiya Eric,

> Is there a tool or set of tools for line profiling of IDL code?

Not that I know of. I usually just put systime(/seconds) around the line I want to check and compare them, but it's certainly not ideal.

> I had been unable to find anything online so wrote a script to perform
> line profiling. I should have checked the newsgroup first, since I'm
> undoubtedly re-inventing the wheel.

If it's generic, fancy sharing?

I just moved from 150 point spectra to 1500 point spectra, and my routines have gone from 2 minute wonderbeasts to 2 hour slugs...

:(

Ben

--

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

Ben Panter, Garching, Germany
email via www.benpanter.co.uk

Subject: Re: Line Profiling in IDL
Posted by Eric Tittley on Mon, 13 Dec 2004 15:01:49 GMT
View Forum Message <> Reply to Message

Hi Ben,

Since I have heard of no other option (I don't appear to be reinventing fire), I will gladly make the code available. I'll set up something on my web page and post the link on this newsgroup.

Otherwise, you can just grab the necessary files via your ROE account, getting them from:
/home/ert/idl/lib/line_profile/

The script pretty much does automatically what you are doing manually. For each line of a .pro file, it brackets the line with systime(1) commands, adding some other code to store the timing data for the line and to save the timing data. The 'profile' code is stored in another, unique, file. Compile and run the 'profiled' code. This produces a file containing the timing data. The timing data is then processed by an IDL script which reports the lines from the source code that consume the top %90 of the CPU time.

There are explicit instruction on use with the files.

It actually does work! systime() seems to have little overhead, so the profiled code runs almost as fast as the original.

There are some caveats, since the code is very much in the alpha stage. At the moment, it can only handle one function or procedure per file. It looks for the words FUNCTION or PRO and starts there, ending at the first END that starts at the first column (to avoid confusion with END's in CASE or SWITCH statements). Fortunately, this meshes with my programming style, which could very well be another caveat: the code works for code that *I've* written, so it probably includes implicit assumptions concerning what it expects. I'd be very interested to hear how it works for other's code.

BTW, it does deal with continued ($) lines properly.

Cheers,
Eric

Ben Panter wrote:

> Hiya Eric,
>
>> Is there a tool or set of tools for line profiling of IDL code?
>
>
> Not that I know of. I usually just put systime(/seconds) around the
> line I want to check and compare them, but it's certainly not ideal.
>
>> I had been unable to find anything online so wrote a script to
>> perform line profiling. I should have checked the newsgroup first,
>> since I'm undoubtedly re-inventing the wheel.
>
>
> If it's generic, fancy sharing?
>
> I just moved from 150 point spectra to 1500 point spectra, and my
> routines have gone from 2 minute wonderbeasts to 2 hour slugs...
>
> :(
>
> Ben
