Subject: GUI states
Posted by Andrew[2] on Tue, 31 May 2005 04:56:25 GMT
View Forum Message <> Reply to Message

Hi all,

I have written an interface (in IDL) to the AFRL moderate resolution
transmittance (MODTRAN) code and have been testing it for some time
now. I am reasonably happy with it although I do not like how long it
takes to load some of the data (couple of seconds).

The data for the various inputs are all passed around via a pointer
variable which contains a large number of structures, and which also
contain pointer variables (the size of the data arrays is required to
be dynamic in some cases depending on what the user does). This
generally works well except in a few instances where the component
(read card2 and its off-shoots for MODTRAN users) of the application is
using a large chunk of the data in the pointer. I assume this is
because the GUI is holding all this data in memory and then passing it
back and forth as required.

I was considering using the SAVE and RESTORE commands within the GUI to
access the data instead (when required rather than having it always
held in memory) i.e. in my various event handling routines I would
restore the save file, make the required changes to the variable/s in
question and then SAVE the state data. Does anyone know if this would
possibly help speed things up a bit or am I just increasing the amount
of overhead I will have to deal with?

I am asking first before trying as someone may have had similair
problems, and it is a rather large change to make (time consuming not
technically difficult).

Cheers
Andrew

Subject: Re: GUI states
Posted by Thomas Pfaff on Wed, 01 Jun 2005 14:44:02 GMT
View Forum Message <> Reply to Message

Hi Andrew,

> The data for the various inputs are all passed around via a pointer
> variable which contains a large number of structures, and which also
> contain pointer variables (the size of the data arrays is required to
> be dynamic in some cases depending on what the user does). This
> generally works well except in a few instances where the component

> (read card2 and its off-shoots for MODTRAN users) of the application is
> using a large chunk of the data in the pointer. I assume this is
> because the GUI is holding all this data in memory and then passing it
> back and forth as required.
>
> I was considering using the SAVE and RESTORE commands within the GUI to
> access the data instead (when required rather than having it always
> held in memory) i.e. in my various event handling routines I would
> restore the save file, make the required changes to the variable/s in
> question and then SAVE the state data. Does anyone know if this would
> possibly help speed things up a bit or am I just increasing the amount
> of overhead I will have to deal with?
>
> I am asking first before trying as someone may have had similair
> problems, and it is a rather large change to make (time consuming not
> technically difficult).
>
> Cheers
> Andrew
>

I'm not sure, if I understood your problem completely, but I would
definitely say that writing and restoring to disk should take much
longer than manipulating data in memory. Well, as long as you have
enough memory - otherwise the OS would do the swapping automatically.

When passing around variables in a GUI do you use the /NO_COPY keyword
like in

WIDGET_CONTROL, event.top , GET_UVALUE=info, /NO_COPY

Otherwise whatever is stored in the uvalue would be copied before being
assigned to the variable info.

However, if you just pass a pointer around, that shouldn't be the problem...

Cheers,

Thomas

Subject: Re: GUI states
Posted by Andrew[2] on Fri, 03 Jun 2005 01:41:08 GMT
View Forum Message <> Reply to Message

Hi Thomas,

thanks for the info. Thought that might be the case. Yeah I have set
/NO_COPY although I may not have been as rigorous as I should be. I
will double check the code.

Cheers
Andrew

Thomas Pfaff wrote:
> Hi Andrew,
>
>> The data for the various inputs are all passed around via a pointer
>> variable which contains a large number of structures, and which also
>> contain pointer variables (the size of the data arrays is required to
>> be dynamic in some cases depending on what the user does). This
>> generally works well except in a few instances where the component
>> (read card2 and its off-shoots for MODTRAN users) of the application is
>> using a large chunk of the data in the pointer. I assume this is
>> because the GUI is holding all this data in memory and then passing it
>> back and forth as required.
>>
>> I was considering using the SAVE and RESTORE commands within the GUI to
>> access the data instead (when required rather than having it always
>> held in memory) i.e. in my various event handling routines I would
>> restore the save file, make the required changes to the variable/s in
>> question and then SAVE the state data. Does anyone know if this would
>> possibly help speed things up a bit or am I just increasing the amount
>> of overhead I will have to deal with?
>>
>> I am asking first before trying as someone may have had similair
>> problems, and it is a rather large change to make (time consuming not
>> technically difficult).
>>
>> Cheers
>> Andrew
>>
>
> I'm not sure, if I understood your problem completely, but I would
> definitely say that writing and restoring to disk should take much
> longer than manipulating data in memory. Well, as long as you have
> enough memory - otherwise the OS would do the swapping automatically.
>
> When passing around variables in a GUI do you use the /NO_COPY keyword
> like in
>
> WIDGET_CONTROL, event.top , GET_UVALUE=info, /NO_COPY
>
> Otherwise whatever is stored in the uvalue would be copied before being
> assigned to the variable info.

>
> However, if you just pass a pointer around, that shouldn't be the problem...
>
> Cheers,
>
>
> Thomas