Subject: Re: Use of Temporary( ) vs an Optimised Compiler
Posted by Martin Downing on Tue, 27 Nov 2001 10:42:10 GMT
View Forum Message <> Reply to Message

"Craig Markwardt" <craigmnet@cow.physics.wisc.edu> wrote in message news:onwv0cahwo.fsf@cow.physics.wisc.edu...
>
> I agree, Martin. A compiler writer would know exactly when a variable
> on the right hand side is being reassigned. However, your example
> points out at least one of the problems.
>
>> a = 2*a + b/TEMPORARY(a)
>
> Since A appears twice on the right hand side, the compiler would need
> to be smart enough to not overwrite A after its first appearance. In
> fact, I am not sure that IDL makes any guarantees about order of
> evaluation and side effects. Isn't it possible that the TEMPORARY()
> gets called before the first A is evaluated? [Not sure on this, but
> that's why I avoid the situation.]
>
Hi Craig

Oh boy, I had thought IDL followed left to right calculation with usual precidence rules, which I reassured myself of by running the above statement to see whether it crashed! (a good proof or what!) Come to think of it I'm not sure I have found such claims anywhere in the manuals! Dont worry I'd use brackets where it matters anyway. Either way I would have thought the compiler could count the number of instances of the left hand variable on the right and release the memory on the last one. I agree a flag would be good for debugging.

Martin

> Second of all, if the compiler automatically TEMPORARY()'ed every
> variable that was reassigned, it makes debugging harder. What if your
> expression was:
>
>> a = 2*a + F(B)
>
> If F(B) crashed after 2*A was evaluated, then there may be no way to
> recover the original value of A. So, there would probably need to be
> a "debug" vs. "performance" compilation flag.
>
Agreed