Subject: Re: Multi-core techniques
Posted by Maxwell Peck on Fri, 16 Apr 2010 07:35:44 GMT
View Forum Message <> Reply to Message

On Apr 16, 11:15 am, Tim B <tim.burg...@noaa.gov> wrote:
> I am working with various satellite datasets (e.g. Pathfinder SST)
> where
> most of the prior work has been producing 50km resolution analysis.
> Even
> an intensive piece of code over a full 30-odd year data set could
> still be
> finished overnight. The algorithms that I've seen don't excessively
> use
> array processing (no FFT's or such) and there is a lot of data
> independence i.e.
> values over the course of a few days are used rather than values over
> a complete year.
>
> With a move to higher resolution, i.e. 4km, there is significantly
> more data (approx 150 4km
> pixels in a 50km pixel). Given the data independence, my usual
> approach would be to
> create a thread pool around the number of available cores and
> calculate each data
> 'piece' independently. However IDL doesn't expose threads to the
> programmer. So I'd
> value any thoughts about how to take advantage of multicore CPU's
> (heck, even my laptop
> is a dual core machine). My thoughts are:
>
> - use C(?) to manage forking separate processes that start IDL and
> pass parameters to the
> appropriate procedure to run in IDL
>
> - run a number of IDL programs in parallel, the same code but
> processing different
> temporal regions of the dataset. I can start up IDL in different
> windows on an 8-core
> machine and each seems to be a separate process.
>
> - use a different language/architecture completely :-)
>
> I'd be interested to hear from anyone else trying to take advantage of
> multicore CPU's..
>
> Tim Burgess

I've perhaps misunderstood your post but IDL does have an automatic

thread pool. http://idlastro.gsfc.nasa.gov/idl_html_help/The_IDL_Thread_P ool.html

It certainly seems to work for me on a quad core machine in ENVI. The trick is making sure the limits on when multicores are used are chosen properly so it doesn't start paging - I've found this is fairly hard on Windows 32 bit because of the ridiculous memory management. I haven't done extensive testing but there is certainly speedup in a lot of operations when the thread pool is used for ENVI/IDL.

Max
