Subject: Re: Multi-core techniques
Posted by David Fanning on Fri, 16 Apr 2010 02:14:08 GMT
View Forum Message <> Reply to Message

Tim B writes:

> 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..

You might be interested in reading about what Tech-X has
done along these lines to speed up IDL processing:

   http://www.txcorp.com/products/FastDL/

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

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

Subject: Re: Multi-core techniques
Posted by Tim B on Fri, 16 Apr 2010 08:19:12 GMT
View Forum Message <> Reply to Message

> IDL does have an automatic thread pool

"automatic" is correct. If you can use the routines that are
multithreaded,
 you can use multiple cores transparently. If you have lots of
calculations to do
that do not use the multithreaded routines, this all occurs on a
single core,
and thus in a serial fashion.

Thanks for the FastDL link, David. I will investigate the $$$
required.

Tim

Subject: Re: Multi-core techniques
Posted by wita on Fri, 16 Apr 2010 09:02:02 GMT
View Forum Message <> Reply to Message

Dear Tim,

My experience with the IDL thread pool is that for certain type of
operations in ENVI, the thread pool is actually slowing things down.
The most notable example was a "Sum data bands" operation in ENVI
which executed extremely slow. After disabling the thread pool (cpu,
TPOOL_NTHREADS=1), the operation executed several times faster. Maybe
some IDL internals that decide on when to use or not to use the thread
pool, performed poorly in the case.

Another approach on parallelizing your process is to use the
IDL_IDLbridge which allows you to spawn multiple IDL session which can
run on different cores. You may want to have a look at my process
manager, which uses this technique to distribute processing tasks over
several bridges. You can get the code here:
 ftp://sc:image1@ftp.alterra.nl/pub/adewit/process_manager.zi p

Allard de Wit

Subject: Re: Multi-core techniques
Posted by Juggernaut on Fri, 16 Apr 2010 10:47:51 GMT

View Forum Message <> Reply to Message

On Apr 16, 5:02 am, Allard de Wit <allard.de...@wur.nl> wrote:
> Dear Tim,
>
> My experience with the IDL thread pool is that for certain type of
> operations in ENVI, the thread pool is actually slowing things down.
> The most notable example was a "Sum data bands" operation in ENVI
> which executed extremely slow. After disabling the thread pool (cpu,
> TPOOL_NTHREADS=1), the operation executed several times faster. Maybe
> some IDL internals that decide on when to use or not to use the thread
> pool, performed poorly in the case.
>
> Another approach on parallelizing your process is to use the
> IDL_IDLbridge which allows you to spawn multiple IDL session which can
> run on different cores. You may want to have a look at my process
> manager, which uses this technique to distribute processing tasks over
> several bridges. You can get the code here:
> ftp://sc:ima...@ftp.alterra.nl/pub/adewit/process_manager.zi p
>
> Allard de Wit

Other methods include the use of the IDL_IDLBridge to spawn a separate
processes on the other core. I've found this works well if the
overhead of setting up the IDL_IDLBridge object is low.

Subject: Re: Multi-core techniques
Posted by Juggernaut on Fri, 16 Apr 2010 10:49:52 GMT
View Forum Message <> Reply to Message

On Apr 16, 6:47 am, Bennett <juggernau...@gmail.com> wrote:
> On Apr 16, 5:02 am, Allard de Wit <allard.de...@wur.nl> wrote:
>
>
>
>> Dear Tim,
>
>> My experience with the IDL thread pool is that for certain type of
>> operations in ENVI, the thread pool is actually slowing things down.
>> The most notable example was a "Sum data bands" operation in ENVI
>> which executed extremely slow. After disabling the thread pool (cpu,
>> TPOOL_NTHREADS=1), the operation executed several times faster. Maybe
>> some IDL internals that decide on when to use or not to use the thread
>> pool, performed poorly in the case.
>
>> Another approach on parallelizing your process is to use the
>> IDL_IDLbridge which allows you to spawn multiple IDL session which can

>> run on different cores. You may want to have a look at my process
>> manager, which uses this technique to distribute processing tasks over
>> several bridges. You can get the code here:
>> ftp://sc:ima...@ftp.alterra.nl/pub/adewit/process_manager.zi p
>
>> Allard de Wit
>
> Other methods include the use of the IDL_IDLBridge to spawn a separate
> processes on the other core. I've found this works well if the
> overhead of setting up the IDL_IDLBridge object is low.

<--- needs to read previous posts more carefully....

Subject: Re: Multi-core techniques
Posted by natha on Fri, 16 Apr 2010 13:54:24 GMT
View Forum Message <> Reply to Message

Hi Allard,

I'm trying to understand your code but it's not clear for me.
It seems that you keep the input variables in an object and then you
put your specific code into cgi_process_client...

Anyway, I don't understand at all. Could you give us an example of how
to use your library ?. For example, if I have this program:

```
FUNCTION SUM, a
  RETURN, a++
END
```

And I've to run it 1000 times using different input parameters and my
computer has 4 CPUs, how could I use your library ?
Thanks for your help,

nata