rtrace multiprocessing option – initial test results

I have had a chance to test out the new multiprocessing option (-n) for rtrace — introduced in the Radiance HEAD distribution on Dec 15, 2009 — and am not getting the results I anticipated. This page includes my assumptions and results (and questions!).

I had assumed that simulation (real) times would decrease as I increased the number or processor cores allocated to the calculation. The calculation consists of a "grid" calculation of 630 points, sampled for illuminance in a complex daylit space. The simulation command was as follows:

```
time cat numeric/calcplane.pts | rtrace @calc.opt -x {n} -n {n} -faa -h -ov -I -dv-
octrees/test.oct | rcalc -e '$1=$1*4.408051709+$2*11.14367512+$3*1.077778939' >>
out/run{n}.pts
```

Where {n} represents the number of processors available. The idea was to iterate on the grid points, calculating irradiance, converting to illuminance (in footcandles), and saving to a file, and increasing the amount of processors available to rtrace by one per iteration, up to 15. The system is a Mac Pro with two hyperthreaded quad core Intel Nehalem processors, essentially providing sixteen cores in the same physical system. Radiance jobs on this system have historically scaled nearly linearly when loaded with

equivalent Radiance simulations with rpict (rad -n{n}) or mkillum (mkillum -n{n}) jobs. I have to confess ignorance on the effect of setting the horizontal resolution (I don't fully understand the impact of flushing the output); but rtrace gives a warning about setting -n to a number higher than the horizontal resolution (set with -x), so for this test run I simply set -x = -n. The simulation options set in calc.opt are as follows:

```
-av 0.2 0.2 0.2 -ar 60 -aa 0.12 -ad 512 -as 256 -ab 4 -dp 256 -ds 0.2 -dj 0.5 -dt 0.2 -
dr 1 -dc 0.5 -sj 0.7 -st 0.1 -lr 6 -lw 0.002
```

Below are the results from my first pass:

As you can see, the real time as reported by the time utility is pretty consistent from 1 to 15 processors. I expected some overhead, but also some speed gains in the real time for the simulation. But this chart shows that the final results take as long with 15 processors as they do with one, and overhead linearly increasing.

Curious about the effect of setting the horizontal resolution (or not), I did a second pass with no -x setting, which means "no output flushing will take place". Here are the results of that run:

This round shows a slight improvement in real time needed for the simulation, but very slight, and diminishing returns occur almost immediately.

Mainly out out of curiosity (desperation?), I enabled ambient caching for a third run, and those results are shown below (for this run, -x = -n as well, as in the first run; note also the change in time scale):

Here, I'm seeing the trend I would have expected, with a rapid improvement in real execution times, a gradual diminishing return, and a steady increase in overhead as the number of processors increases. I sort-of expected to see a similar graph with no ambient caching, but with a slightly different slope and higher simulation times across the board. Instead, there seems to be almost no net gain in real simulation times.

For ease of comparison, here are all all the real times for all three scenarios plotted:

As a quick gut check, I split the 650-point grid file into two files with 315 points each and ran rtrace once on each file. The end result was that the real time was nearly identical for each run, even though they had half the original number of points. Finally, I did the same calculation on one single point, and that took 3 minutes. I would have assumed the 315 points to take roughly half the time as the 650 points to calculate, and the single point to be computed very quickly. Clearly, there is a flaw in my understanding of how rtrace works, or my test case is not ideal. Probably both.

UPDATE:
As usual, the members of the radiance-online.org mailing list provided swift feedback. Andrew McNeill writes:

> “Hi Rob,
>
> When there is no ambient file there is usually an ambient cache created in memory. If you don’t have an ambient file each rtrace process will build its own cache. I’m guessing that each process spends most of the time building the cache with a first point and the subsequent calculations are very quick. Because getting the cache to saturation is the majority of the calculation time and each processor is doing this independently before really getting going on the points it won’t matter if you have 1 or 16 processors, the simulation will take the same amount of time.
>
> To completely turn off the ambient cache use -aa 0. I expect you will see a linear correlation between number of processes and simulation times, but the sim times will be much much longer.
>
> On a side note, I’m very jealous of your 16 core mac pro!
>
> Andy”

So, there you have it. The thing to take away here is that one would almost certainly take advantage of ambient caching in normal practice, so of course this new feature of rtrace works great.

Leave a Reply

Your email address will not be published. Required fields are marked *

Name *

Email *

Website

Comment

You may use these HTML tags and attributes: `<a href="" title=""> <abbr title=""> <acronym title=""> <b> <blockquote cite=""> <cite> <code> <del datetime=""> <em> <i> <q cite=""> <strike> <strong>`

Post Comment

- [ ] Notify me of follow-up comments by email.
- [ ] Notify me of new posts by email.

We ain't got no blueberries.

Overheard

"I do meditation at that hour of the morning, and I was irritated at having it interrupted."*Creepy hypocritical hippie guy in the final "NYPD Blue" episode who reminds me of half the population of Boulder*

Obligatory Tag Cloud

snow technology death Yankees laffs love humor music Outlook reviews hiking cycling Hooper complaints photography pets Colorado baseball ray tracing commute cats Radiance rants Apple dogs travel cancer media Ellie family politics idiots Boulder aviation Emma friends 14ers blog sustainability stupidity climbing New Jersey Bush computers people

Content Categories

Select Category

Archives

Select Month

Hate Mail

- Pedro padin on An Old Friend Comes to Visit
- John Mills on Bil
- Spiffy Joe on Bil

Meta

- Log in
- Entries RSS
- Comments RSS
- WordPress.org

Die, Spammers!

44,307 spam
blocked by **Akismet**

Proudly powered by WordPress