

Analyzer example: Mandelbrot

This page shows commands and screenshots to illustrate the usage of the Studio analyzer, to resolve a scaling problem for the Mandelbrot example.

1. Pre-requisites: you must have an OpenMP version of the Mandelbrot example, which a 'schedule(runtime)' in the OpenMP for/do loop, compiled with '-xopenmp'. You should also comment the line, where writepng() is called (we don't want to time the writing of the PNG image!).
2. Run a first test, with 1 to 8 threads:

```
$ echo "#thrds wall user sys"; for t in 1 2 4 8; \  
do echo -n " $t "; \  
OMP_NUM_THREADS=$t time -f "%e %U %S" ./mandelbrot ; done
```

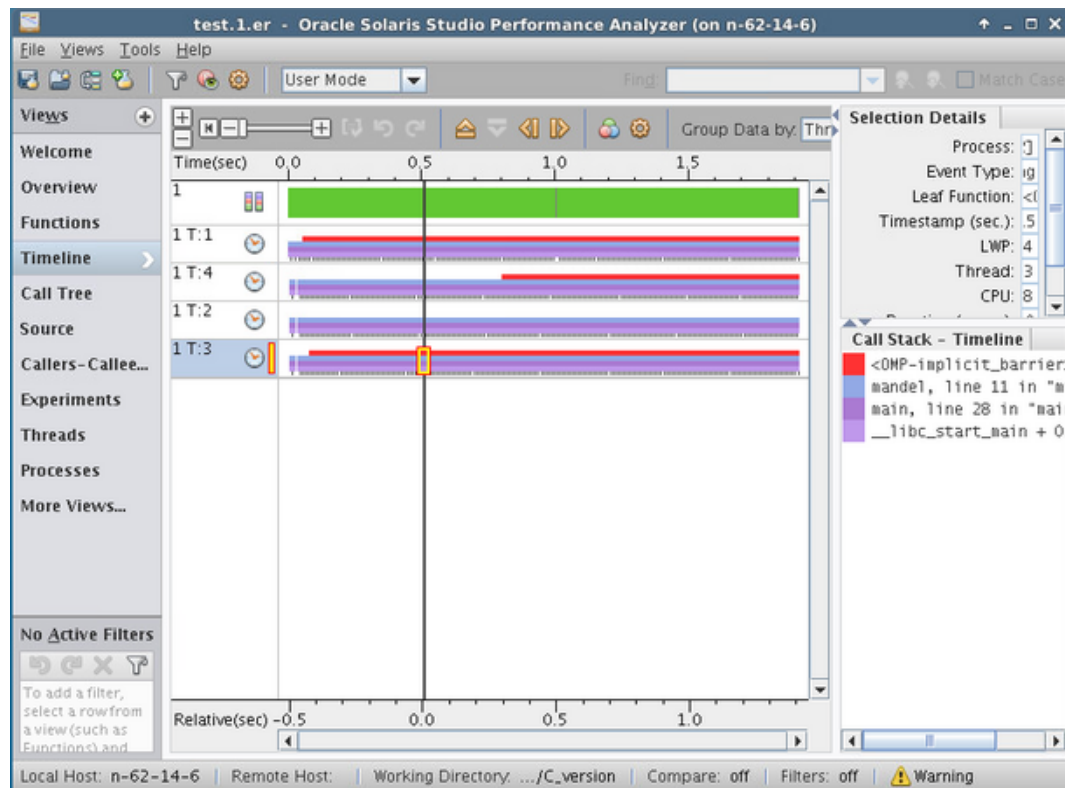
```
#thrds wall user sys  
1      2.70 2.68 0.02  
2      1.92 2.70 0.01  
4      1.88 2.71 0.00  
8      1.03 2.73 0.02
```

The run time (wall) is reduced by adding more threads, but the scaling is not linear!

3. Let's do a first analyzer experiment, using 4 threads:

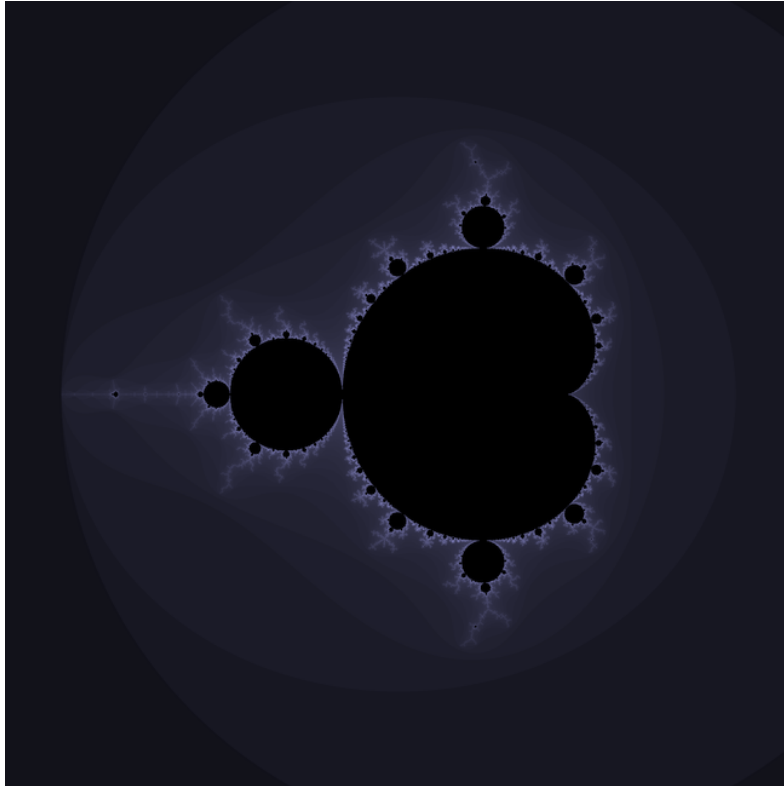
```
$ OMP_WAIT_POLICY=active OMP_NUM_THREADS=4 collect ./mandelbrot  
Creating experiment database test.1.er (Process ID: 20107) ...
```

```
$ analyzer test.1.er
```

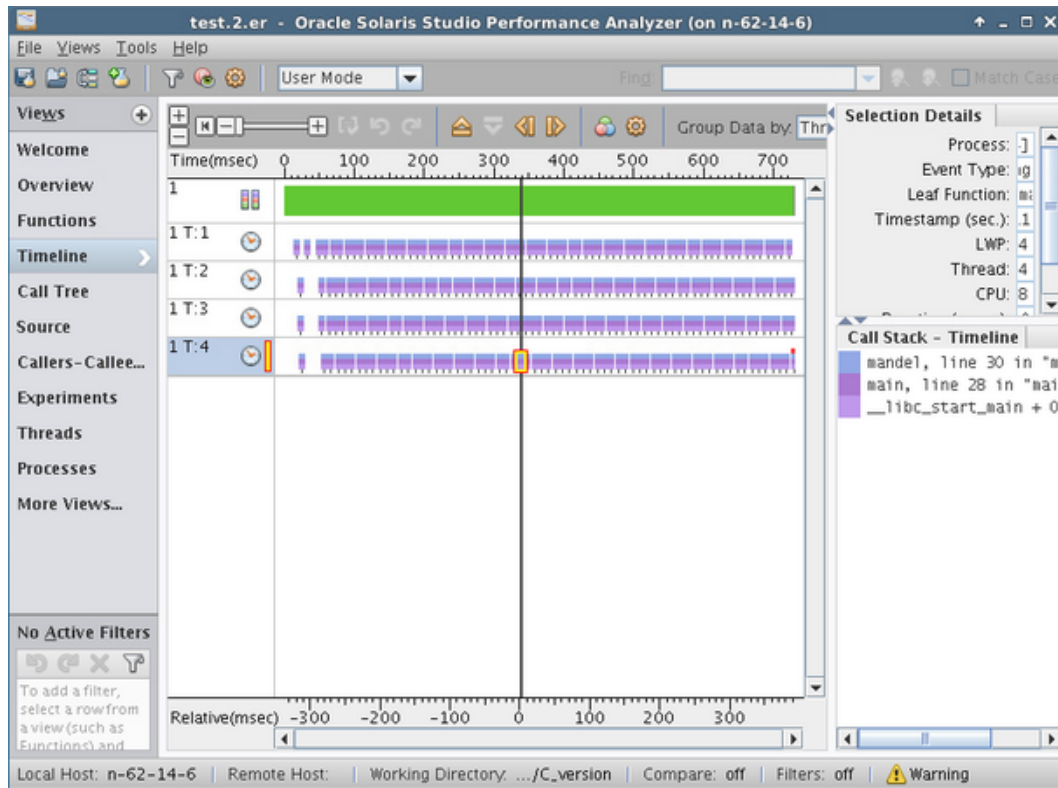


In the timeline above, the OpenMP implicit barrier has been color coded in red, to make it more visible. As we can see, there is a load balancing issue, i.e. the 4 threads are doing different amounts work.

This can be easily seen when looking at the Mandelbrot set - light pixels take more time:



4. Let's now adjust the schedule, using the OMP_SCHEDULE environment variable - and re-rerun the analyzer experiment above:
\$ OMP_WAIT_POLICY=active OMP_NUM_THREADS=4 OMP_SCHEDULE=dynamic,50 collect ./mandelbrot
Creating experiment database test.2.er (Process ID: 28634) ...
\$ analyzer test.2.er



As we can see in the timelines above, the red spots, representing the implicit OpenMP barriers, are gone, i.e. the work is scheduled in a better way to our resources.

5. Now that we know how to get better scheduling with dynamic, let's re-run our little experiment from above (2.) again:

```
$ echo "#thrds wall user sys"; for t in 1 2 4 8; \
do echo -n " $t "; OMP_NUM_THREADS=$t OMP_SCHEDULE=dynamic,50 \
time -f "%e %U %S" ./mandelbrot ; done
```

```
#thrds wall user sys
1      2.70 2.69 0.00
2      1.35 2.68 0.01
4      0.68 2.69 0.01
8      0.35 2.72 0.02
```

That's good news: we get close to perfect scaling!