

# Q1 - Benchmarking SATI.c

---

*Dean Yockey*

## My computer

My personal computer is a Lenovo ThinkCentre I bought for \$75 on eBay. It's running the Linux Mint operating system. When running `lscpu`, the first few lines read:

```
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Address sizes:         39 bits physical, 48 bits virtual
Byte Order:            Little Endian
CPU(s):                4
On-line CPU(s) list:   0-3
```

My computer has 4 CPUs.

## SATI.c

I compiled and ran SATI.c after commenting out all printing statements, aside from one for reporting time after the stopwatch stops.

```
if (!id) {
    printf ("T: %8.6f\n", elapsed_time);
    fflush (stdout);
}
```

## Benchmarking

I created a bash file for benchmarking. It reads:

```
#!/bin/bash
echo "Q1" > "q1log.txt"
for p in {1..10}
do
    echo "P: $p" >> q1log.txt
    for i in {1..20}
    do
```

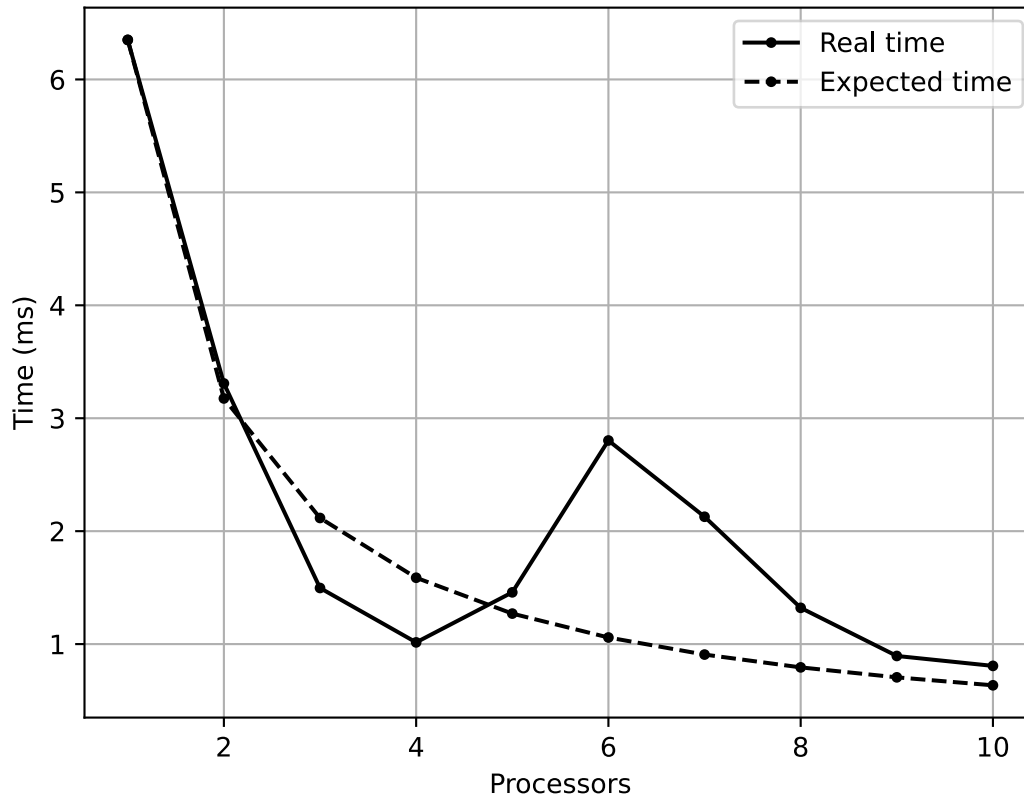
```
mpirun -n $p --oversubscribe sati >> q1log.txt  
done  
done
```

I ran the program 20 times for each process count. The output to `q1log.txt` looked like:

```
Q1  
P: 1  
T: 0.003368  
T: 0.002617  
T: 0.002598  
T: 0.007540  
T: 0.009259  
T: 0.009185  
T: 0.002607  
T: 0.003049  
T: 0.006355  
T: 0.006347  
T: 0.009194  
T: 0.009211  
T: 0.009220  
T: 0.002878  
T: 0.007891  
T: 0.004834  
T: 0.003291  
T: 0.009180  
T: 0.009191  
T: 0.009187  
P: 2  
T: 0.001403  
T: 0.002068  
T: 0.004863  
...
```

## My results

To extract the results from the log file, and visualize them in a line graph, I wrote a python script. I used regex and matplotlib. This is my graph.



## Takeaways

I expected the program to have peak performance at 4 processors, since that's the number of processors my computer actually has, but it actually has the best performance at 10. It's also interesting that at 3 and 4 processors, the real performance is better than the expected performance.

Additionally, 5, 6, 7, and 8 processors perform worse than either 4 or 9 processors. It's hard to guess why these middle numbers have worse performance than either side of them.

It's certainly possible, since the task of testing circuit satisfiability takes a variable amount of time (due to short-circuiting), that some process counts just had very lopsided distribution by chance (perhaps process 3 got all the long ones, for example). That's my best guess.

I performed 20 tests for each count, and closed as many background programs as I could, so I doubt I can practically push my tests to be much more accurate, but I also doubt my computer is well suited for this kind of test.