Code Profiling and Benchmarking

By Thomas Coard

What is Time? Kantianism

“[Kant] describes time as an a priori notion that, together with other a priori notions such as space, allows us to comprehend sense experience.”

https://en.wikipedia.org/wiki/Philosophy_of_space_and_time

Oh, Wrong Workshop...

What is Time? Shell Time

real/total/wall: the real time it took to run the program (according to the time on a clock on a physical wall).

user: the time the computer spent running just your program, without including external factors such as operating system startup times.

sys: the time spent within the program during system-related tasks such as memory allocation.

Profilers

Python has cProfile, memory_profiler, line_profiler

C++ has gprof

R has lineprof

They are all largely measuring the same things. But some have more or less features.

cProfile Example

python -m cProfile [-o output_file] [-s sort_order] (-m module | myscript.py)

```
   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
     26/5    0.000    0.000    0.051    0.010 <frozen importlib._bootstrap>:1002(_find_and_load)
        3    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1033(_handle_fromlist)
       26    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:112(release)
       26    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:152(__init__)
       26    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:156(__enter__)
       26    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:160(__exit__)
       26    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:166(_get_module_lock)
       26    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:185(cb)
     36/6    0.000    0.000    0.043    0.007 <frozen importlib._bootstrap>:220(_call_with_frames_removed)
      428    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:231(_verbose_message)
        1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:241(_requires_builtin_wrapper)
       15    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:35(_new_module)
       26    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:351(__init__)
       39    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:385(cached)
       25    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:398(parent)
```

What is Time? cProfile

ncalls: the number of calls.

tottime: the total time spent in the given function (and excluding time made in calls to sub-functions)

percall: is the quotient of tottime divided by ncalls

cumtim: is the cumulative time spent in this and all subfunctions (from invocation till exit). This figure is accurate *even* for recursive functions.

percall: is the quotient of cumtime divided by primitive calls

From https://docs.python.org/3/library/profile.html

Other Profilers

memory_profiler

python -m memory_profiler script.py

line_profiler

kernprof -l script.py

python -m script.py.lprof

Advice

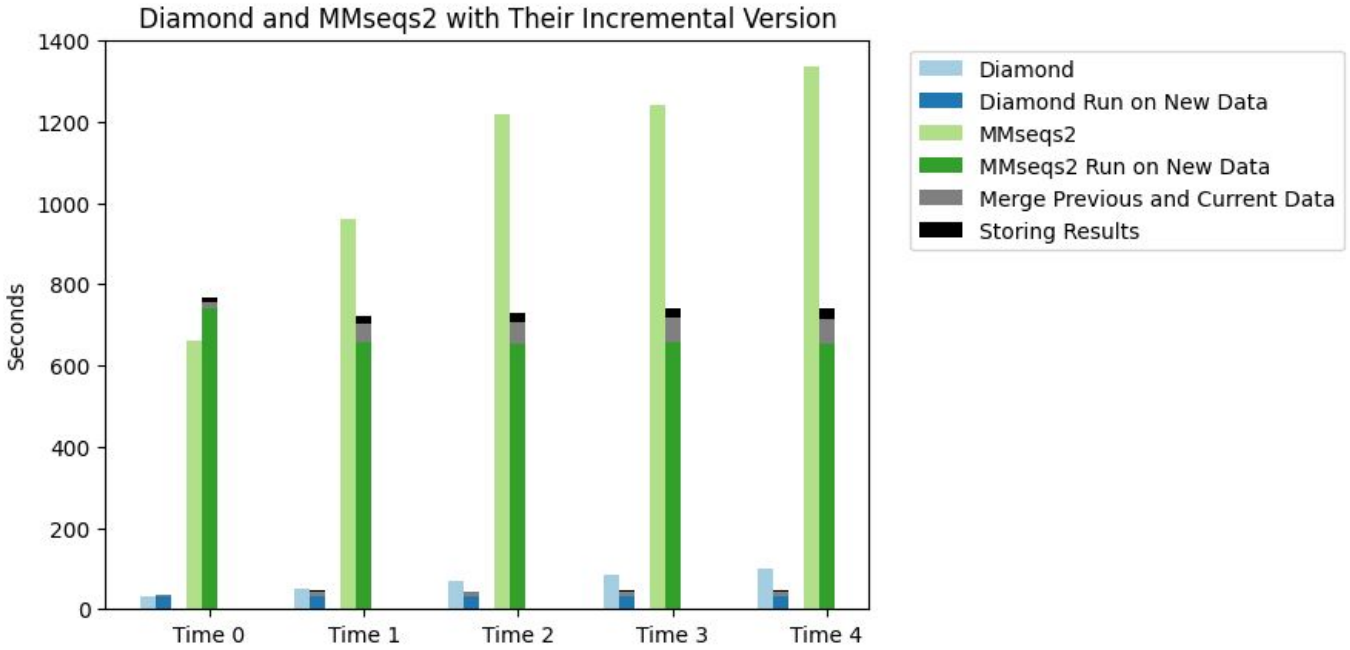
Loops are slow!

- If there is a built-in function that can replace a loop, it is almost always faster.

Search for fast libraries.

- Some libraries allow for better CPU utilization and more efficient calculations.
- Numpy will utilize multiple CPU cores and does matrix mathematics quicker compared to python's lists.
- Numba can speed up Numpy in some circumstances, but might run slower in others.

Example Use


Diamond and MMseqs2 with Their Incremental Version
Seconds
0
200
400
600
800
1000
1200
1400
Time 0
Time 1
Time 2
Time 3
Time 4
Diamond
Diamond Run on New Data
MMseqs2
MMseqs2 Run on New Data
Merge Previous and Current Data
Storing Results

Benchmarking on Picotte

seff: “takes a jobid and reports on the efficiency of that job's cpu and memory utilization.”

sacct: “displays accounting data for all jobs and job steps in the Slurm job accounting log or Slurm database”
https://slurm.schedmd.com

Extra Material:
https://proteusmaster.urcf.drexel.edu/urcfwiki/index.php/Slurm_Utility_Commands
https://proteusmaster.urcf.drexel.edu/urcfwiki/images/URCF_Workshop_Nov_2021.pdf