Energy Profiling

Patrick May

College of Wooster 04/24/2023

Overview

- Increasing demand for *energy efficient* software
- Abstraction makes it difficult to understand energy efficiency
- Faster does not ***always*** mean more energy efficient

GOAL: understand energy usage throughout runtime

Energy Measurement

- External
  - Multimeter
  - Sit between power source and device
- Internal
  - Software API
  - Access readings of hardware sensors
- Limitations
  - architecture, manufacturer
  - OS, containment

Profiling

Statistical

-outside target

-sampling, interrupts

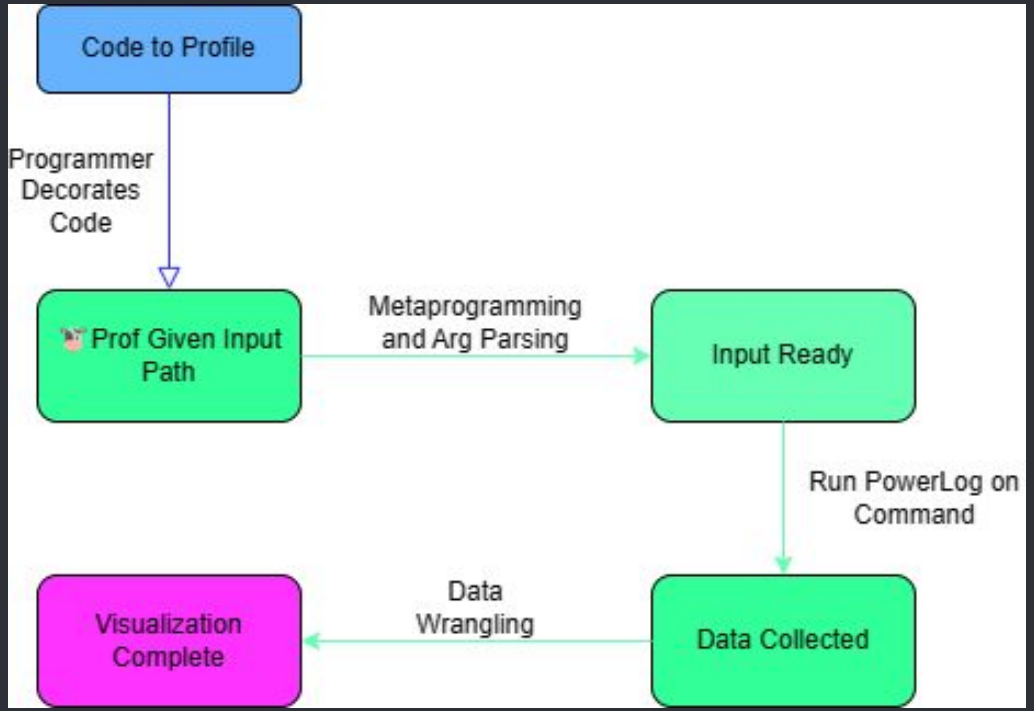
-only an approximation

Instrumental

```
fn func_to_profile(*args, **kwargs){
    // "instrumental" code that takes and log metrics
    datafile.log(timestamp,
        measure1(),
        measure2(),
        measure3(),
        )
    // pre-existing code of function here
    ...
}
```

🐮CowProf🐮

- Script and CLI tool to profile prepared code

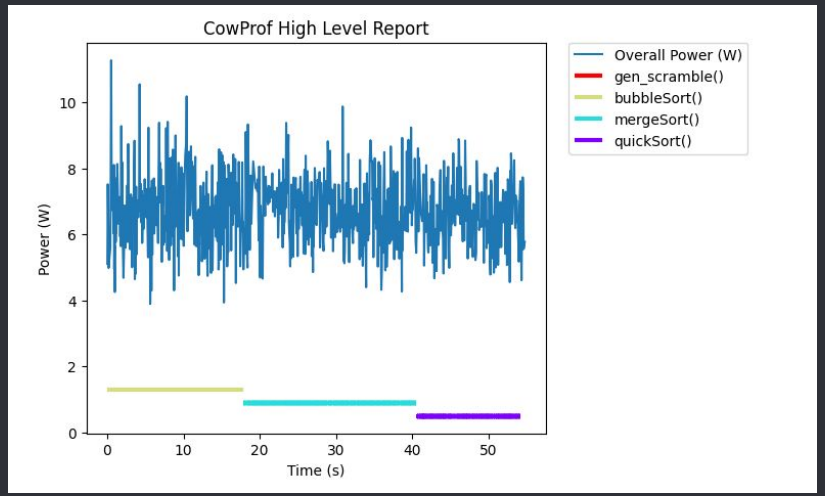


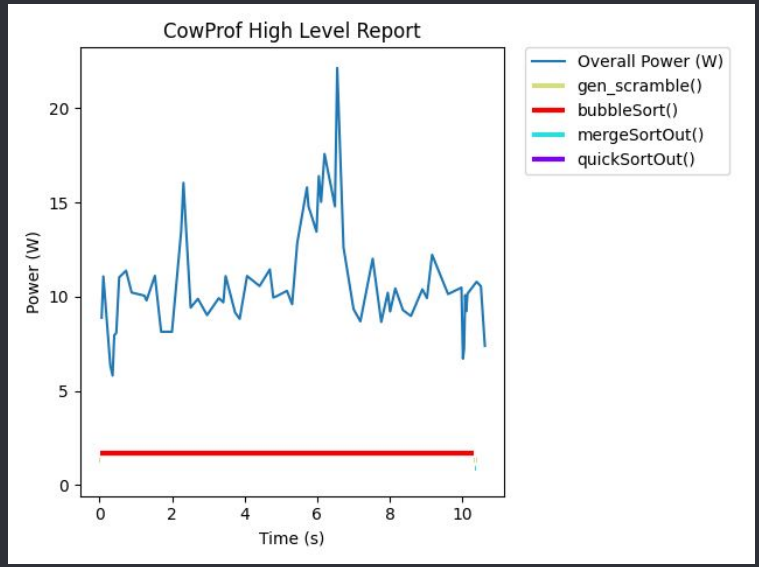
CowProfiler Implementation

- Higher-order functions
  - Functions that take other *functions* as their input
- Metaprogramming
  - Using programs to write programs
- Data Interpolation
  - Wrangled with polars

Software Demo

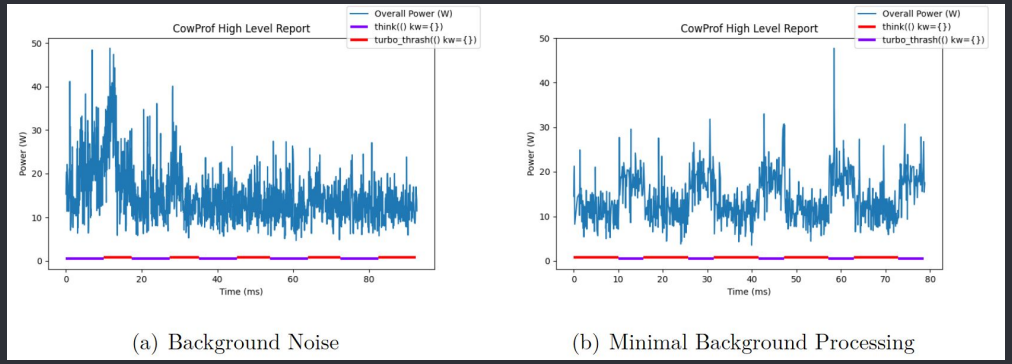
Experimenting w/ CowProf





Limitations

- System Noise
- Implementation
  - Visualization prioritized over statistical rigor
  - Visualization breaks down with large number of tracing functions
- Other Concerns
  - Injective Overhead
  - Cache (L0, Branch-Predictor, etc)



(a) Background Noise (b) Minimal Background Processing

Next Steps

- Extend languages, systems, architecture that CowProf works on
- Overhead Reduction
- Further language-internal investigations
  - *energy optimal* functions
  - multithreading power usage