New Developments in BenchExec

Philipp Wendler

LMU Munich, Germany

@ CPAchecker/LDV Workshop, Moscow, 2018-09-25

Overview

- Energy measurements
- RAM disks for temp files
- Metadata format

Why measure energy?

- Verification consumes a lot of energy (Apollon: ~50 000 kWh per year)

Why measure energy?

- Verification consumes a lot of energy (Apollon: ~50 000 kWh per year)
- "Green" verification needed

Why measure energy?

- Verification consumes a lot of energy
  (Apollon: ~50 000 kWh per year)
- “Green” verification needed
- You cannot improve what you cannot measure!

How to measure energy: RAPL

- Intel Running Average Power Limit
- API for accessing energy-consumption counters
- Available in common Intel CPUs
- Measurements per CPU and per CPU component (cores, GPU, memory controller)
- Resolution $\sim 10^{-5}$ J (e.g., 10 mW for 1 ms)
- No official statements on precision and accuracy, but experiments found good accuracy

How to measure energy: RAPL

- Intel Running Average Power Limit
- API for accessing energy-consumption counters
- Available in common Intel CPUs
- Measurements per CPU and per CPU component (cores, GPU, memory controller)
- Resolution $\sim 10^{-5}$ J (e.g., 10 mW for 1 ms)
- No official statements on precision and accuracy, but experiments found good accuracy

But:

- Granularity per CPU (not per core)
- Cannot distinguish between active processes

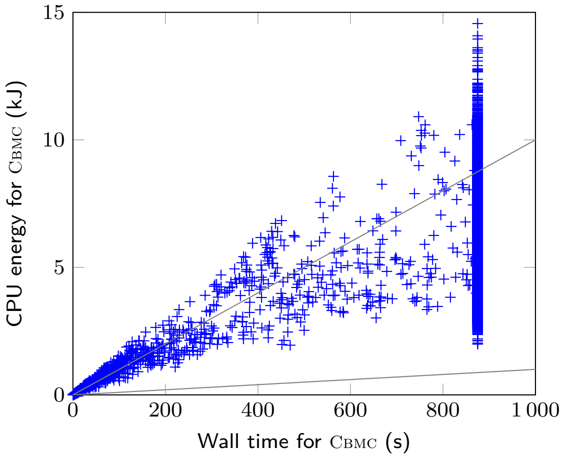
CPU Energy Meter

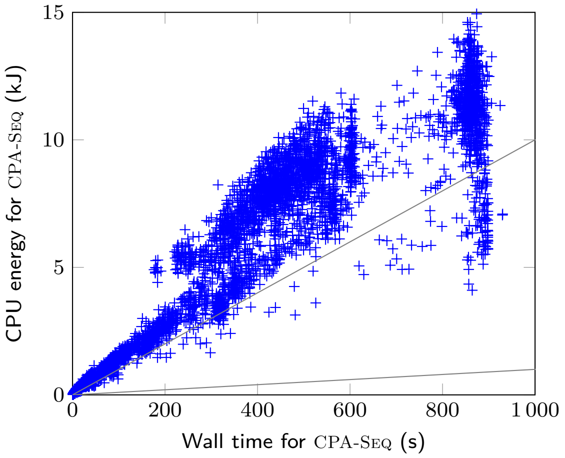
- Easy to use command-line tool for reading RAPL values
- Available as Debian package,
  no manual configuration necessary
- Integrated in BenchExec
  (will be used automatically if installed,
  only if whole CPUs are used)
- Installed and usable on most VerifierCloud workers

Evaluation

- Can energy measurements give new insights when comparing algorithms/tools?
- Experiment:
  - Official SV-COMP'18 results measured on Apollon
  - Energy usage of whole CPU (results for CPU components similar)

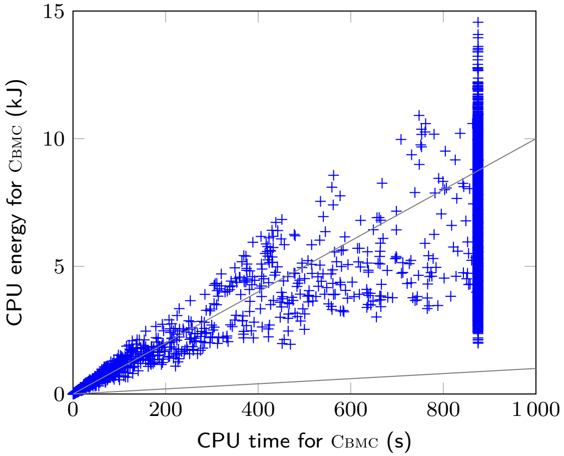
Power usage varies across tools and runs

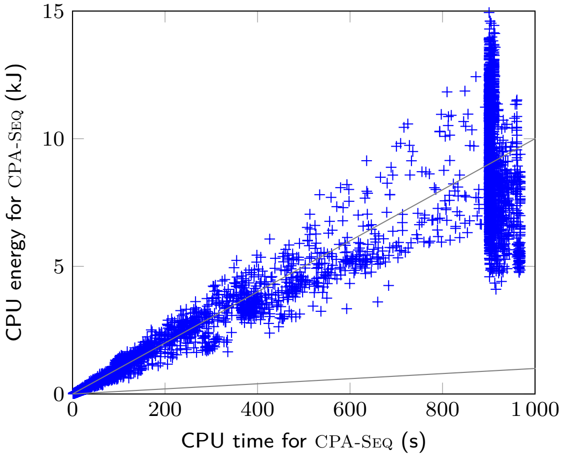




Lines through origin represent average power
(gray lines: 1 W and 10 W)

High values for CPU time do not imply high values for energy





"CPU-Power" usage between 2 W and 16 W

Summary

- Energy-aware research is important
- Energy measurements more difficult than time measurements
- Time is not a good proxy for energy
- At least measure energy as far as possible (and discuss in evaluation)
- BenchExec + CPU Energy Meter leave no excuse ;-)

RAM Disks for Temporary Files

- Tools with temp files are problematic:
  - I/O has nondeterministic performance
  - Restricting size of temp files is imprecise and creates overhead

RAM Disks for Temporary Files

- Tools with temp files are problematic:
  - I/O has nondeterministic performance
  - Restricting size of temp files is imprecise and creates overhead
- Solution: use RAM disk for all temporary files (`tmpfs`)
  - RAM-disk usage is included in memory limit and memory measurements
  - Also applies to output files of tool
  - Only possible in container mode
  - Only for `--hidden-dir` and `--overlay-dir` (default), not for `--full-access-dir`
  - Will soon be implemented in BenchExec

Why New Metadata Format?

Goals:

- Verification of arbitrary properties (LTL formula) instead of fixed known string
- Verification of several properties in single run, with verdict per property
- Verification of tasks with several input files
- Verification of Java tasks (with source directory and library JARs)
- Tool-independent way to specify information about tasks (e.g., machine model, language)

Task-Template Files

C program:

```
format_version: 0.1
input_files:
  - task1.c
  - task2.c
required_files: task.h
properties:
  ...
```

Java program:

```
format_version: 0.1
input_files:
  - src/
  - lib/*.jar
properties:
  ...
```

Properties in Task-Template Files

```
...
properties:
  - property_file: unreach-call.prp
    expected_verdict: true
  - property_file: termination.prp
    expected_verdict: false
    # future possibilities:
    #violation_location: ...
  - property_file: memsafety.prp
    expected_verdict: false
    subproperty: invalid-free
  - property_file: ltl.prp
    expected_verdict: true
```

Use in Benchmark Definition

Direct replacement of input files:

```
<tasks>
  <include>task.yml</include>
  <propertyfile>unreach−call.prp</propertyfile>
</tasks>
<tasks>
  <include>task.yml</include>
  <propertyfile>ltl.prp</propertyfile>
</tasks>
```

Background Concepts

- All tasks related to single program in same YAML file (to avoid repetition)
- Property in benchmark definition is used to select tasks from template
- BenchExec does not read property file anymore, name of property file is used to identify property
- BenchExec does not interpret result strings anymore
- Property files unchanged, no change to verifier necessary
- One verdict per property file

State & Future

- WIP: branch `yaml`
- Extend benchmark definition with several property files
  Provide attribute to select between
  - Create run for every property
  - Verifier is given all properties in single run
- Allow additional attributes
  and pass them to the tool-info module
  (e.g., machine model)