

## Literaturanalyse

- Ohne Vorhersagemodell (nur Messungen) oder mit Vorhersagemodell (welches trainiert werden muss)
- Mit Modell:
  - Würde den Umfang der Arbeit sprengen
  - Lieber auf die Validierung eines existierenden Modells mit neueren Methoden konzentrieren
- Architektur: Ein Ansatz kann entweder auf eine Spezial- oder eine Allzweckarchitektur abzielen
- Granularität: Die Ansätze unterscheiden sich in der Granularität, mit der sie messen können
- Annahmen: Die Annahmen, die Ansätze über das gemessene Programm oder seine Umgebung machen
- Prof. Siegmund hat noch keine auswertbaren Daten für seine Messungen veröffentlicht.

## Erkenntnisse

- Es gibt kein fertiges Standard-Framework, welches man einfach verwenden kann.
- Es wird viel mit wenig Strom verbrauchenden Prozessoren gearbeitet (ARM, Mikrocontroller-Architekturen, x86 mit geringem Energieverbrauch)
- Messungen mit Serverprozessoren wie der Intel Xeon Reihe kommen fast gar nicht vor.
- Ich würde mich auf x86-Laptop/Desktop/Server-Prozessoren beschränken, da ich dort die meisten Erfahrungen habe
- Direktes Messen vom Stromverbrauch von Methoden ist nur auf primitiven Mikrocontrollern möglich
- Es besteht die Möglichkeit den Stromverbrauch des ganzen Computers, der einzelnen Komponenten zu bestimmen oder nur den der CPU. Man misst entsprechend direkt von der Hardware, nutzt den Stromverbrauch der Batterie oder nutzt ein Modell zur Schätzung
- Der modernste Weg ist die Energieverbrauchsmetriken der CPU zu verwenden über RAPL (Running Average Power Limit). Das war bis vor kurzem nur bei Intel Prozessoren möglich, ist aber auch in modernen AMD Prozessoren enthalten.
  - Blendet andere Komponenten aus, aber solange die Applikation nicht grafisch (Display und GPU) ist, ist es trotzdem repräsentativ
  - Speichernutzung kann anhand des Energieverbrauchs des DRAM-Speichercontrollers ermittelt werden
  - Messungen werden rund alle 10-100ms aktualisiert
- Energiemodelle für Java-Bytecode existieren

- wurden aber nie mit Hardware validiert
- Methodengranularität scheint das häufigste und machbarste zu sein
- Für Mobile Geräte
- oft nicht fortgeführt und kein Quellcode vorhanden
- Energiemodelle für C
  - basieren darauf eine Methode oft hintereinander auszuführen, sodass eine Energiemessung über Zeit erfolgen kann
  - gibt auch Ansätze über den AST
  - Es werden auch unter AOP einzelne Sortieralgorithmen verglichen
- Alle Ansätze mit beigelegtem Quellcode leiden unter der Tatsache, dass Programme nach Veröffentlichung des Papers nicht mehr weiterentwickelt werden. Ältere Benchmarks sind häufig selbst gebaut und können so nicht reproduziert werden. Deshalb möchte ich möglichst viel Standard- bzw. FOSS-Software verwenden.
- Angaben verbrauchter Joules kann man nicht übernehmen, da der Energieverbrauch eines Prozessors vom Modell zu Modell unterschiedlich ist. Validierung muss so über das Verhältnis des Verbrauchs verschiedener Instruktionen zueinander erfolgen oder mit einer anderen Methode wie das Messen des Batteriestandes
- Alle bisherigen Ansätze über RAPL messen den Energieverbrauch der gesamten CPU zum Zeitpunkt x, Interferenzen von anderen laufenden Programmen kann nicht ausgeschlossen werden. Scaphandre ist ein auf RAPL aufbauendes Framework welches den Energieverbrauch der Prozesse trennen kann, d.h. man kann einen einzelnen Prozess betrachten. Scaphandre ist neu auf GitHub und hat noch keine Publikationen, ich würde meine Arbeit auch gerne nutzen darauf aufmerksam zu machen.

## Zur Verfügung stehende Hardware

- Linux-Laptop mit Intel-Prozessor
- Windows-Desktop mit AMD-Prozessor
- Linux-Server mit Intel-Prozessor

## Messsoftware

- Intel RAPL - Running Average Power Limit
  - <https://01.org/blogs/2014/running-average-power-limit-%E2%80%93-rapl>
  - Bibliothek um eine Prozessor-API auszulesen
  - Stromverbrauch eines Prozessors wird in einem Zeitraum in Jouls gemessen
- <https://github.com/hubbl-org/scaphandre>
  - Stromverbrauch eines isolierten Prozesses mit RAPL messbar
- VTune Profiler
  - <https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/vtune-profiler.html>

- Profiler für C/C++/Fortran und Java von Intel
- VisualVM
  - Profiler für Java, zeigt kumulierte verbrauchte CPU-Zeit pro Funktion an
  - Gut um einen Überblick zu bekommen
- <https://kieker-monitoring.net/>
  - Für Feinmessungen, gut einstellbar, Integration mit dem Dashboard bereits vorhanden
- <https://gitlab.com/MarcoCouto/c-lem> <https://gitlab.com/MarcoCouto/serapis>
  - Um Energiekosten von C-Instruktionen zu ermitteln
  - Wurde nach Veröffentlichung des dazugehörigen Papers nicht fortgeführt
- <https://github.com/rouvoy/powerapi>
  - Energieverbrauch auf Prozess-Level
  - kein Update seit 5 Jahren
- Powertop
  - <https://01.org/powertop/>
- Linux Bordmittel
  - power\_now
  - powercap
  - Valgrind
  - gperftools
- Prime95
  - Zum Auslasten aller CPU-Kerne, damit man den maximalen Stromverbrauch des Prozessors ermitteln kann
- Die Evaluierung der gezeigten Ansätze mache ich über das nächste Wochenende.

## Programmiersprachen

- Wenn Genauigkeit auf Methodenlevel erzielt werden soll, muss zwangsläufig eine Eingrenzung auf eine Programmiersprachenfamilie erfolgen
- Ich habe aus meiner Sicht genug Hintergrundwissen in Java und C um dies zu bewerkstelligen, bei anderen Programmiersprachen fehlt mir zu sehr der Kontext
- Java
  - Garbage Collection verursacht Hintergrundrauschen
  - Threading-Modell verursacht unter Linux ggf. Schwierigkeiten bei der Messung
- C/C++/Fortran
  - Erzeugt plattform-spezifische Binaries
  - Kein dynamischer Overhead durch Garbage Collection
  - Genaue Angabe von Optimierungsoptionen nötig um ggf. abweichende Ergebnisse zu verwenden

## Fallstricke

- Hyperthreading
- Frequenzanpassung der CPU
- Sind Größen korreliert?
  - e.g. Laufzeit/CPU-Zeit und Energieverbrauch

## Derzeitiger Plan

### C-basiert

- Benchmarks aus **Statically Analyzing the Energy Efficiency of Software Product Lines** validieren mit **Scaphandre**
  - Nutzt verschiedene C-SPLs und Community Benchmarks
  - Rechnet anhand eines Modells und RAPL die Energiekosten aus, diese werden mit **Scaphandre** validiert
- Energiekosten werden nochmal zusätzlich anhand einer Messung des Batteriestandes meines Laptops validiert
- C-Funktion und Instruktionskosten einer SPL mit Metapher visualisieren

## Wichtigste Literatur

- Survey of Approaches for Assessing Software Energy Consumption
  - Einordnung der verschiedenen Möglichkeiten zum Messen des Energieverbrauchs
- Automating Energy Optimization with Features
  - Frühes Paper vom Prof. Siegmund zum Thema
  - Etabliert grundlegende Zusammenhänge
  - Feature-Oriented Programming
- Statically Analyzing the Energy Efficiency of Software Product Lines
  - C basierend
  - Nutzt abgeleiteten AST (C-Lem, Serapis) und RAPL um Energieverbrauch von Komponenten (Statements, Methoden, ganzes Programm)
  - <https://gitlab.com/MarcoCouto/c-lem>
  - <https://gitlab.com/MarcoCouto/serapis>
- Estimating Mobile Application Energy Consumption using Program Analysis
  - eLens Framework, keine Weiterentwicklung, kein Quellcode auffindbar
  - Für Smartphones
  - Nutzt software environment energy profile (SEEP) vom Hersteller
- Component-Level Energy Consumption Estimation for Distributed Java-Based Software Systems
  - often cited, uses Java
  - Uses Computational, Communication, Infrastructure Energy Consumption for distributed systems
  - Energy consumption estimation framework including static and run-time estimation

- Evaluation over experiment (achives method level granularity)
  - only works with Interpreter-JVMs, not with modern JIT-JVMs
  - states that execution time and energy consumption are not correlated
- CPU Energy Meter: A Tool for Energy-Aware Algorithms Engineering
  - Misst Energieverbrauch via RAPL
  - Vergleicht Sortieralgorithmen
  - <https://github.com/sosy-lab/cpu-energy-meter>
- GreenOracle: Estimating Software Energy Consumption with Energy Measurement Corpora
  - Für Smartphones
  - Basiert auf CPU-Auslastung und Anzahl von System-Calls