**Summary**

This team wants to create tools that helps developers determine the energy consumption of their individual programs. Specifically, tools that can provide fine-grained information about energy consumption the same way performance debugging tools can provide fine-grained about the time and resources occupied by their programs’ subroutines.

This paper describes such a tool called EPerf. EPerf uses linear optimization[[1]](#footnote-1) to build a model that can predict the energy consumption of a given program based on micro-architectural information like cache misses, TLB misses, etc.

This is challenging for two reasons:

* Micro-architectural events occur too briefly for existing process level interfaces to detect.
* Energy consumption varies based on architecture family[[2]](#footnote-2) and not every processor has energy measurement hardware built in.

EPerf will first build a model using existing technology\* to establish the power usage of micro-architectural events on different types of processors. Then it can be used on a particular application to determine its energy use.

\* Something called RAPL (Running Average Power Limit) already exists to help monitor the power usage of processors, but it has some shortcomings, and the technology is not present everywhere.

\* But other processor models also have things called performance counters: special-purpose registers built into processors that can keep track of hardware events. (RAPL has it’s own set of performance counters). These can be used to get fine-grained information about micro-architectural events. This is exact the kind of information EPerf needs, but they currently don’t measure energy consumption.

The current EPerf prototype has been implemented as a shell script and only provides process-level information. The end goal is providing information at the subroutine level.

**A few shortcomings**

The current prototype relies on RAPL which treats a CPU socket as a single domain. Thus, it cannot measure energy use per-core or per-hyperthread[[3]](#footnote-3). I think this means that the prototype cannot measure energy use for multi-core/multi-threaded programs? But I’m not entirely sure.

According to the paper, RAPL counters are also “not updated atomically, which means that users can read a mixture of stale and inconsistent values”. This can be problematic because it means that EPerf may end up using incorrect/state data when building the model which probably leads to the high error margin.

The paper did mention ways they ensured more correct energy readings, but the current prototype predicts energy with an error rate of 1.2%-14% (mean 6%) on single socket servers and an error rate of 3.7%-31.6% (mean 19%) on double socket servers which still seem high to me.

**Potential future directions for the project**

I want to address the issue of different processors and the fact that each processor has a different set of performance counters.

It would be very useful to have some kind of abstraction/set of APIs that would easily allow EPerf’s developers to interface with the data of the program counters of different processors.

The program counters can already tell you what hardware events occurred and presumably the processor can tell you what family it belongs to.

You can then create a table (if it doesn’t already exist) that lists the energy usage of different hardware events depending on which type of processor it occurred on.

Then you can create some API that takes in a specific hardware event and the processor it occurred on as input, searches through the table, and outputs the energy usage for that event. If you put this in a loop for all hardware events (as reported by the performance counters) then you could calculate the exact energy usage for a given program as it runs on a processor.

Diagram

Description automatically generated

I just picked some processor families from here: <https://en.wikichip.org/wiki/x86/list_of_processor_families>

Not sure how useful this is if you’ve already collected all the data you need, but as a programmer this could make collecting data easier.

1. A method to achieve the best outcome (such as maximum profit or lowest cost) in a [mathematical model](https://en.wikipedia.org/wiki/Mathematical_model) whose requirements are represented by [linear relationships](https://en.wikipedia.org/wiki/Linear_function#As_a_polynomial_function). [↑](#footnote-ref-1)
2. Each processor belongs to a “family”. Families are determined by what feature set the processor has. [↑](#footnote-ref-2)
3. According to Wikipedia, Hyperthreading is “[Intel](https://en.wikipedia.org/wiki/Intel)'s [proprietary](https://en.wikipedia.org/wiki/Proprietary_hardware) [simultaneous multithreading](https://en.wikipedia.org/wiki/Simultaneous_multithreading) (SMT) implementation”. [↑](#footnote-ref-3)