Building a Multi-Kernel GPU Frequency Predictor

Background

GPUs have become the de facto standard in High Performance Computing (HPC) and AI training & inference, but with today's sustainability challenges and energy limitations, performance engineers are (and need to be) more conscious than ever about reducing waste, i.e., utilizing the full potential of the hardware.

As such, Dynamic Voltage and Frequency Scaling (DVFS) is used to dynamically tailor the GPU to the workload. By setting a lower clock frequency, the cores in the GPU will execute instructions at a lower speed and with a lower power draw. Because the relation between execution time and power is not always linear, some workloads can maintain the same execution time at lower clock frequencies [1,2]. Doing so intelligently and efficiently however requires intimate knowledge of the GPU, the workload, and the interaction between the two. Knowing *when* to provide the *right* frequency is a nontrivial task.

Moreover, for some optimization goals (like *compute-waste*: max. reduction in energy with no performance loss), improving the energy-efficiency of a *sequence* of kernels can yield better results if they are optimized *together* [1].

Goals

The goal of this thesis is to create a tool which can (1) **predict** the best (core & memory) frequencies for a sequence of kernels, (2) **apply** said frequencies, and (3) **measure** the performance and energy. You will evaluate your tool on a mix of kernels from GPU benchmarks or AI workloads.

In this is performance/energy engineering project you will:

- Implement a tool that can set frequencies and measure the time, power and energy of a (potentially recurring) *sequence* of GPU kernels.
- Experiment with different ways of predicting what the best frequencies will be, e.g. (depending on the time/progress):
    - vary input variables: performance counters, workload characteristics,
    - analytical vs. statistical models (decision trees, neural networks).
- Experiment with different goals/restrictions (reduce energy, reduce power, maintain highest performance), policies (aggressive, conservative), and granularities (#*x* kernels per frequency, *y*-second window).
- Evaluate your tool using varied mix of kernels in order to discover:
    - How accurate is the prediction?

    - How quickly can the tool (re)learn?
    - How much time/power/energy can the tool save?

Requirements

You have experience in (or a strong interest to learn):

- GPU programming (understanding GPU architecture, C/CUDA)
- Benchmarking (devising a methodology, doing measurements (time, power, performance counters))
- Modeling / machine learning (ML)

Supervision

- Ana-Lucia Varbanescu (a.l.varbanescu@utwente.nl)
- Kuan-Hsun Chen (k.h.chen@utwente.nl)
- Jeffrey Spaan (j.p.spaan@utwente.nl)

References

[1] Reducing Compute Waste in LLMs through Kernel-Level DVFS
[2] Going green: optimizing GPUs for energy efficiency through model-steered auto-tuning (arxiv)