GPU colocation for energy

Background

In recent research, colocation for GPU kernels (=running at the same time, a.k.a. hardware partitioning) has made a revival with the arrival of official [1,2,3] and unofficial support [4].

The main benefit of colocating kernels is increasing the utilization of the hardware. The idea is twofold: different kernels can stress different hardware resources, meaning that combined, they stress *more* parts of the GPU than they would individually. Second, some kernels may only utilize part of some (and the same) resource (e.g., 50% of the memory bandwidth); colocating two of these kernels would, ideally, utilize 100%.

In practice, efficiently finding the best 'match' of kernels and deciding how to partition them is hard (especially a priori). Furthermore, although colocation is focused on improving the hardware utilization, traditionally, the ultimate goal of colocation has been to increase throughput [5], not necessarily energy(-efficiency). Today's sustainability challenges require us to also consider the latter.

A different energy-efficiency improvement technique is Dynamic Voltage and Frequency Scaling (DVFS). By lowering the clock frequencies, the hardware utilization for kernels that underutilize the hardware can increase. Although thought to be orthogonal, the impact of DVFS on colocation has not been thoroughly studied.

Goals

The goal of this thesis is to evaluate GPU colocation methods and strategies from an energy-efficiency viewpoint.

In this is performance/energy engineering project you will:

- Build a (small) framework for colocating GPU kernels (from known GPU benchmarks (or your favorite LLM)) and measure the performance & energy. You will combine this data to form meaningful metrics about the efficacy of colocation-for-energy.
- Experiment with different colocation strategies for **matching** and **partitioning** kernels based on simple characteristics (e.g., % bandwidth utilization).
- Experiment with the effect of setting different core & memory clock frequencies for different match-ups.
- Investigate the difference between colocating for performance (throughput or latency) and colocating for energy.

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))

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] CUDA streams
[2] NVIDIA Multi-Process Service (MPS)
[3] NVIDIA Multi-Instance GPU (MIG)
[4] Hardware Compute Partitioning on NVIDIA GPUs
[5] KACE: Kernel-Aware Colocation for Efficient GPU Spatial Sharing