Automated GPU Kernel Transformation as an Optimization Problem

Kristian Kadlubiak, xkadlu01@stud.fit.vutbr.cz

Abstract

The number of kernels in large HPC applications accelerated on the GPU can easily reach a few dozens. In many cases, such kernels use the same data arrays for calculations. By fusing such kernels in to one, data used in both kernels can be stored in the on-chip scratchpad memory. By doing so, the traffic to the off-chip memory is reduced which results in performance boost. This optimization share many similarities with loop merging optimization common in modern compilers. However, there exists some constraints specific to the GPU programing which require different approach to account for.

The presentation starts with the quick introduction into the GPU programing and its specific concepts important in the kernel transformation. Afterwards, we present an basic formal definition of an combinatorial optimization problem used in the kernel transformation.

However, the problem definition on its own is not able to capture all specific aspect and constraints of the GPU programing. Therefore, we present additional tools. Namely two DAG representations of the computation task. The first graph captures relationship between the kernels and the data arrays. The vertices of this graph represent the kernels or arrays and edges between them represent fact that the kernel is using particular data array as an input or output, depending on mutual position. The second graph is derived from the previous representation and captures the order of execution of the kernels. In this graph, the vertices represent kernels and edges represent the order of execution.

Finally, complete formal definition of the optimization problem is presented. This definition uses both graph representations and captures all constraints of GPU programing as well as hardware limitations.

At the end of the presentation, we present proposed methods of optimization problem solving with few real-life examples and discuss disadvantages and limitations of this method.