Vector Primitive Operations on GPUs using CUDA

Sean LEE
seanl@cse.unsw.edu.au
Programming Languages and Systems Group
The School of Computer Science and Engineering
The University of New South Wales

Abstract

The evolvement of *graphical processing units* (GPUs) made it possible to have data parallel processing on GPUs at low cost. Shading languages such as *Cg*, *GLSL*, and *HLSL* were used to manipulate graphics pipelines for *general purpose computing on GPUs* (GPGPU). However, programming general purpose computations using shading languages exposed a number of limitations: the texture-based memory management, the absence of scatter-write operation, etc.

The release of *Compute Unified Device Architecture* (CUDA) has lifted restrictions even further, and has made the graphics architecture more programmable and reconfigurable for GPGPU. It is specifically designed for GPGPU and exists as a form of an extension to C programming language. The main advantages that CUDA provides over other shading languages are threefold: (1) The *scatter-write* operation supported by CUDA broadens the range of algorithms that can be mapped on GPUs; (2) CUDA comes with the standlone compiler that compiles CUDA codes to standard C object files, which can be linked with other C object files using the standard C compilers; and (3) the memory allocation on GPUs and data transfer between CPUs and GPUs is performed in a more generic and efficient way.

In order to take these advantages, *ndpgpu*, the NDP library written in *Cg* for GPUs and introduced in the last SAPLING, has been ported to CUDA recently. In the talk, (1) how programming in other shading languages, especially Cg, and programming in CUDA differ, (2) what benefits CUDA has brought to the research, (3) the limitations of CUDA, and (4) how CUDA affected my research focus will be dicussed.