Universität Augsburg
University

Vertiefte Multicore-Programmierung

Universität Augsburg
Institut für Informatik
Prof. Dr. Theo Ungerer
Dr. Sebastian Weis

1. Juni 2016

Aufgabenblatt 7

1. Aufgabe (2 + 3 + 5 Punkte)

Implementieren Sie drei Varianten einer parallelen Matrixmultiplikation $AB = C$. Die Matrizen A und B sollen dabei quadratisch sein und mit zufälligen `float` Werten befüllt sein. Die Multiplikation soll auf Teilblöcken der Ergebnismatrix C erfolgen, wobei die Blockgrößen variabel für die jeweilige Teilaufgabe angepasst und optimiert werden sollen.

a) Erstellen Sie zunächst eine mit C++11-Threads parallelisierte Version und messen Sie die Laufzeiten für verschiedene Matrizen unterschiedlicher Dimensionen.

b) Implementieren Sie eine einfache Basisversion in CUDA. Verzichten Sie in dieser Teilaufgabe auf die Verwendung von geteiltem Speicher (Shared Memory). Teilen Sie die Arbeit in mehrere CUDA-Blöcke auf und verwenden Sie *Loop unrolling* (`pragma unroll`).

c) Implementieren Sie eine Matrixmultiplikation in CUDA mit geteiltem Speicher (Shared Memory). Achten Sie darauf, dass Ihre Zugriffe auf den globalen Speicher verschmolzen (coalesced)[1] werden können und vermeiden Sie Bank-Konflikte beim Zugriff auf den gemeinsamen Speicher.

Untersuchen Sie ihre beiden CUDA-Implementierungen mit dem Nvidia Visual Profiler, identifizieren Sie mögliche Flaschenhälse und versuchen Sie ihre Lösungen zu optimieren. (siehe *CUDA C Best Practices Guide*).

Hinweise

- Die Abnahme für das Blatt soll bis Dienstag, 7. Juni 2016 erfolgen.

[1] `https://cvw.cac.cornell.edu/gpu/coalesced?AspxAutoDetectCookieSupport=1`

Linksammlung

- http://hpc.oit.uci.edu/nvidia-doc/sdk-cuda-doc/
- http://docs.nvidia.com/cuda/cuda-c-best-practices-guide/
- http://docs.nvidia.com/cuda/kepler-tuning-guide/index.html