論　文　の　要　旨

題　目　　Memory Access Optimal Algorithms for the GPU

(GPU 用のメモリアクセスの最適なアルゴリズム)

氏　名　　笠　置　　明　彦

A GPU (Graphics Processing Unit) is a specialized circuit designed to accelerate computation for building and manipulating images. Latest GPUs are designed for general purpose computing and can perform computation in applications traditionally handled by the CPU. Hence, GPUs have recently attracted the attention of many application developers. The GPU has two types of memories: the shared memory and the global memory. The performance of applications using a GPU depends greatly on the usage of these memories. Because of the above background, we consider memory access optimal algorithms on a single GPU. This dissertation shows theoretical memory machine models and memory access optimal algorithms which are as follows:

We introduce simple parallel memory machine models that capture the essential features of NVIDIA GPUs. The Discrete Memory Machine (DMM) and the Unified Memory Machine (UMM) reflect the essential features of the shared memory and the global memory of NVIDIA GPUs. In both architectures, a sea of threads are connected to the memory banks (MBs) through the memory management unit (MMU). Each thread is a Random Access Machine (RAM), which can execute fundamental operations in a time unit. Threads are executed in SIMD fashion, and the processors run on the same program and work on the different data. The Hierarchical Memory Machine (HMM) is a hybrid of the DMM and the UMM. The HMM is a more practical parallel computing model that reflects the hierarchical architecture of CUDA-enabled GPUs.

Offline permutation is a task to move data along a permutation P given beforehand. The conventional algorithm of offline permutation performs $b[P(i)] \leftarrow a[i]$ for all i $(0 \leq i \leq n-1)$. We present conflict-free offline permutation algorithm on the DMM and implement it to run on the shared memory in the GPU. Our idea is to use two permutations S and D which can be obtained

from P. Using these two permutations our conflict-free permutation algorithm performs $b[D(i)] \leftarrow a[S(i)]$ for all i. In the experimental results, our conflict-free permutation algorithm runs in 167ns for any permutation including the random permutation and the worst permutation. We also present optimal offline permutation algorithm on the HMM. This algorithm has no stride access requests and no bank-conflicts. We evaluate these algorithms using several parameters and implement these algorithms to the GPU. The experimental results show that our optimal offline permutation algorithm on the HMM is faster than the conventional permutation algorithm for most cases.

The summed area table (SAT) of a matrix is a data structure frequently used in the area of computer vision which can be obtained by computing the column-wise prefix-sums and then the row-wise prefix-sums. The previously published best algorithm (2R1W SAT algorithm) performs 2 read operations and 1 write operation per element. We present a more efficient algorithm (1R1W SAT algorithm) which performs 1 read operation and 1 write operation per element. Clearly, since every element in a matrix must be read at least once, and all resulting values must be written, our 1R1W SAT algorithm is optimal in terms of the global memory access. We also show a combined algorithm ((1+r)R1W SAT algorithm) of 2R1W and 1R1W SAT algorithms that may have better performance. The experimental results show that our (1+r)R1W SAT algorithm runs faster than any other SAT algorithms for large input matrices. Also, it runs more than 100 times faster than the best SAT algorithm using a single CPU.