文本, 信件

描述已自动生成文本

描述已自动生成

Global and constant memory can get from the host. Shared mem is in each block, higher speed. Divide global mem into tile and load into chip. Barrier synchron: at first and after calculation ensures elements are used in each thread.

Process:Each thread load one element in each matrix and calculate inner product in each iteration Boundary 2d matrix index exceeds width (notify multiplication’s row and column) true load element else load 0.

Embarrassingly Parallel Computation: Be done in completely independent parts can be done in each thread. Operation: x’ = xcos + ysin, y’ = -xsin + ycos;

Monte Carlo Method: large dataset and draw identical samples and independent. Advantage: Error reduced to 1/sqrtn. Work with parallelism. Find estimation faster.

Random Number Generation: impossible to generate actual random one on deterministic machine. Usually generate x and x\_i+1 = (axi+c) mod m

cuRAND: Timing: clock\_t start & stop, in device returns the value of a per-multiprocessor counter that is incremented every clock cycle.

Parallel algorithms time depends on input size, communication, processors, architecture.

Speedup Factor: p = number of pro S(p) = time using one / time using multiple. 图示

描述已自动生成

Time component: inherently sequential cop, parallel comp, communication.

Efficiency: S(p)/ p

Amdahl’s law: just ignores commu time to calculate maximum speed up and set f = portion of sequential time S(p) = p/1+(p-1)f

Gustafson-Barsis’s law: predict scaled speedup. Set s = portion of parallel. S(p) = p+s(1-p)

Warps: 32-threads, scheduling units, all threads must execute same instruction at some time become efficiently when all follow same control flow path.

Control divergence: thread in warp take different control flow, threads with same path will run in parallel

Histogramming: Interleaving Partition of Input:coalescing and memory accessing. Race condition: order of which threads accessing is undefined. Thread Coordination: \_\_syncthreads & atomics operations: modifying a value back to memory without the interference of any other threads, can be used in shared and global (expensive)

Hierarchical Atomics: per-thread atomic to shared sum, per-block add to total.

N-body problem: predicting motion of N objects.

Parallel verlet neighbor list algorithm: making “neighbor list”, only compute smaller layer radius and larger cutoff radius. in GPU, using CUDPP (sorting and parallel).

Barnes-Hut algo: recursively dividing bodies storing into quad tree. Create square for whole graph, divided into four sub squares. If no body sub, don’t consider, if there is one return. Or keep doing sub. Cost: O(nlogn). Parallel algo: thinking cluster of bodies which one is very large like star.

Dynamic parallelism: kernel can do kernel call and also support the recursion one.