A GPU-Friendly Skiplist Algorithm

Nurit Moscovici, Erez Petrank
Computer Science Department
Technion – Israel Institute of Technology

Nachshon Cohen
École Polytechnique
Fédérale de Lausanne

Very few dynamically-updated data structures adapted to GPU

Divergence
Threads on the GPU are divided into groups called warps, which proceed through shared code in lockstep.

if tid%2==0
  do something
else
  do something

Pitfalls
Access to global memory is most efficient when it can be coalesced into as few memory transactions as possible.

Memory Coalescing

Our Solution: GFSL
A Fine-Grained Lock-Based Skiplist

Lock-Free Traversal
1. Team reads chunk
2. Each thread decides whether it is a possible next step
3. Team shares results and traverses pointer from “winning” thread

Lock-Based Insert/Delete
1. Find and lock bottom-level chunk.
2. Traverse levels to insert/delete while locking & releasing
   (Insert: Bottom up, Delete: Top down)
3. Unlock bottom chunk

Handling Full/Empty Chunks

Evaluation
Implemented in CUDA C on GeForce GTX 970 Nvidia GPU

Compared to Misra and Chaudhuri's GPU skiplist implementation [2012] (M&C)

M&C's implementation shows performance meltdown as structure grows
• Crashes on OOM for large structures

Speedup over existing implementation for different operation mixtures and key ranges

Throughput given different key ranges for an operation mixture of [{i, d, c}={5, 5, 90]