CS 798-003 Paper Summary
A Dynamic Hash Table for the GPU

Jun-Qing Lim

jq3lim@uwaterloo.ca

1 Brief Overview

The paper proposes a new GPU hash table (called slab table) that enables concurrent updates with high performance, unlike traditional GPU data structures that address updates by rebuilding the entire data structure from scratch. It uses a new linked list (called slab list) that operates on warp-cooperative work sharing strategy and modeled closely to GPU's hardware characteristics. This slab list supports asynchronous, concurrent updates and search queries that minimizes memory divergence with an efficient use of space.

The slab hash has also adopted a newly designed warp-synchronous dynamic memory allocator (called SlabAlloc) that scales up to 1 TB of data structures (larger than current GPU memory size) without any CPU intervention. The performance of this slab list and slab hash are then evaluated with various benchmarkings and found significant speedup compared to previous semi-dynamic hash tables.

2 Positive Points

- Well-written with good clarity on the design of the strategy, its implementation details and rationale behind the decisions.
- Highly scalable and dynamic memory allocator without any CPU intervention.
- Many novel proposals (slab list, slab table, slab memory allocator) compacted in one paper, laiding a good ground work for future expansion in dynamic data structures with specialized analytics.
- Interesting benchmarking approaches of using both static and dynamic methods, giving good coverage of the evaluation.

3 Negative Points

- The performance was evaluated on specific GPU architecture (NVIDIA Tesla K40c) and it is unclear if it will have similar results on GPU models and as such the performance might require further validation/evaluation.
- Lacks discussion on the potential limitations of the proposed data structures (i.e., any memory usage limitation, overhead, etc?)

4 Insightful Discussions

- How does the memory allocator scales beyond the 1 TB limit in environment like a distributed environment?
- Can this memory allocator made separate and generalized for other data structures in adopting similar strategy?