Our approach

FKS and cuckoo hashing are problematic for GPU:
- FKS hashing is fast, but requires too much space
- Cuckoo hashing can be space-efficient, but requires slow global memory and global synchronization for parallel insertion

Get efficient algorithm by combining both:
- Like with FKS hashing, first partition into small buckets of at most 512 items
- Build parallel cuckoo hash on smaller buckets in fast on-chip memory
- Retrieval takes at most 3 probes: one for each cuckoo sub-table of the bucket item hashes into

Algorithm can be generalized to handle multiple values per key, or to generate two-way index between keys and unique IDs.

We parallelized cuckoo hashing insertion for the GPU. Parallel cuckoo hashing:
- Inserts all items simultaneously, iterating through sub-tables in round-robin fashion
- Assumes that exactly one write will succeed for colliding items
- Typically completes in $O(g(n))$ iterations for two sub-tables

Results

Timings for increasingly finer voxelizations of Lucy. We compare against a sorted list, using binary searches for retrieval. Construction takes roughly the same amount of time as the radix sort, while our retrievals are consistently faster than binary searches for random access.

References