Tip of the Iceberg: Low Associativity Paging
Nirjhar Mukherjee

The TLB Coverage Problem

- Traditional fully-associative page mapping is slow.
- TLBs are designed to solve this issue by caching mappings.
- Unfortunately, TLB coverage is often poor, resulting in frequent misses.
- Increasing coverage may help → but TLB size grows slowly.

Ways to reduce associativity

Hashing is the preferred mechanism to reduce associativity.
Most commonly used hashes include left[$d$] and single.
However they both have flaws:
- Left[$d$]'s bound don’t hold under dynamic load
- Single has high variance, resulting in fragmentation

Implementing and tuning in the kernel

One caveat of hash based low associativity page mapping schemes is the existence of associativity conflicts: What happens when a virtual page is hashed to a set of L pages, which are all already mapped?

We tune iceberg with various sizes of L to find a good fit – Minimize L (or rather b as in bucket (of pages) size), while also minimizing associativity conflicts.

A solution

Iceberg[$d$] Hashing uses a mix of the two:
- It uses single for most mappings
- And uses Left[$d$] for a small number of mappings
This can increase TLB coverage by up to 8x