05 – Scalable Garbage Collection for In-Memory MVCC Systems

1. Overview of Idea
   Garbage collection can be a bottleneck of in-memory MVCC systems. But, many GC are too coarse-grained so that HTAP can occur vicious cycle. Doing GC when updates occurs can be a good solution.

2. Main Finding
   If we do GC more frequently, the time for GC can be lower, maintaining version chain short.

3. Systems used and its Specifications
   HyPer is used and added custom GC method 'STEAM'

   A. Basic Idea

      i. Maintain two linked list, *activetxn, commitedtxn*

      ii. If txn commits, version created by this txn added to version chain

      iii. When appending version chain,
      check versions which have *commitId* < *startTs,* which means commited timestamp and oldest timestamp in *activetxn*

   B. Eagerly Pruning of Obsolete Versions

      i. When touching version chain, merge versions to reduce version chains' length
      which can reduce intermediate versions that is not visible.

4. Workloads evaluated
   TPC-C, TPC-H, CH benchmarks are used.