Compression For LSM Trees Outline

* Introduction
  + Goal of Databases/Theory: Minimize latency for basic operations such as read, writes
    - Tradeoffs: Cannot optimize for reads, writes, space simultaneously (RUM Conjecture)
    - Compression: it minimizes I/O (space) while increasing CPU utilization
    - Latency = CPU Time + I/O Time
    - Research Objectives:
      1. Can we use compression to minimize overall latency?
      2. Can we design an algorithm/approach that can design the most optimal compression scheme for any given LSM Tree (Data Sketches/Learned Model)?
      3. What compression techniques can be employed for LSM Trees to minimize CPU while maximizing I/O (Block Compression)
* Motivation / Background
  + Rocks DB Paper (Foundational Work)
    - Open Ended Question: is it optimal to use the same level multiplier with regards to space amplification while using compression?
    - Main Ideas:
      1. Reduce storage in Rocks DB using compression while not significantly impacting reads or writes
      2. Use of SSDs makes storage a main bottleneck (not enough space)
      3. Compression Techniques Used:
         * Key Prefix Encoding
         * Sequence ID garbage collection
         * Data Compression (Snappy, LZ)
         * Dictionary Based Compression
      4. Strong compression at lower level, weak or no compression in upper levels
      5. Evaluation:
         * Reduce Storage by 50% over InnoDB
         * Increasing transaction throughput
         * Decreasing write amplification
         * Increasing read latency marginally
  + RUM Conjecture
    - Tradeoff between:
      1. Read
      2. Write
      3. Space (Storage or Memory)
    - Optimize two at the expense of the third
    - Can optimize to some point for all three factors
    - Access method that is optimal with respect to one of the read, update, and memory overhead cannot reach the optimal value for both remaining overheads
    - Sets upper bound for two out of the three:
      1. Update, Read, Space
    - Claim: Compression does NOT effect the overall RUM conjecture model (because of CPU costs increasing and increasing latency)
  + LSM Tree Paper
    - Explain basic architecture about LSM Trees
    - Level based Structure
    - Buffer flush (sorted)
    - Basic properties of LSM trees
* Methodology / Experimentation
  + Mathematical Formulation
  + Basic Learning of space based on compression schemes
  + Experiments for read and write performance based on data scheme (Snappy)
* Future Plans
  + Block Compression (applying block compression to LSM Trees)
  + Data Sketching to Optimize Compression Schemes

**TODO:**

* Design basic learning experiment before Thursday for Snappy vs RLE compression for the database and get the results for it
* Read the wacky continuum paper for data sketching and on-the-fly statistics
* Read Ravi Uncle’s paper more carefully and fully understand the tradeoffs that are taking place there