ENDURE: A Robust Tuning Paradigm for LSM Trees Under Workload Uncertainty

Andy Huynh, Harshal A. Chaudhari, Evimaria Terzi, Manos Athanassoulis
Age of Log-Structured Merge-Trees

High impact tuning knobs

Compaision  Buffer size  Size ratio

Dictates performance!

How do we go about tuning these knobs?
LSM Trees

Buffer

Level 1

Level 2

... Level L

Buffer fills → Sort and Flush to disk

Size Ratio (T)

$\pi$: Compaction Policy
$T$: Size Ratio
$m_{\text{filter}}$: Filter memory
$m_{\text{buff}}$: Buffer memory

How do we define a workload?
Query Types

Workload: \((z_0, z_1, q, w)\)

Empty Reads: \(z_0\)

Non-Empty Reads: \(z_1\)

Range Reads: \(q\)

Writes: \(w\)

Cool! How do we go about tuning?
The LSM-Tuning Problem

\( w : \) Workload \((z_0, z_1, q, w)\)

\( \Phi : \) LSM Tree Design \((m_{buff}, m_{filter}, T, \pi)\)

\( C : \) Cost

\[ \Phi^* = \arg\min_{\Phi} C(w, \Phi) \]
Point Reads

Empty Reads : $z_0$

Non-Empty Reads : $z_1$

Sum of false positives

$Z_0(\Phi) = \sum_{i=1}^{L} f_i$

Probability query is satisfied at level $i$

$Z_1(\Phi) = \sum_{i=1}^{L} \frac{T^{i-1} \cdot (T - 1)}{N_f(T)} \cdot \frac{m_{buf}}{E} \left( 1 + \sum_{j=1}^{i-1} f_j \right)$

False positives from levels above

Range-Reads and Writes

\[ Q(\Phi) = S_{RQ} \cdot \frac{N}{B} + L \]  
1 I/O per Seek per level

Average number of merges a write will participate in

\[ W(\Phi) = \frac{L}{B} \cdot \frac{T - 1}{2} \cdot (1 + A_{rw}) \]

Writes only flush once buffer is full

---

The LSM-Tuning Problem

\( \mathbf{w} : \) Workload \( (z_0, z_1, q, w) \)

\( \Phi : \) LSM Tree Design \( (m_{buff}, m_{filter}, T, \pi) \)

\( C : \) Cost \( (I/O) \)

\[ \Phi^* = \text{argmin}_{\Phi} C(\mathbf{w}, \Phi) \]

Define our cost function

\[ C'(\hat{\mathbf{w}}, \Phi) = \hat{\mathbf{w}}^\top \mathbf{c}(\Phi) = z_0 \cdot Z_0(\Phi) + z_1 \cdot Z_1(\Phi) + q \cdot Q(\Phi) + w \cdot W(\Phi) \]
Tuning Problems

\[ w_0 : \text{Workload } (z_0, z_1, q, w) \]

- Optimal configuration for the workload

Optimal tuning depends on workload

Workload uncertainty leads to **sub-optimal** tuning
Outline

Introduction

LSM Trees Notation

Nominally Tuning LSM Trees

**ENDURE: Robustly Tuning LSM Trees**

The ENDURE Pipeline

ENDURE Evaluation
The LSM-Tuning Problem

\( \mathbf{w} : \) Workload \((z_0, z_1, q, w)\)

\( \Phi : \) LSM Tree Design \((m_{buff}, m_{filter}, T, \pi)\)

\( C : \) Cost (I/O)

\[ \Phi^* = \arg\min_\Phi C(\mathbf{w}, \Phi) \]

\( U_\mathbf{w}^\rho : \) Uncertainty Neighborhood of Workloads

\( \rho : \) Size of this neighborhood

\[ \Phi^* = \arg\min_\Phi C(\mathbf{\hat{w}}, \Phi) \]

s. t., \( \mathbf{\hat{w}} \in U_\mathbf{w}^\rho \)
Robust Tuning

\( w_0 : \text{Workload} \ (z_0, z_1, q, w) \)

\[ \Phi^* = \arg\min_{\Phi} C(\hat{w}, \Phi) \]

s.t., \( \hat{w} \in U^\rho_w \)

Optimal configuration for the workload

Robust configuration for the workload neighborhood

\[ \Phi^* = \arg\min_{\Phi} C(\hat{w}, \Phi) \]

s.t., \( \hat{w} \in U^\rho_w \)

Optimal configuration for the workload

Robust configuration for the workload neighborhood
Uncertainty Neighborhood

Neighborhood of workloads ($\rho$) via the KL-divergence

$$I_{KL}(\hat{w}, w) = \sum_{i=1}^{m} \hat{w}_i \cdot \log\left(\frac{\hat{w}_i}{w_i}\right)$$

$U_{W}^{\rho}$: Uncertainty Neighborhood of Workloads

$\rho$: Size of this neighborhood
Calculating Neighborhood Size

**Workload Characteristic**

**Historical workloads**
maximum/average uncertainty among workload pairings

**User provided workload uncertainty**

\[ U^\rho_W: \text{Uncertainty Neighborhood of Workloads} \]

\[ \rho : \text{Size of this neighborhood} \]
Solving Robust Problem

Iterating over every possible workload is expensive

\[
\Phi_R = \arg\min_{\Phi} \mathbf{w}^T \mathbf{c}(\Phi)
\]

s.t. \( \hat{\mathbf{w}} \in \mathcal{U}_w^\rho \)
Solving Robust Problem

Iterating over every possible workload is expensive

Rewrite as a min-max

Find the dual of the maximization problem to reduce to a feasible problem [2]

\[
\Phi_R = \arg \min_{\Phi} \ \hat{w}^T c(\Phi)
\]
\[
\text{s.t. } \hat{w} \in \mathcal{U}_w^\rho
\]

\[
\min \ \max_{\Phi \ \hat{w} \in \mathcal{U}_w^\rho} \hat{w}^T c(\Phi)
\]

\[
\min_{\Phi, \lambda \geq 0, \eta} \left\{ \eta + \rho \lambda + \lambda \sum_{i=1}^{m} w_i \phi_{KL}^* \left( \frac{c_i(\Phi) - \eta}{\lambda} \right) \right\}
\]

ENDURE Pipeline

Workload Characteristic

System Information
Page Size
Memory Budget

Expected performance

ENDURE Solves the Robust Problem

RocksDB Configuration
Testing Suite

ENDURE in Python, implemented in tandem with RocksDB

Uncertainty benchmark

• 15 expected workloads
• 10K randomly sampled workloads as a test-set

Normalized delta throughput

\[ \Delta_w(\Phi_1, \Phi_2) = \frac{1/C(w,\Phi_2) - 1/C(w,\Phi_1)}{1/C(w,\Phi_1)} \]

Nominal vs Robust: > 0 is better
1 means 2x speedup
Impact of Workload Type

Unbalanced workloads result in overfitted nominal tunings.
Impact of Workload Type

Unbalanced workloads result in overfitted nominal tunings
Tuning with uncertainty ($\rho > 0.5$) provides benefits
Relationship of Expected and Observed $\rho$

**Observed $\rho$:** distance from executed workload to expected workload

**Expected $\rho$:** workload given to tuner

**Highest throughput** when observed and expected $\rho$ match

**Lowest throughput** when $\rho$ is mismatched
Impact of Observed vs Expected $\rho$

- **Expected $\rho$:** given to tuner
- **Observed $\rho$:** distance from executed workload to expected workload
Impact of Observed vs Expected $\rho$

- Higher expected $\rho$ accounts for more uncertainty,
- Potential speed up of 4x
- Higher expected $\rho \rightarrow$ anticipates writes $\rightarrow$ shallow tree
\( \rho \) and Performance Gain Distribution

\( w_{11} = (33\%, 33\%, 33\%, 1\%) \)

**Expected \( \rho \):** given to tuner

**Throughput**
\( \rho \) and Performance Gain Distribution

\[ w_{11} = (33\%, 33\%, 33\%, 1\%) \]

Peak of the distribution moves towards higher throughput as we consider higher uncertainty
Workload Sequence on RocksDB

RocksDB instance setup with 10 million unique key-value pairs of size 1KB

Each observation period is 200K queries, with 5 observations per session 6 million queries to the DB

Writes are unique, range queries average 1-2 pages per level
Workload Sequence

**Model I/O**
- Nominal: h: 8.2, T: 8.4
- Robust: h: 1.0, T: 4.7
- Tiering
- Leveling

**System I/O**
- \( \rho : 2.31 \)
- \( I_{KL}(\hat{w}, w) : 2.31 \)

1. Reads (30%, 58%, 12%, 0%)
2. Range (7%, 10%, 84%, 0%)
3. Empty Reads (86%, 9%, 4%, 0%)
4. Non-Empty Reads (8%, 86%, 6%, 0%)
5. Reads (30%, 58%, 12%, 0%)
6. Reads (30%, 58%, 12%, 0%)
Workload Sequence

Small subset of results! Take a look at the paper for a more detailed analysis.
Thanks!

Workload uncertainty creates suboptimal tunings

ENDURE: robust tuning using neighborhood of workloads

Deployed ENDURE on RocksDB

disc.bu.edu/
www.ndhuynh.com/
@nd_huynh