Constructing and Analyzing the LSM Compaction Design Space

BOSTON UNIVERSITY

Subhadeep Sarkar Dimitris Staratzis Zichen Zhu Manos Athanassoulis

LSM Compactions

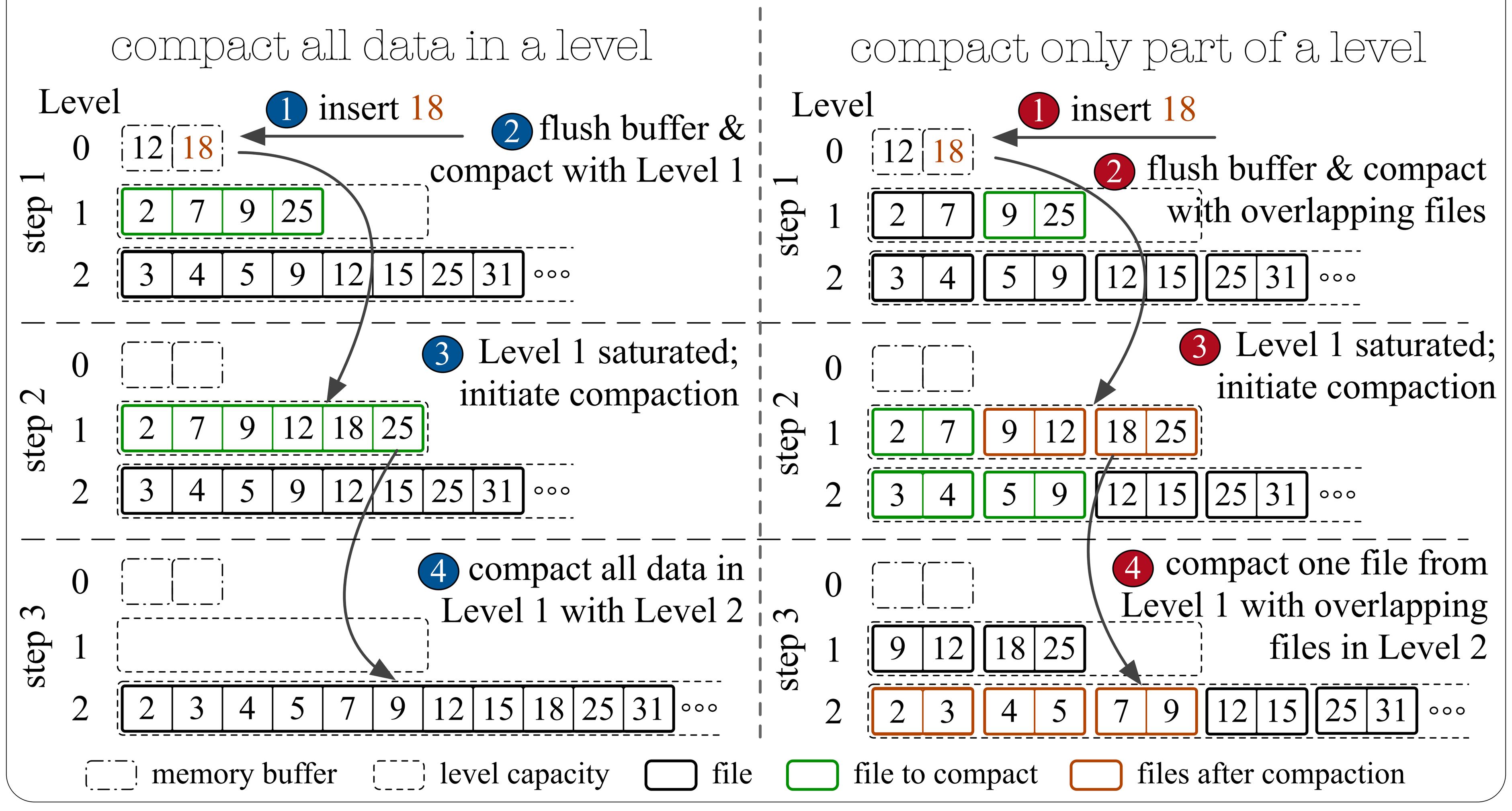
LSMs everywhere

RocksDB, WT, cassandra, accumulo, QuasarDB, influxdb, SQLite

Fast Ingestion
+
Good Reads
+
Great Space Utilization

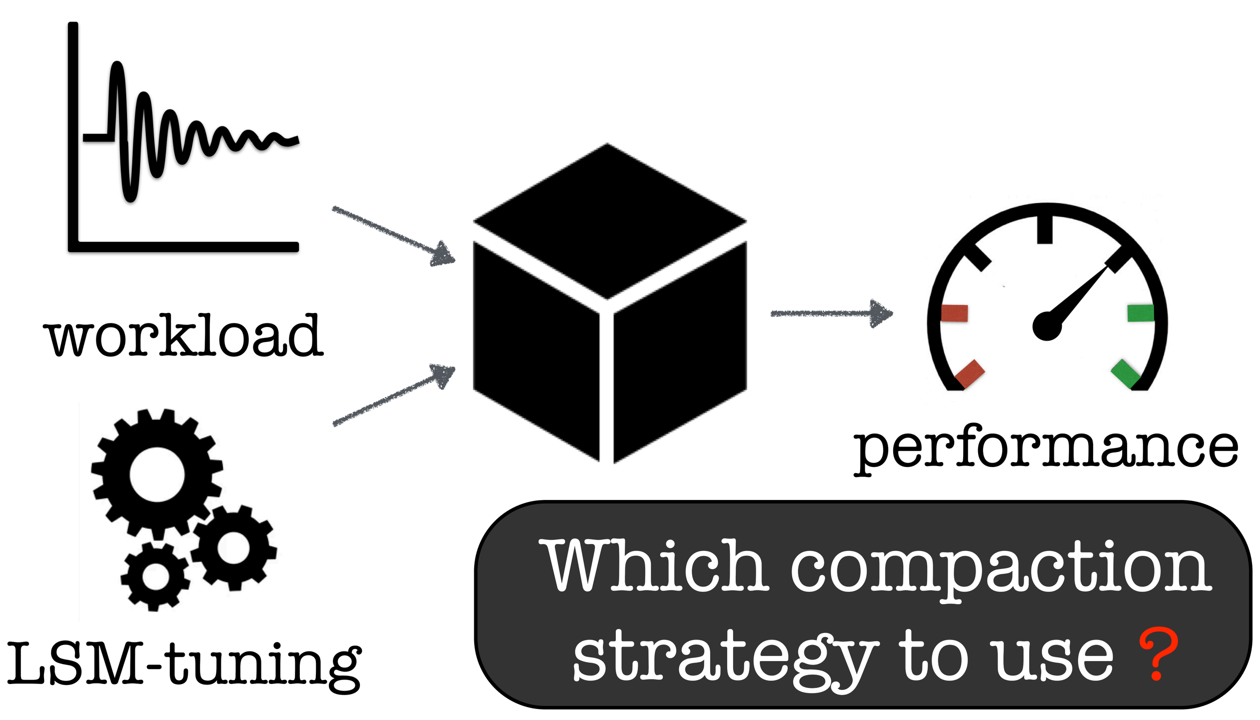
▶ **Periodic Compaction**

Full Compaction vs. Partial Compaction



Design Choice

The Compaction Black Box

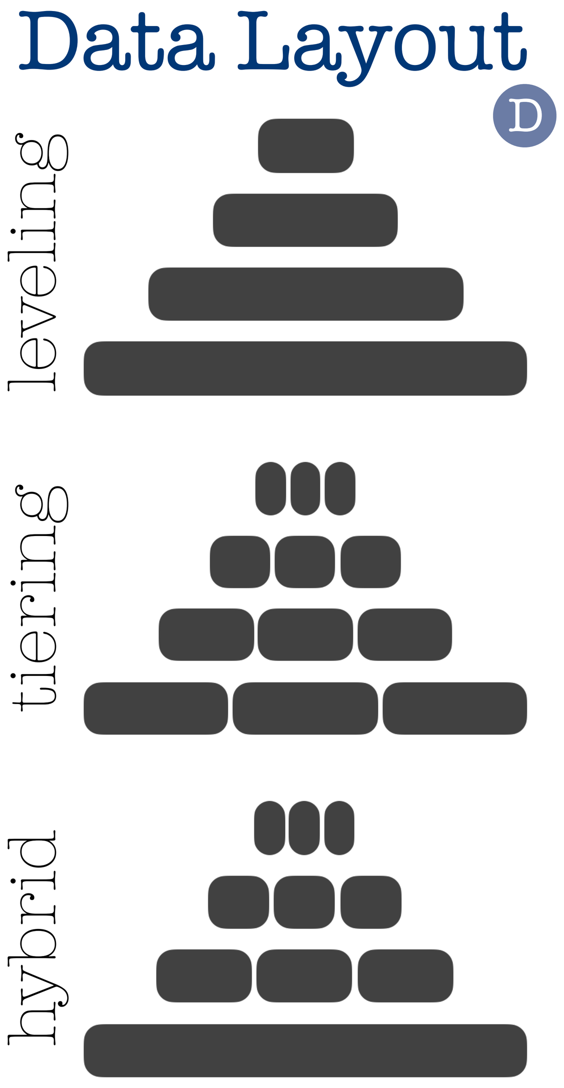


The Design Questions ▶ Design Primitives

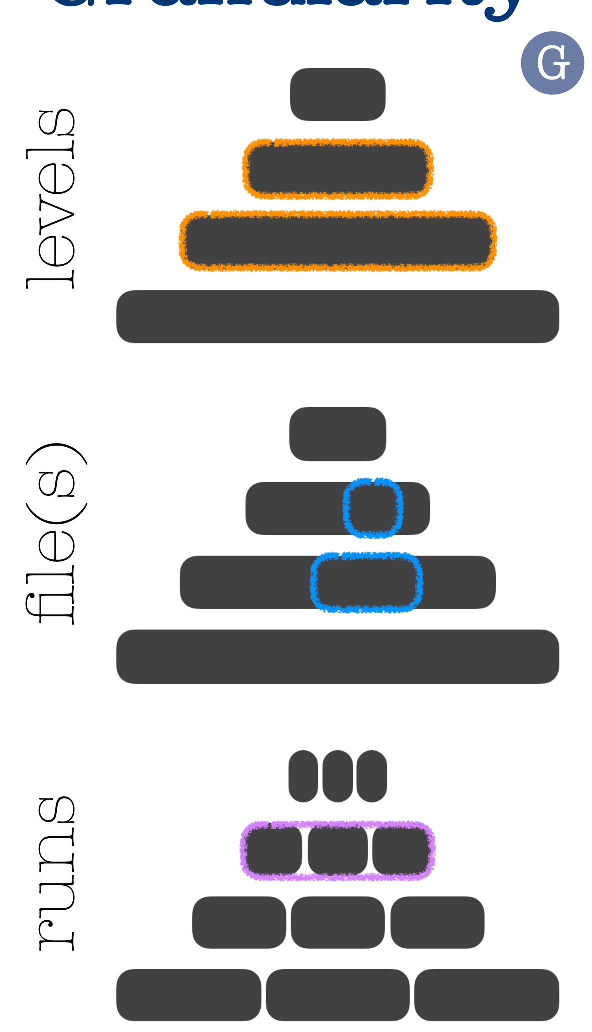
How to lay out the data?	**Data Layout**
How much data to move?	**Compaction Granularity**
Which data block to be moved?	**Data Movement Policy**
When to re-organize layout?	**Compaction Trigger**

The Design Space

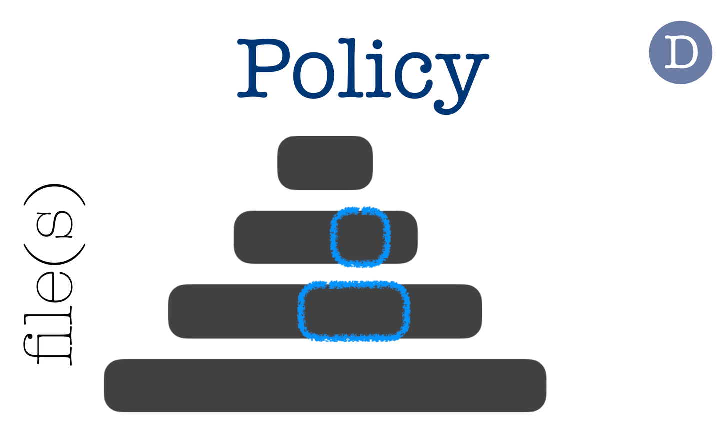
Data Layout



Granularity



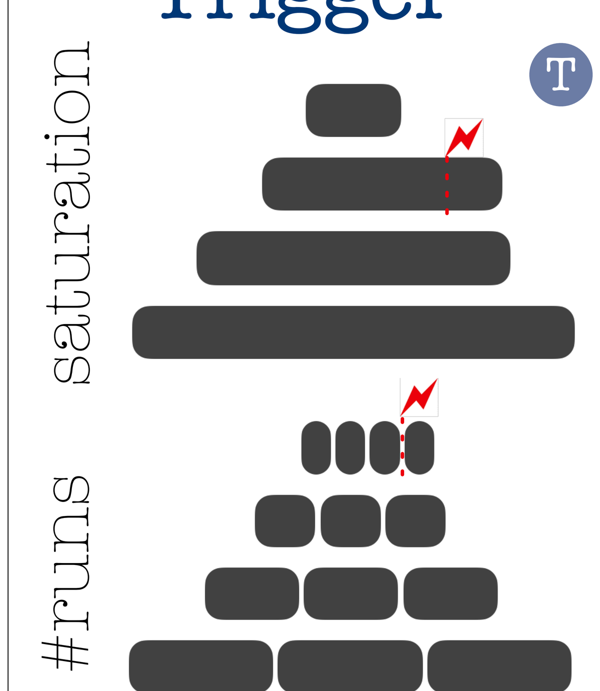
Data Movement Policy



Which file(s) to pick?

- Round-robin
- Least overlap
- Coldest
- Oldest
- Most tombstones

Trigger



- Space ampl.
- Age of a file
- Read ampl.

Primitive Ensemble

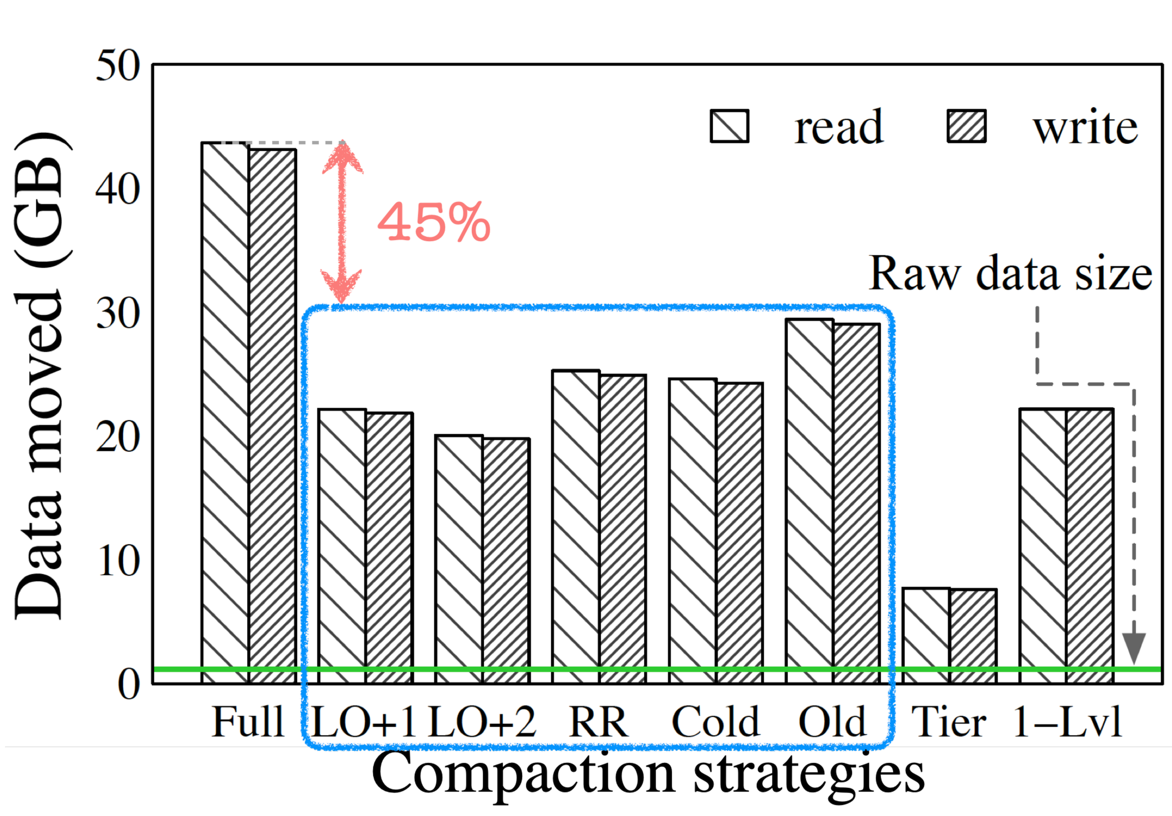
Full
- L leveling
- G levels
- D N/A
- T level saturation

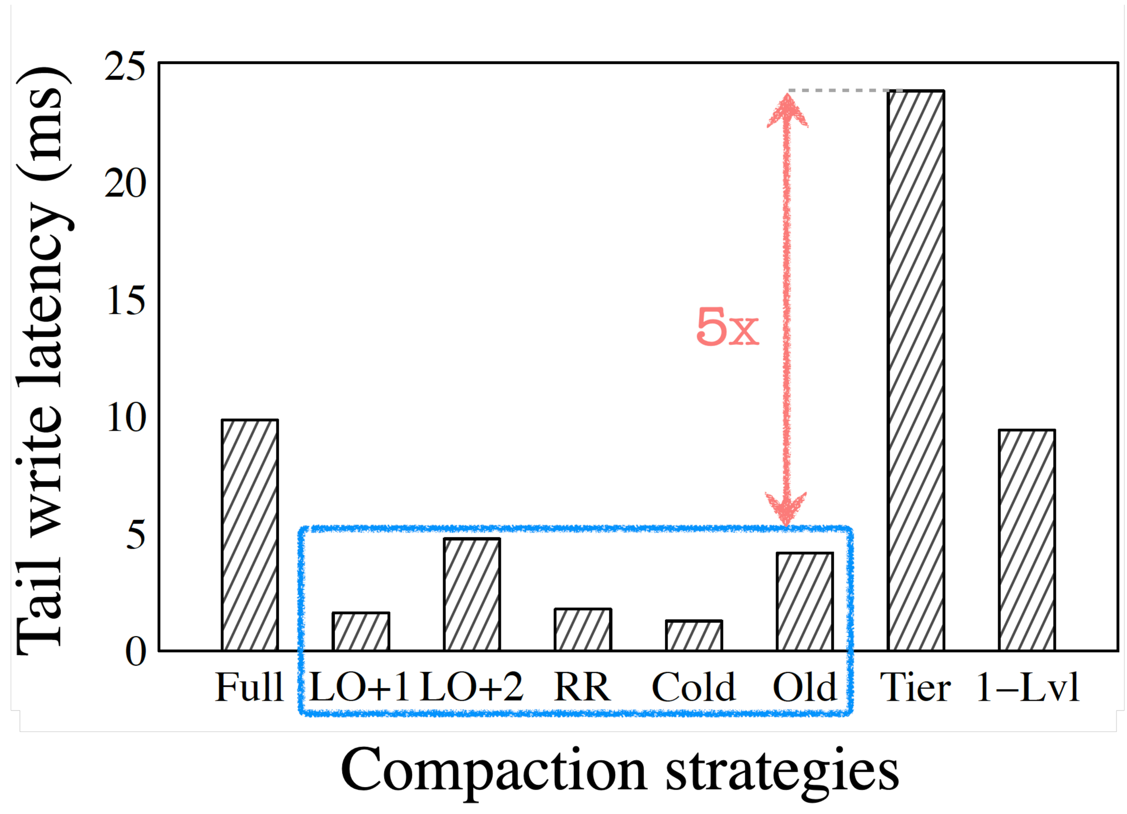
1-Lvl
- L hybrid leveling
- G file
- D least overlap w/ parent
- T level saturation

Tier
- L tiering
- G runs (in same level)
- D N/A
- T #runs / space ampl.

Observation

Experimental Results





Smaller compaction granularity reduces data movement

Tiered layout has extremely high tail write latency

Key Takeaways

- Know your LSM-compaction
- Avoid the worst designs
- Adapt choices w/ workloads
- Design new compactions

BOSTON UNIVERSITY
College of Arts & Sciences
Department of Computer Science
DiSC lab