FloDB: Unlocking Memory in Persistent Key-Value Stores

Oana Balmau Rachid Guerraoui Vasileios Trigonakis Igor Zablotchi

Introduction

Key-value (KV) stores - very widespread

Online shopping Messaging Advertising

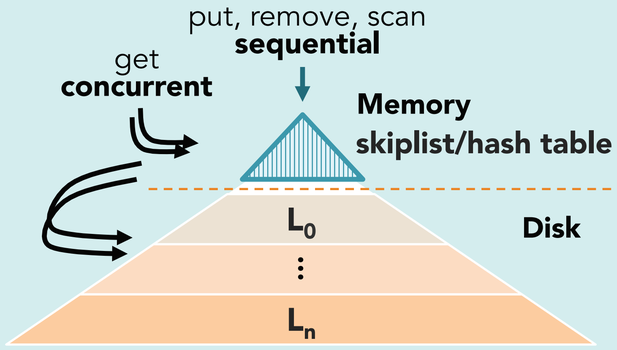
LEVELDB cassandra redis RocksDB mongoDB

LSMs = classic solution for persistent KV stores

LSM structure

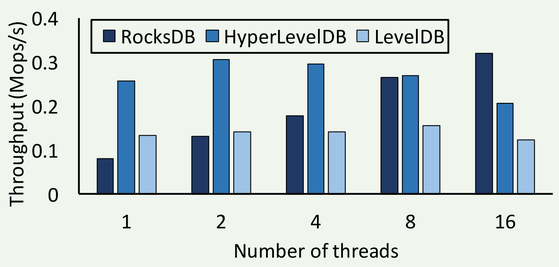
Two components:

- **Memory**
  fast; absorbs updates.
- **Disk**
  persists data in background.

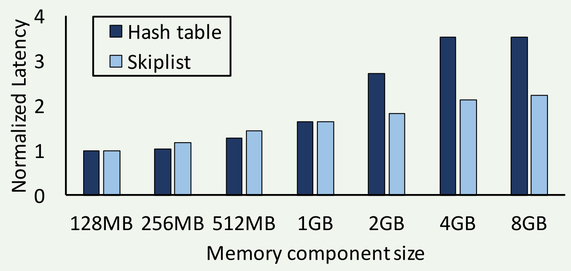


LSM memory components have inherent **scalability limitations.**

Limitation 1:
Scalability with threads



Limitation 2:
Scalability with memory size

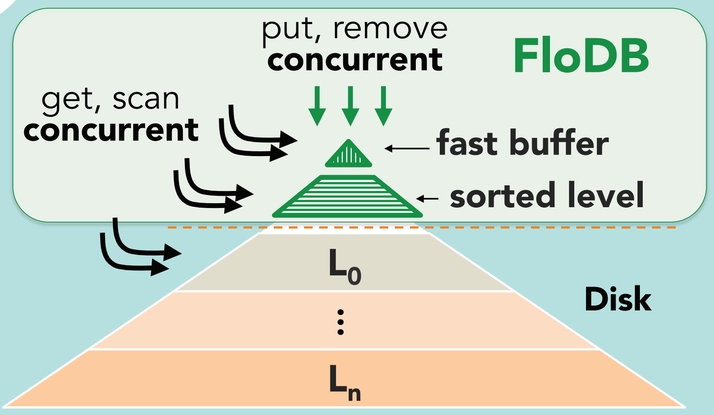


Objectives

Design a **memory component** for LSMs that:

- Scales with **memory size**.
- Uses **minimal synchronization** to scale up.

FloDB



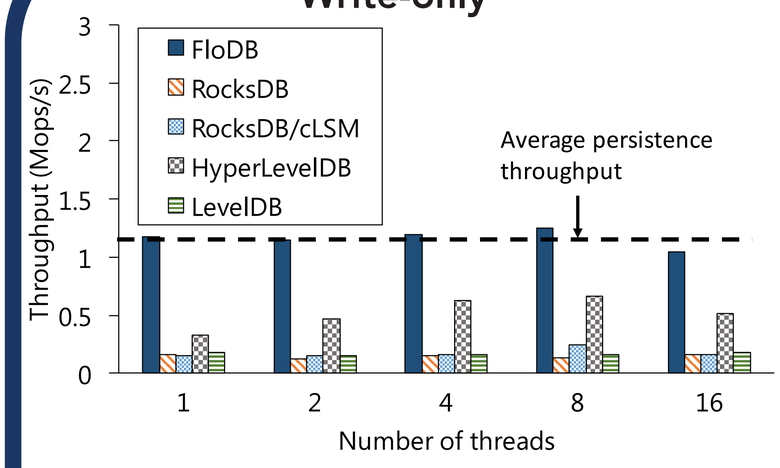
Main Ideas

- Two-level memory component
- Each level: a highly concurrent data structure

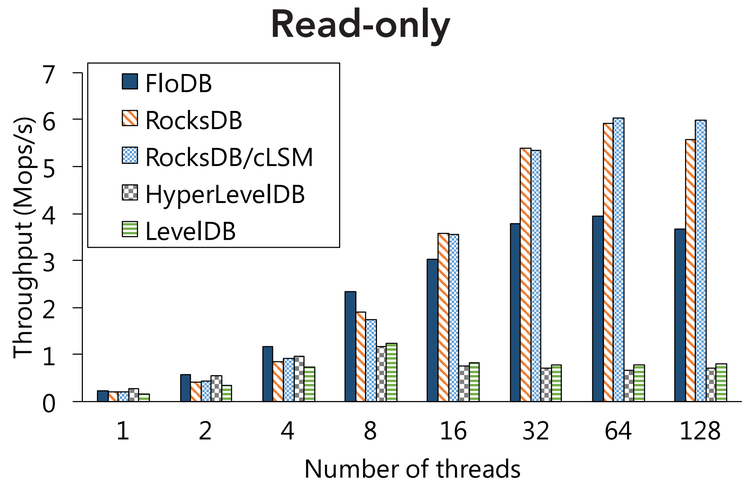
- Small, fast **top-level** + large, slower, **bottom-level.**
- Top-level ~ **concurrent hash table.**
- Bottom-level ~ **concurrent skiplist.**
- Reads/updates/scans proceed in parallel.
- Most **updates complete in fast top-level.**
- Can increase **memory size** w/o hurting performance.
- Data **flows**: top-level → bottom-level → disk.

Results

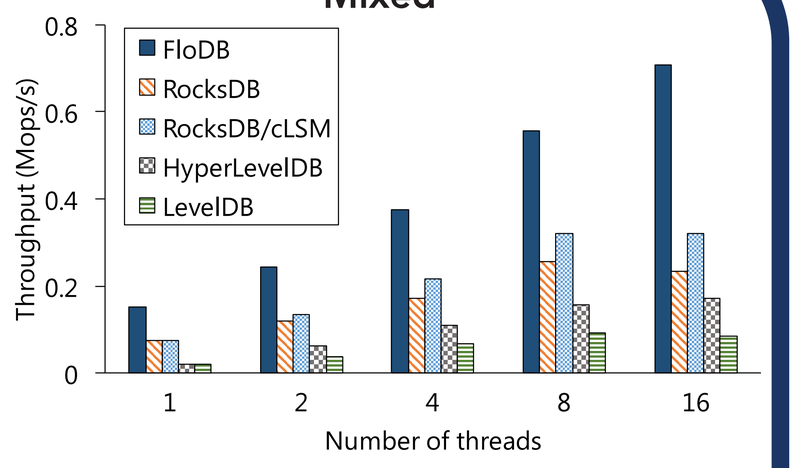
Write-only



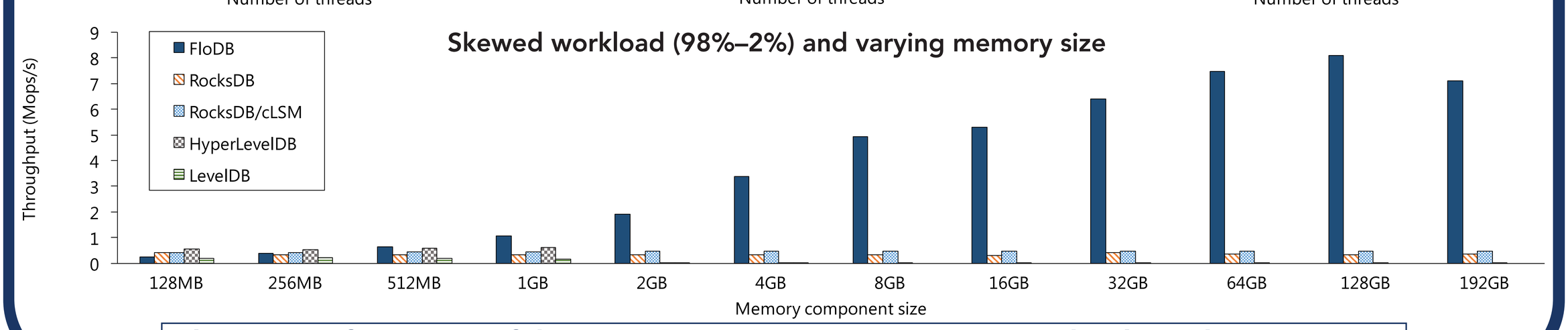
Read-only



Mixed



Skewed workload (98%–2%) and varying memory size



FloDB outperforms state-of-the-art LSMs in write-intensive scenarios and scales with memory size.

Find out more in our paper:

O. Balmau, R. Guerraoui, V. Trigonakis, and I. Zablotchi. FloDB: Unlocking Memory in Persistent Key-Value Stores. In Proceedings of the 12th European Conference on Computer Systems (EuroSys '17)

Check out our website: lpd.epfl.ch/site/flodb

This work has been supported in part by the European Research Council (ERC) Grant 339539 (AOC) and by the Swiss National Science Foundation (SNSF) Grant 166306.

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE