UC SANTA CRUZ | Baskin Engineering

LEAD (LEArned DHT), with Learned Hash Function

New field for further research on integrating learned models with distributed systems.

Range Query

KV Cache Management

Content Delivery Network Storages

LLM Serving & RAG

Edge-AI & IoT Data Lakes

A Distributed Learned Hash Table

Vector Databases

Blockchain Indexing

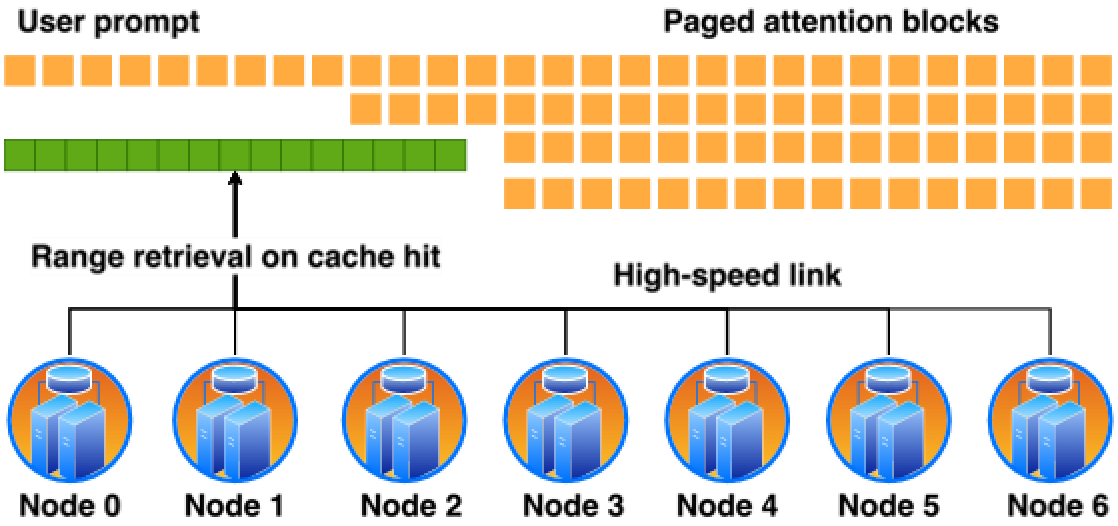
Shengze Wang[1], Yi Liu[1], Xiaoxue Zhang[2], Liting Hu[1], Chen Qian[1]

[1]University of California, Santa Cruz [2]University of Nevada, Reno

Shengze@ucsc.edu | Shengze.io

Fetching KV Cache you already computed.

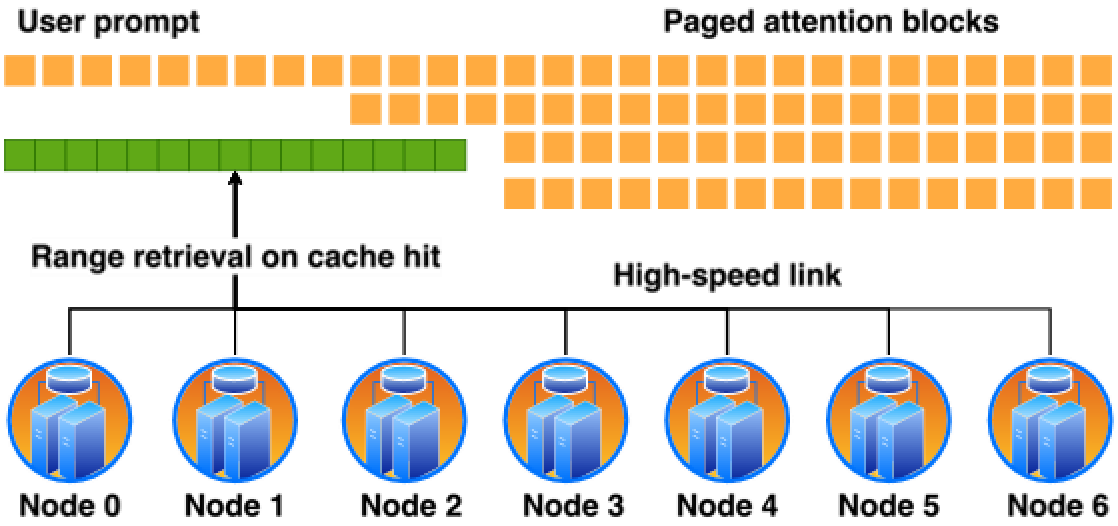
New Bottleneck in LLM serving



- Long contexts ⇒ massive KV; **Prefix sharing** across workers
- Need fast retrieval of **contiguous ranges**
- Avoid bottlenecks from **central KV router**

Fetching KV Cache you already computed.

New Bottleneck in LLM serving



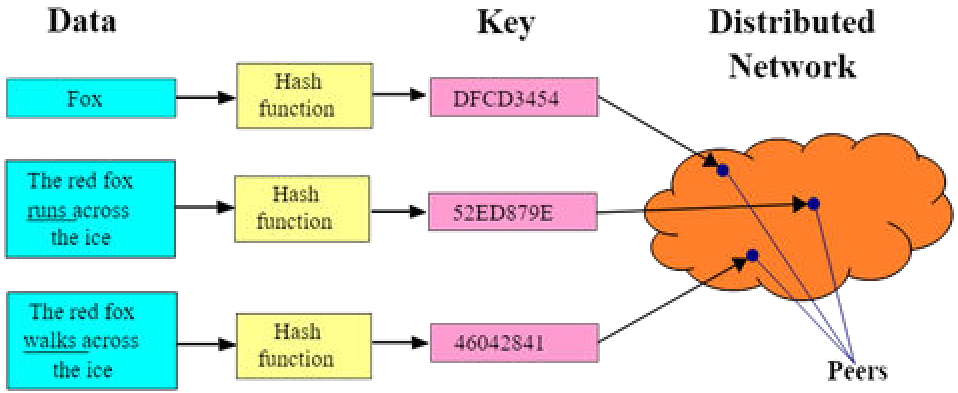
- **Fetch spans** (contiguous or semantic) , not only points
- **Few hops** per request; **Robust** to churn; Works at **scale**

Distributed Hash Table (DHT)

DHTs are pivotal in high-impact key-value applications

- **Consistent hashing** spreads key–value data across peers
- Map key → ID; place by ID; route via finger tables in **O(log N)**

Deployments: Storage (*IPFS*), DBs (*Cassandra*), Blockchains (*Ethereum*), Protocols (*BitTorrent*), CDNs, Edge/IoT, etc.



Classical DHTs fail for ranges.

Limitations of DHT for AI- and Data-driven workloads

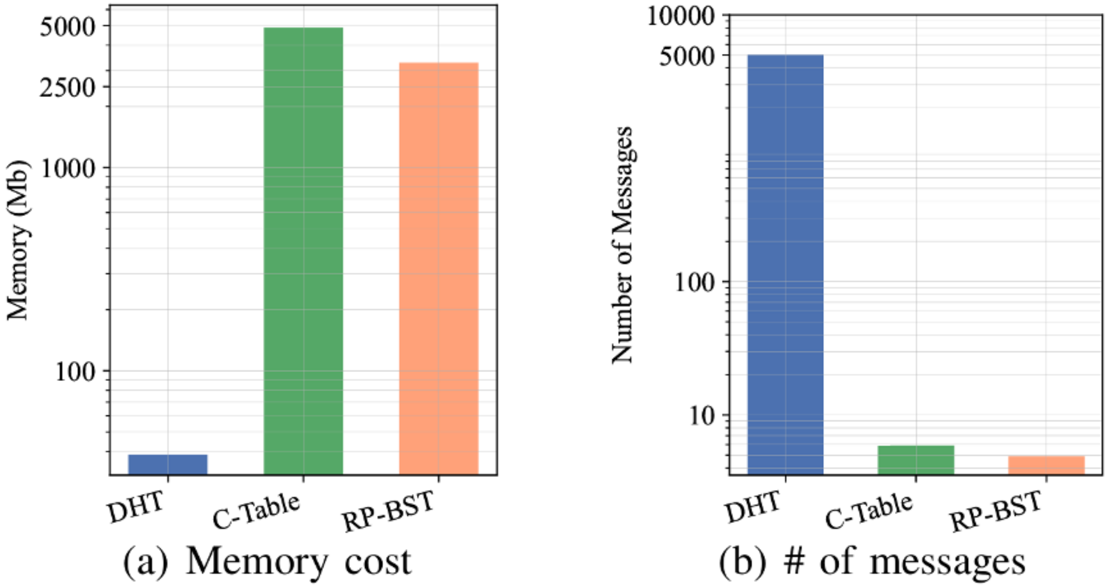


Fig. 1. Micro benchmark of Range Query Performance

- Hashing destroys **order**
- **Nearby** keys ⇒ **random** nodes
- **Range** ⇒ **massive** lookups

- Central tables/trees: **heavy** for maintenance, **brittle** for churns

LEAD = LEArned DHT.

Make DHTs even greater with efficient range query

- Learn the keys' **ECDF**
- Map to **ordered hash** IDs
- DHT routing & ownership
- **Ranges query ⇒ contiguous hash ranges**

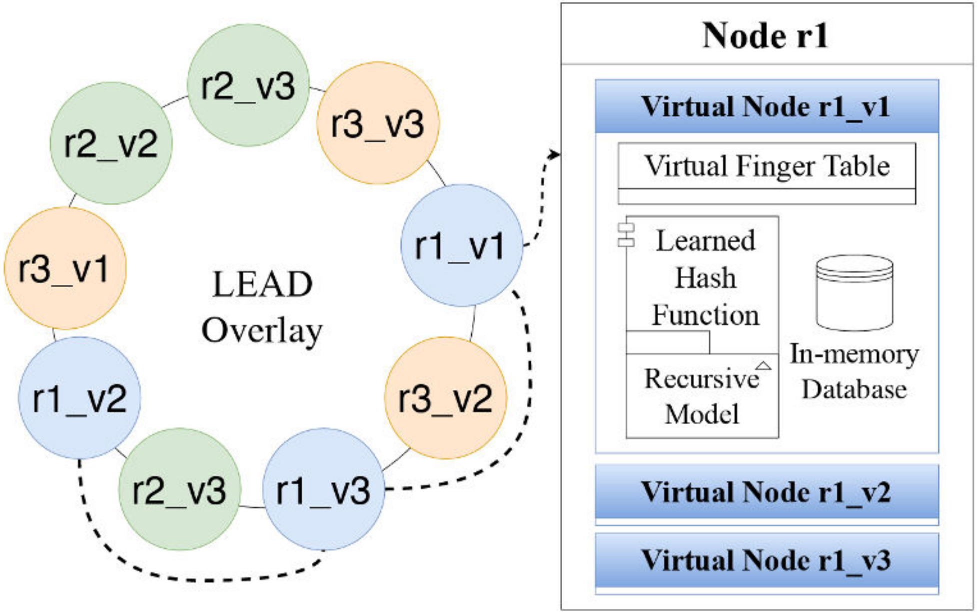
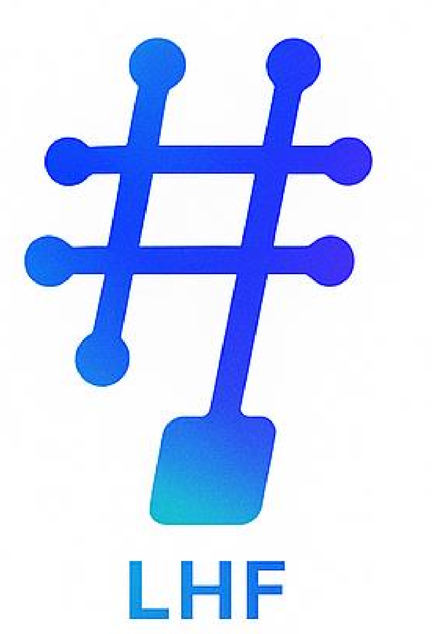


Fig. 2. LEAD System Design

Learned Hash Function (LHF).

Make DHTs even greater with efficient range query



- We argue that a **learned model** can **replace the hash function** to distribute keys in networked systems.
- LEAD **first** introduces the concept of the **Learned Hash Function** under the realm of distributed key-value systems.

Learned Hash Function (LHF).

Mapping keys into order-preserving hash

- Learning hash with **Recursive Model (RMI)** structure
- At each stage, the sub-model determines the appropriate child model to engage for a key.

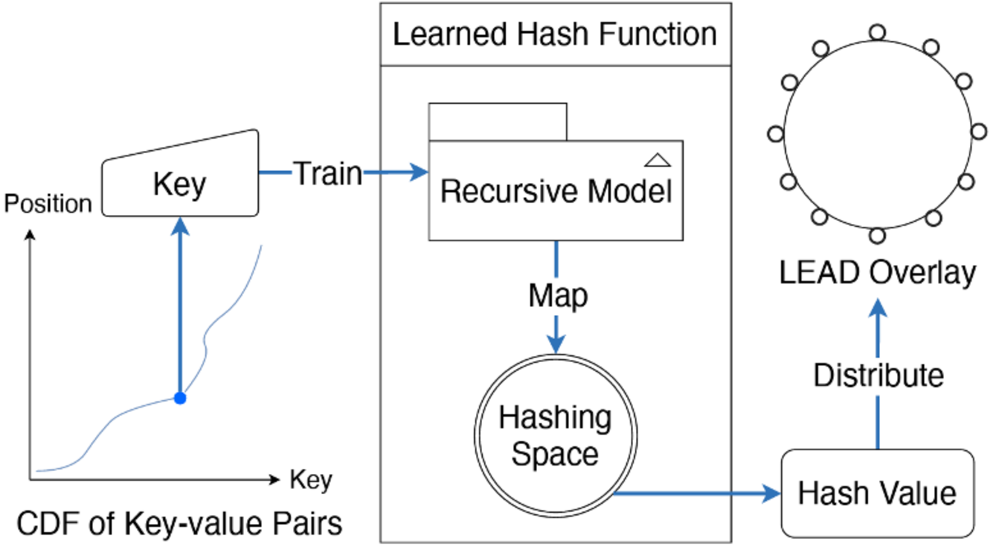


Fig. 3. Key mapping with a learned hash function

Learned Hash Function (LHF).

Mapping keys into order-preserving hash

- At the final stage, leaf models **predict the relative position of a key** and **map the key to the hashing space**.

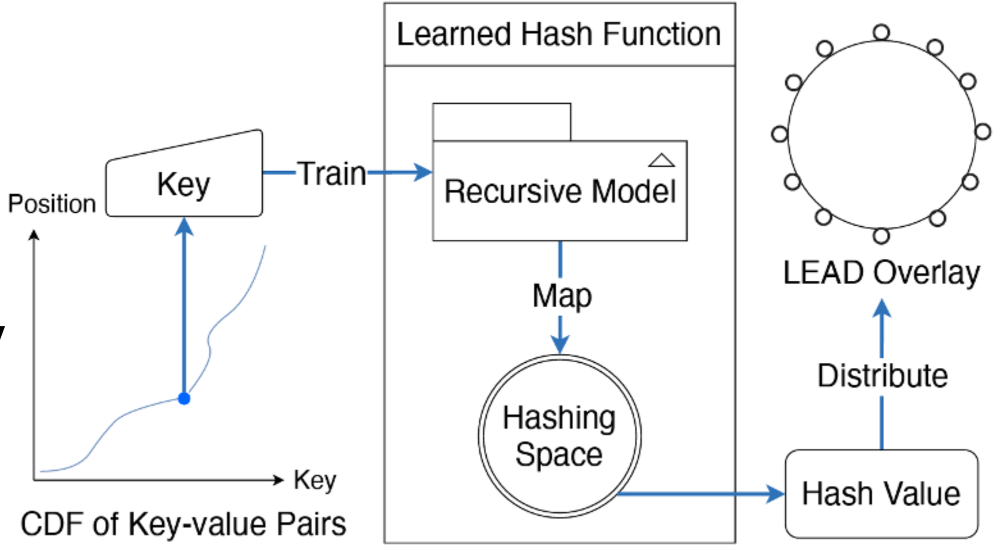


Fig. 3. Key mapping with a learned hash function

Learned Hash Function (LHF).

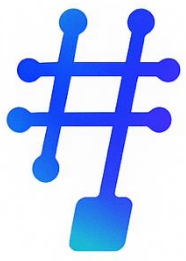
Mapping keys into order-preserving hash

Considering a two-stage LH trained on N key-value pairs, the learned hash function, denoted as $LearnedHASH$, can be articulated as:

$$LearnedHASH(key) = \lfloor \frac{N}{H} \times f_2^{\lfloor \frac{B^{a} \times f_1(x)^{b}}{N} \rfloor}(K) \rfloor \quad (1)$$

[a] B referred to as the branching factor that determimines the number of "buckets" that data is divided into by the stage-one model

[b] f_i referred to as the ith stage model



Data Retrieval with LHF.

Peer Addressing

- Route via **picking farthest preceding finger**
- O(log N) hops to owner
- Separated hash space for ownership with ***PeerHASH***

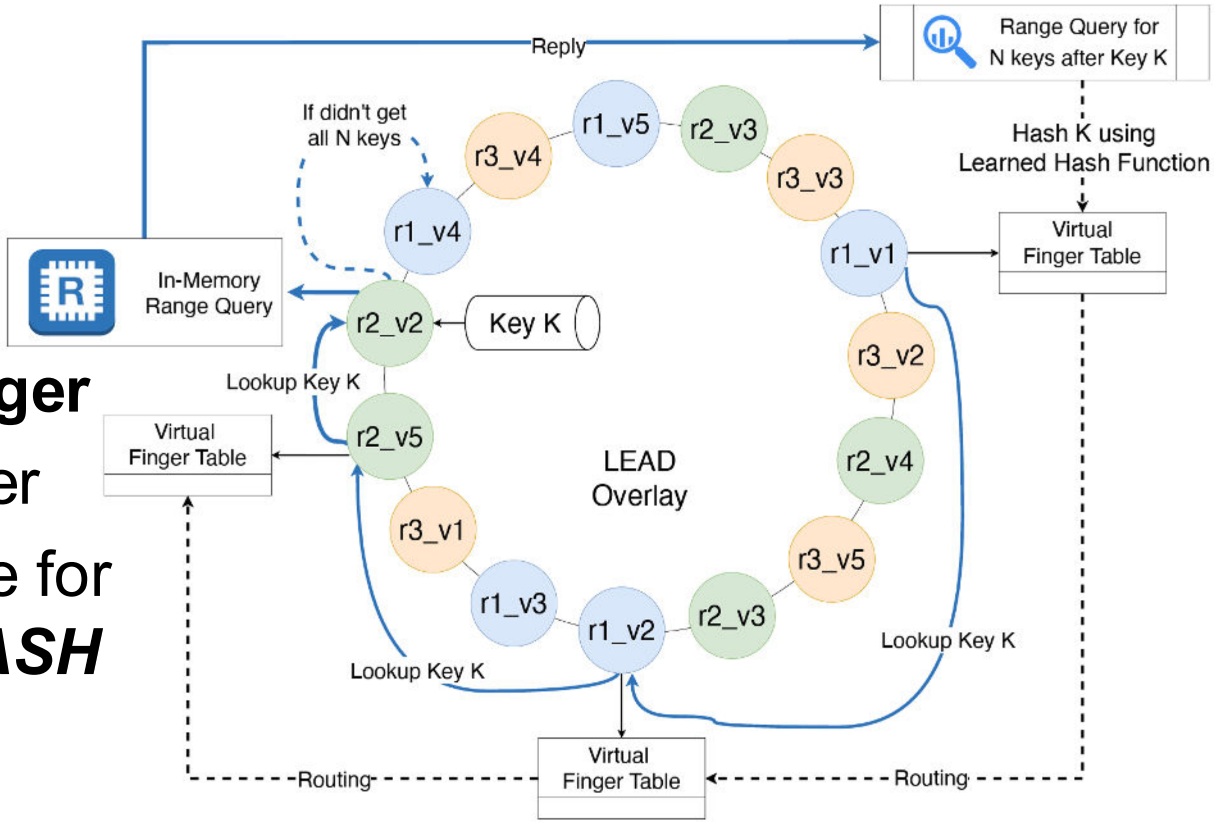


Fig. 5. Range Query in LEAD

Data Retrieval with LHF.

Range Query

- **Hash start key *K*** via Learned hash function
- **Route to owner *S***, do local in-memory range, then **successor chain** if needed (typically 0/1 hops)

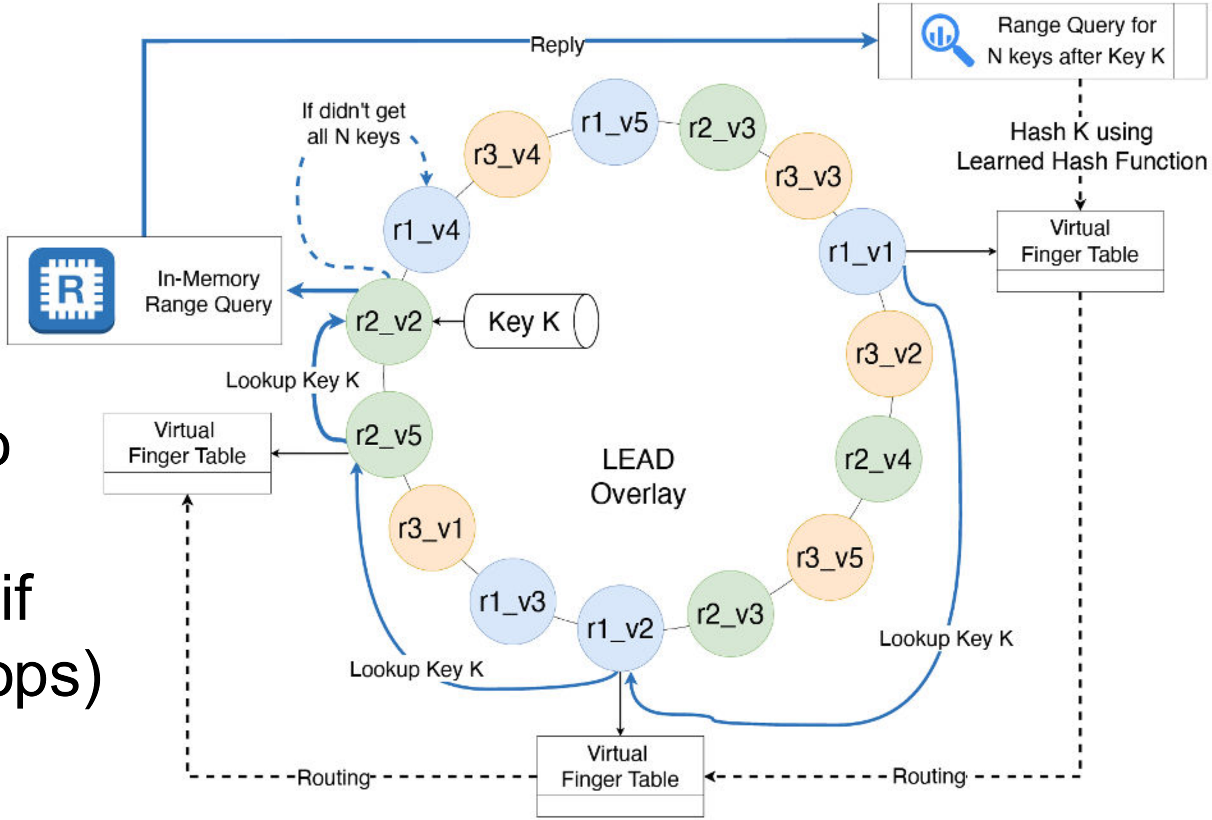


Fig. 5. Range Query in LEAD

Federated Recursive Model (FRM)

Model Update for LH

- **Leaves** refined locally
- **Transient** coordinator aggregates neighbors (**avg-reduce**)
- **Consistency** via heartbeats
- System remains **operational**

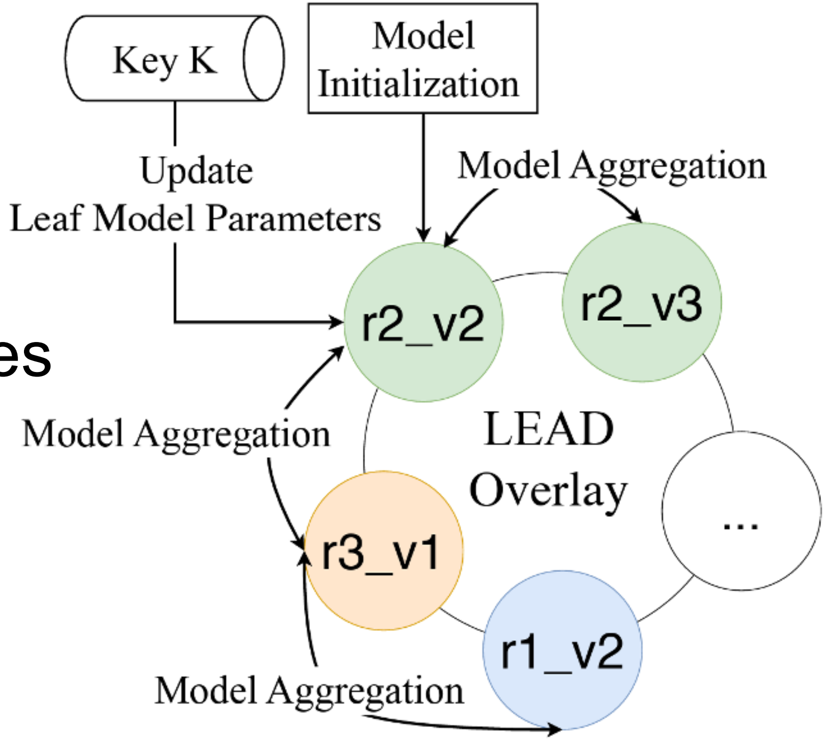


Fig. 4. Collaborative Model Update

LEAD, Leading Performance

Large-scale simulation and Real-world measurement

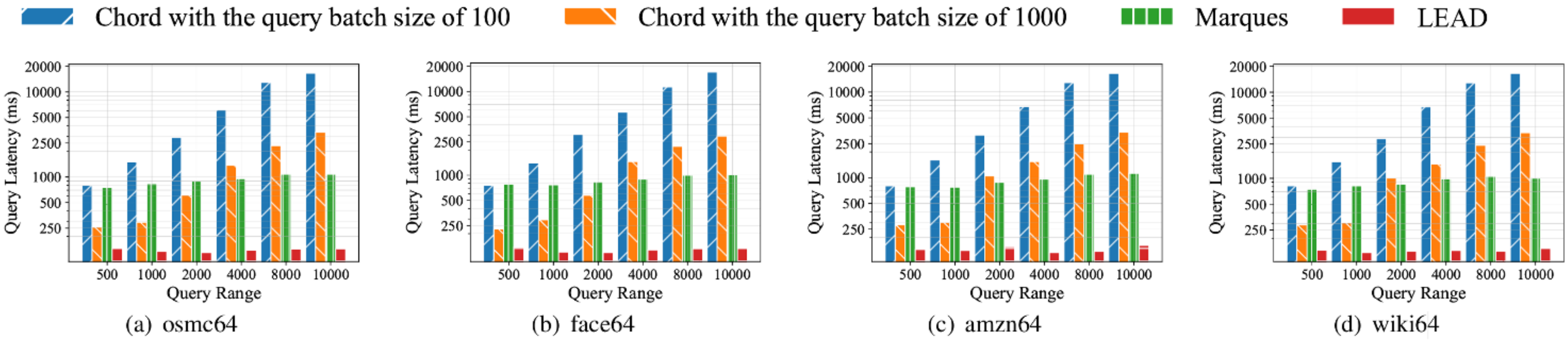


Fig. 6. Latency of range queries on various datasets in the real-machine testbed

- Four baselines covering **batched query** and **range-partition** style (*Marques*)
- LEAD **reducing query latency and message cost by 80% to 90%+**

LEAD, Leading Performance

Large-scale simulation and Real-world measurement

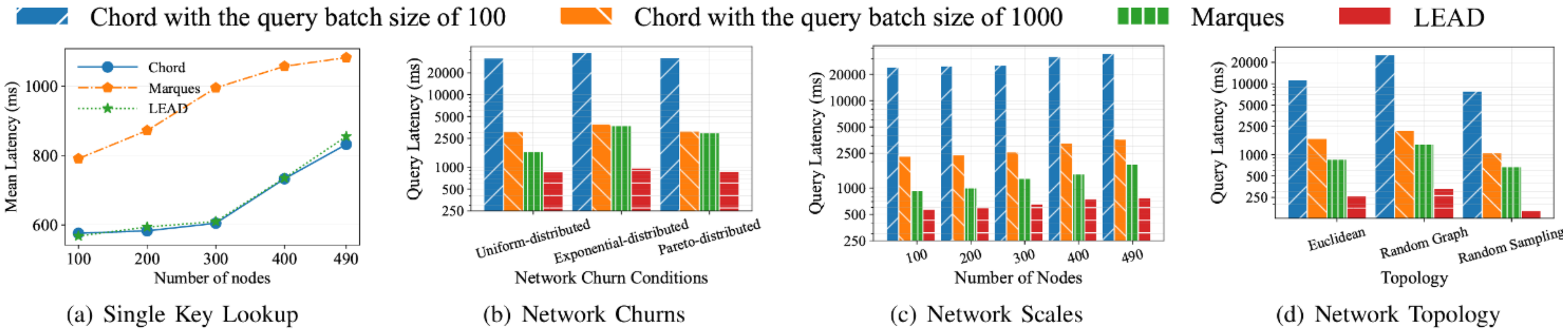


Fig. 9. Latency of range queries under various conditions

- <u>No trade-off with range improvement</u>
- SoTA single-key in O(log N) hops

- **Scale** with network
- Strong network churn **resistance**
- **Minimal** maintenance overhead

LEAD, Leading Performance

Case study I : KV Cache Management for LLM Serving

- 8 × A100×1 nodes, Llama-3 8B, block size 16, **~500M KV blocks**
- Long-DocQA workload; need **contiguous KV ranges**
- LEAD ≈ upper-bound retrieval latency; **dramatically fewer messages** than DHT; **better load balance and scalability**

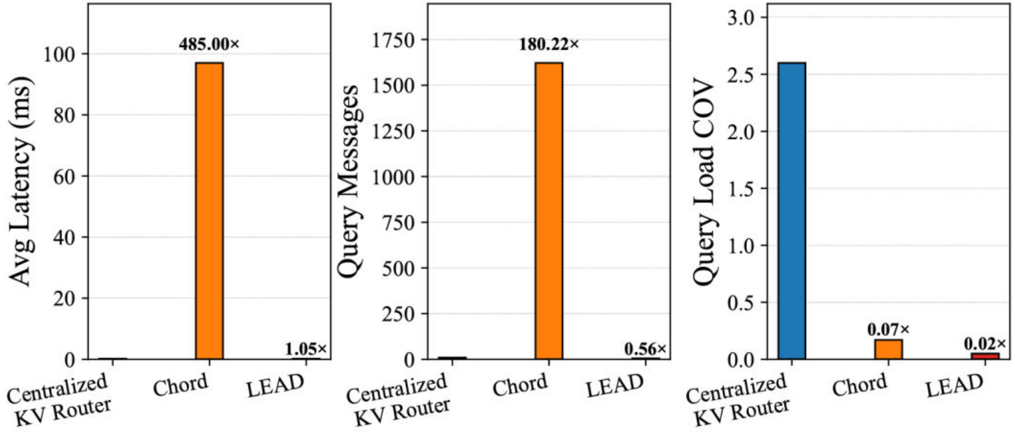


Fig. 15. KV Cache Management

LEAD, Leading Performance

Additional details available

- Detailed setups and results
- Load balancing with *Shadow Balancer*
- Model update during churn
- Benchmark of Learned models
- Shadow Balancer, Learned Model overhead analysis
- Stabilization and Failure Recovery
- Security Considerations
- Case study II : InterPlanetary File System (IPFS)
- Case study III: Blockchain application

LHF

Expanding LEAD into Emerging Domains

Integrating learned models with distributed systems.

- LEAD, a novel distributed key-value storage and lookup system designed to **significantly enhance the efficiency** of range queries by incorporating learned models with DHTs.
- LEAD opens <u>a completely new field for further research on integrating learned models with distributed systems</u>. E.g., **Vector Databases**, **LLM Cache**, **Edge-AI**, **CDN**, **Blockchains**, etc.
- The implementation details are publicly available at https://github.com/ShengzeWang/LEAD.

LHF

Thank you!

Learning to Hash for Networked Systems

- **Opportunity: <u>Your Domain!</u>** , Switch, Blockchain, Edge, etc.
- **Our efforts with LHF: LEAD** (LEArned DHT, ICNP’25), **Vortex** (Vector Overlay for Similarity Search and Delivery, undergoing, ICNP’25 Best Poster), **Knowledge Delivery for LLM Serving** (undergoing)
- Also, check our recent work on **Open LLM Serving**:

GenTorrent: Scaling Large Language Model Serving with An Overlay Network (arXiv:2504.20101)

LHF

Connect! Shengze@ucsc.edu | Shengze.io