Mitigating Asymmetric Read and Write Costs in Cuckoo Hashing for Storage Systems

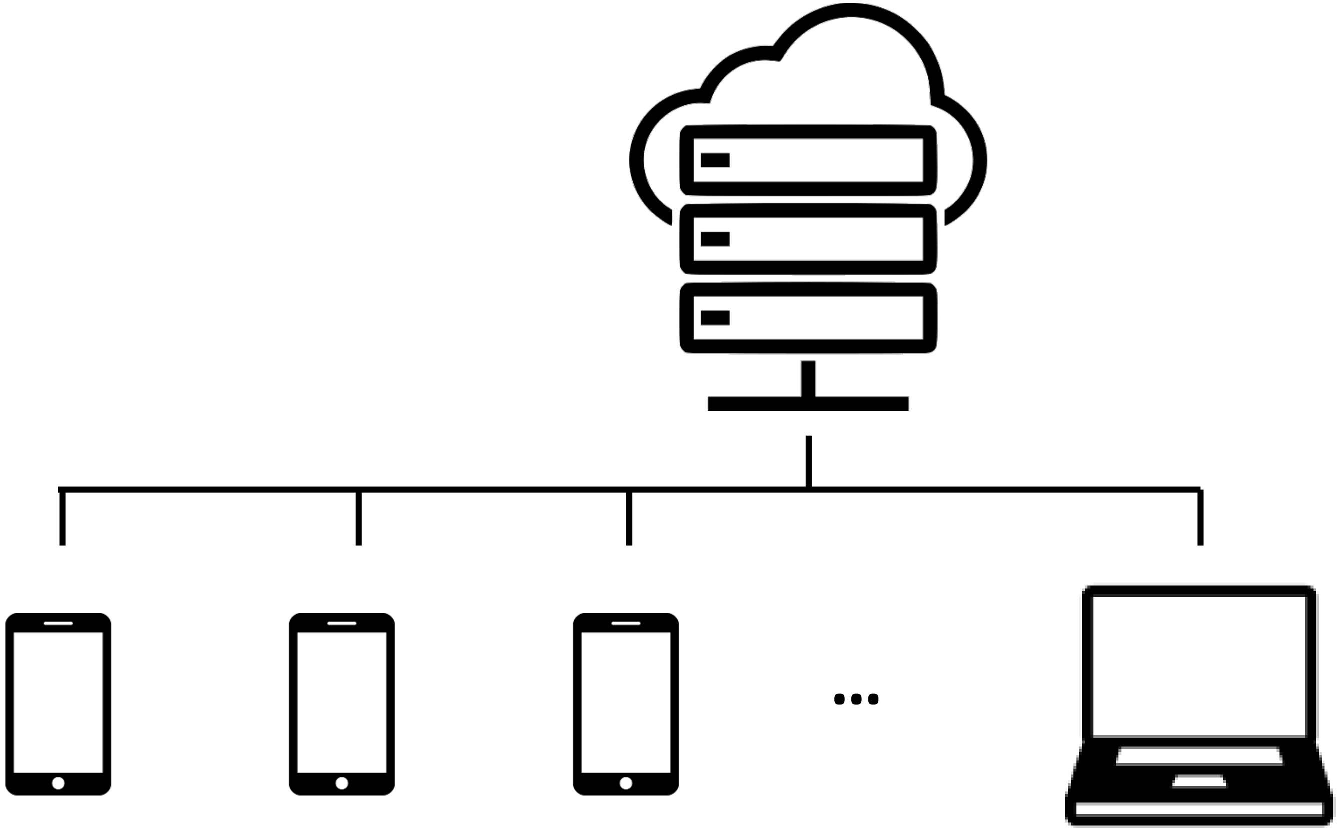
Yuanyuan Sun, Yu Hua, Zhangyu Chen, Yuncheng Guo

Huazhong University of Science and Technology

USENIX ATC 2019

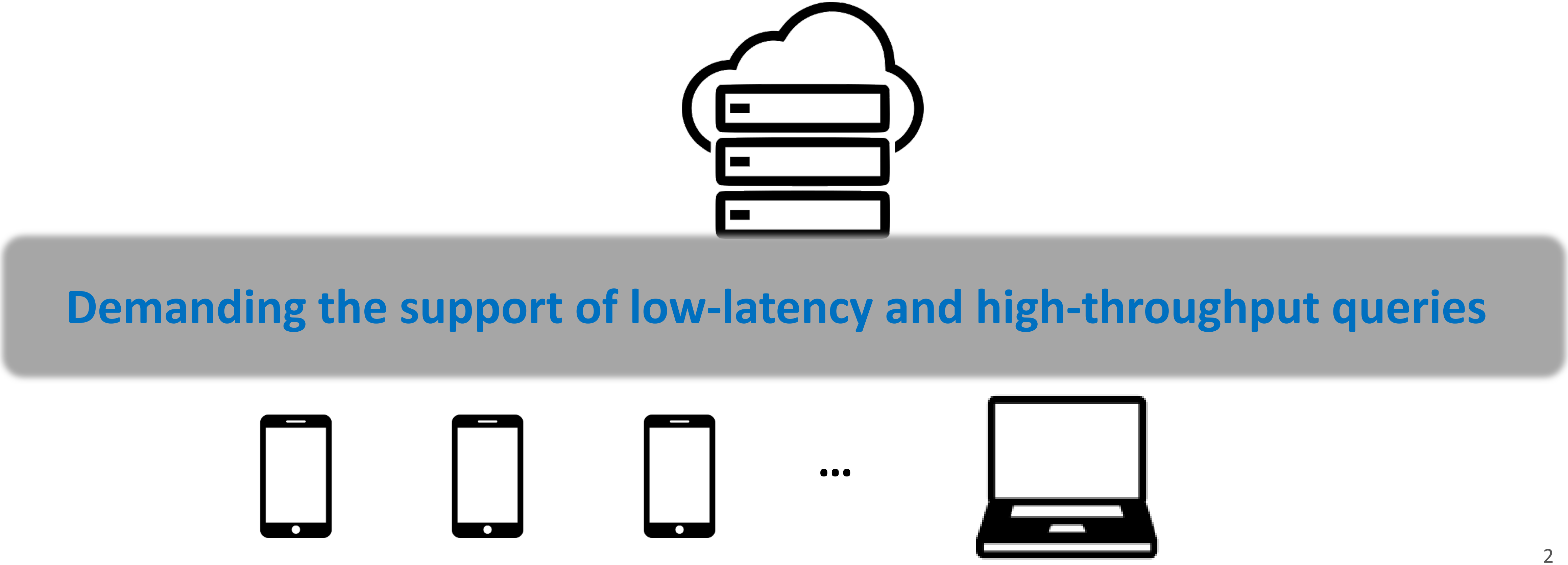
Query Services in Cloud Storage Systems

- **Large amounts of data**
  - **300 new profiles and more than 208 thousand photos per minute [September 2018@Facebook]**



Query Services in Cloud Storage Systems

- **Large amounts of data**
  - **300 new profiles and more than 208 thousand photos per minute [September 2018@Facebook]**



Demanding the support of low-latency and high-throughput queries

Hash structures

✓ **Constant-scale read performance**

- **Widely used in key-value stores and relational databases**

redis

MEMCACHED

monetdb

HyPer

Hash structures

✓ **Constant-scale read performance**

- **Widely used in key-value stores and relational databases**

redis

MEMCACHED

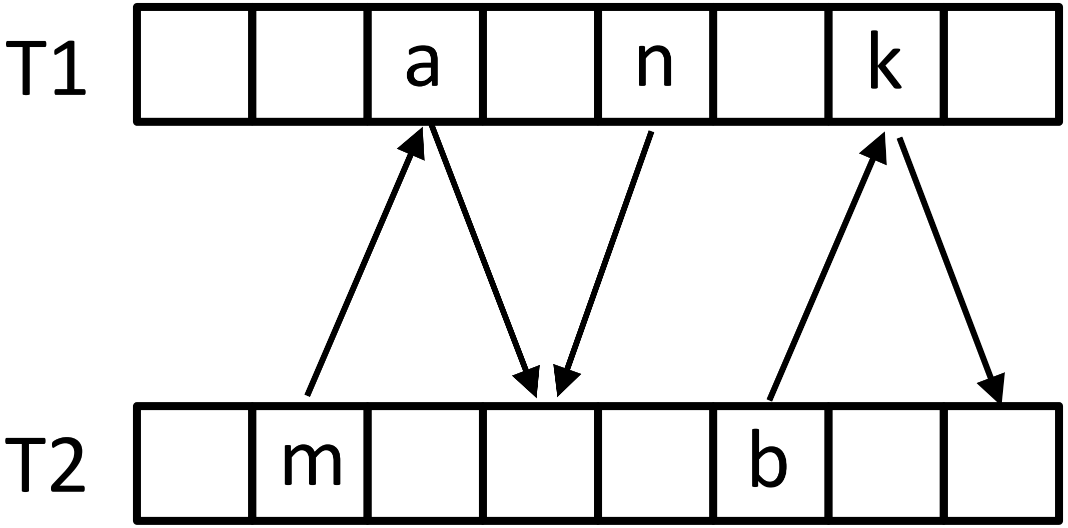
monetdb

HyPer

χ **High latency for handling hash collisions**

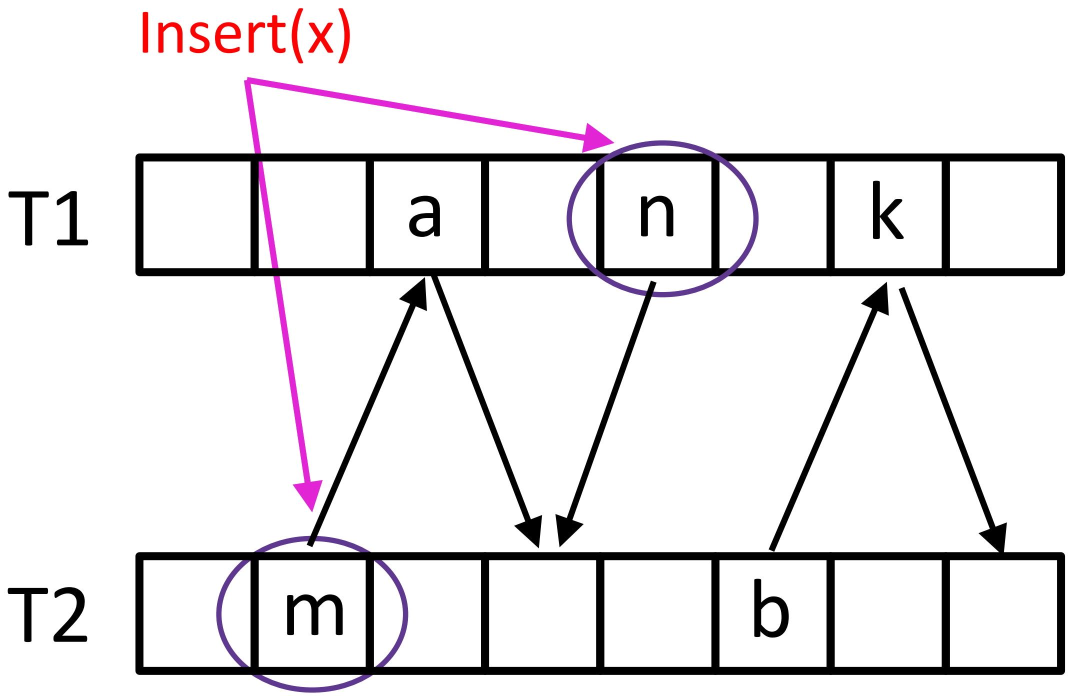
Cuckoo Hashing

- **Multi-choice hashing**
- **Handling hash collisions: kick-out operations**



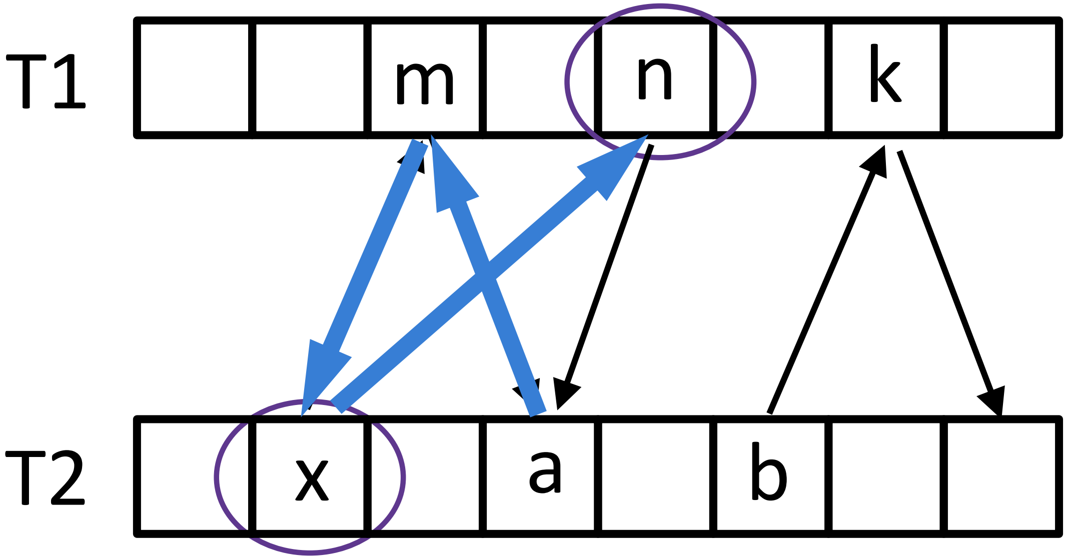
Cuckoo Hashing

- **Multi-choice hashing**
- **Handling hash collisions: kick-out operations**



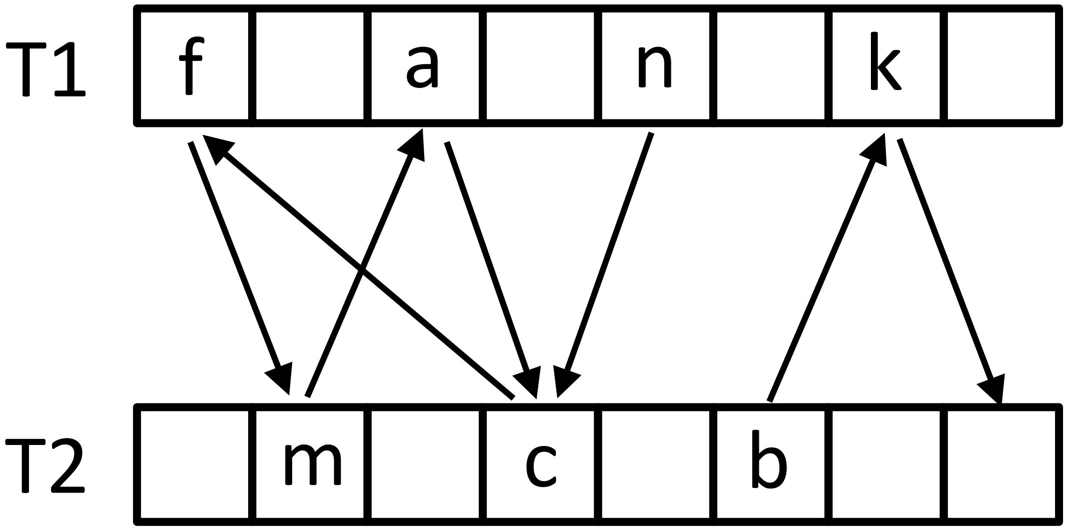
Cuckoo Hashing

- Multi-choice hashing
- Handling hash collisions: kick-out operations



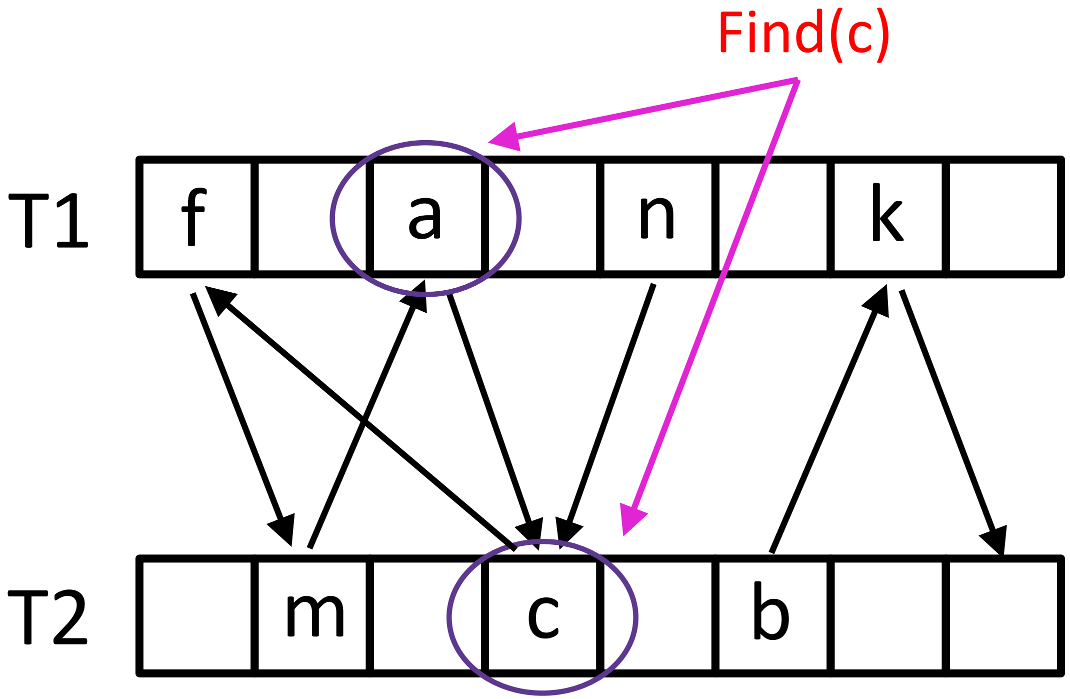
Cuckoo Hashing

- **Multi-choice hashing**
- **Handling hash collisions: kick-out operations**
- **For lookups, only limited positions are probed => O(1) time complexity** 😄



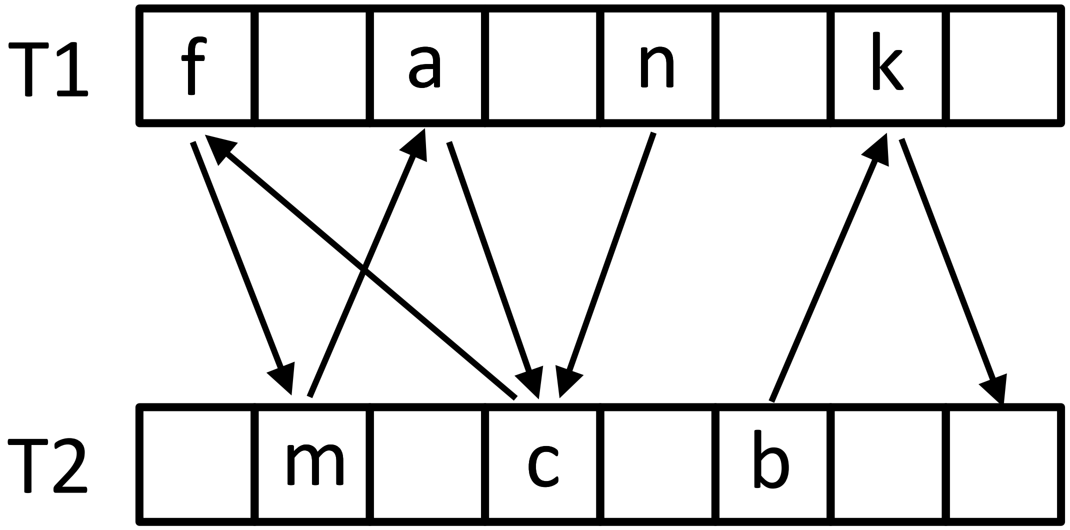
Cuckoo Hashing

- Multi-choice hashing
- Handling hash collisions: kick-out operations
- For lookups, only limited positions are probed => O(1) time complexity 😄



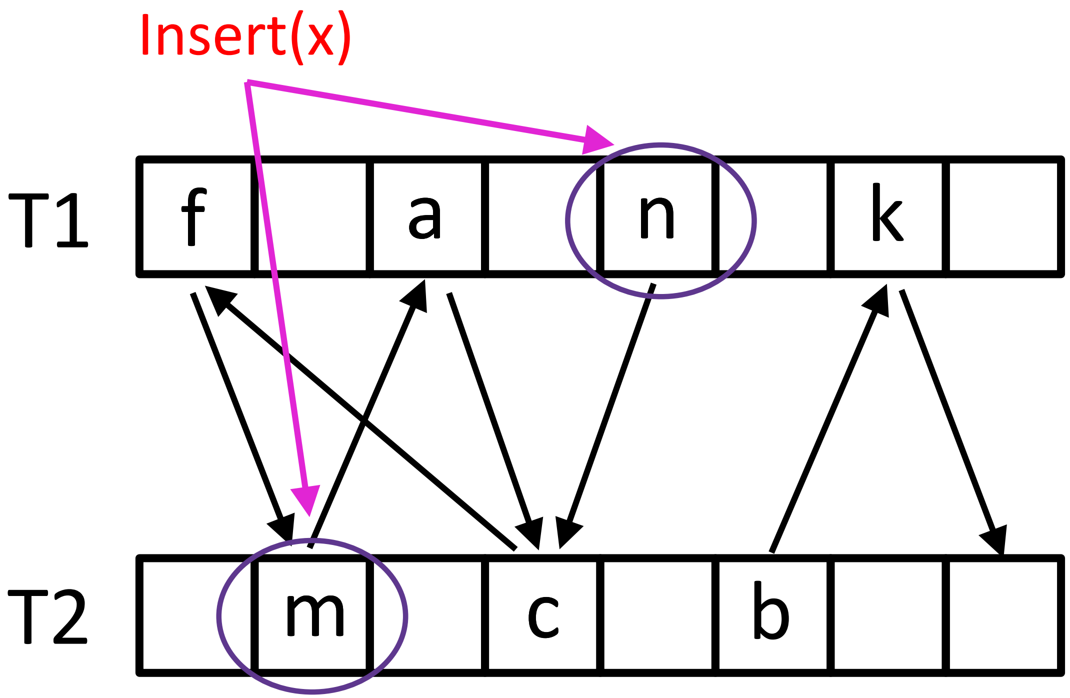
Cuckoo Hashing

- Multi-choice hashing
- Handling hash collisions: kick-out operations
- For lookups, only limited positions are probed => O(1) time complexity 😄
- For insertions, endless loops may occur! => slow-write performance 😔



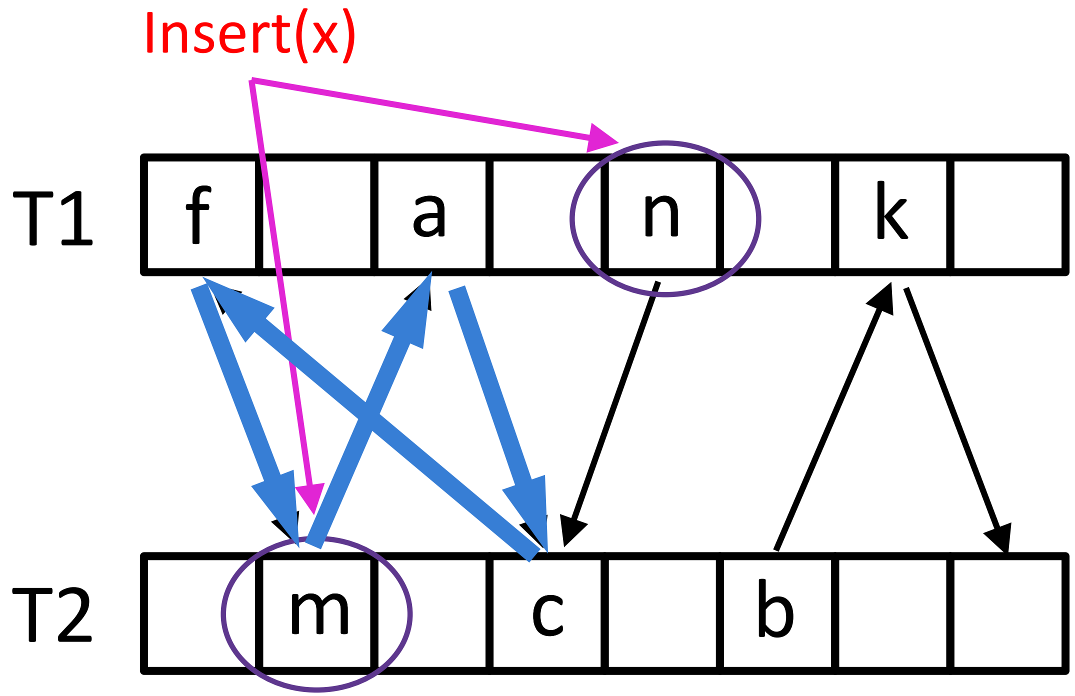
Cuckoo Hashing

- ➢ **Multi-choice hashing**
- ➢ **Handling hash collisions: kick-out operations**
- ➢ **For lookups, only limited positions are probed => O(1) time complexity** 😄
- ➢ **For insertions, endless loops may occur! => slow-write performance** 😔



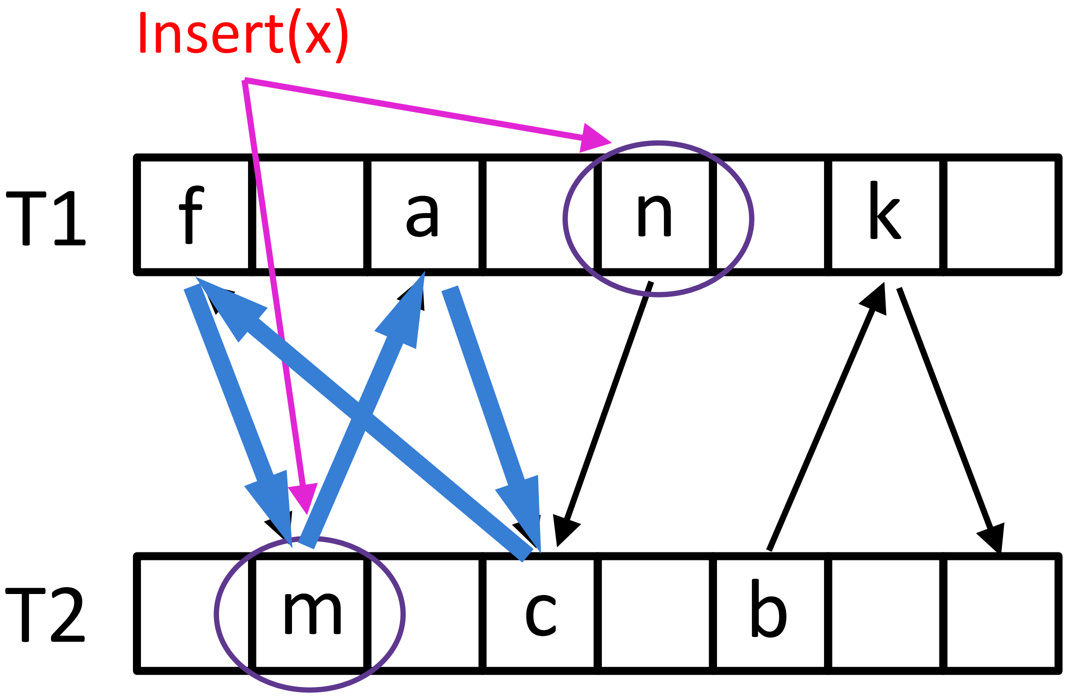
Cuckoo Hashing

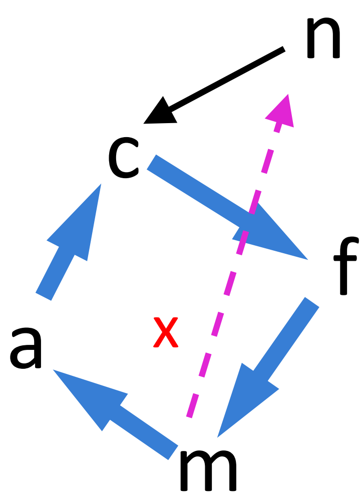
- Multi-choice hashing
- Handling hash collisions: kick-out operations
- For lookups, only limited positions are probed => O(1) time complexity 😄
- For insertions, endless loops may occur! => slow-write performance 😔



Cuckoo Hashing

- Multi-choice hashing
- Handling hash collisions: kick-out operations
- For lookups, only limited positions are probed => O(1) time complexity 😄
- For insertions, endless loops may occur! => slow-write performance 😔

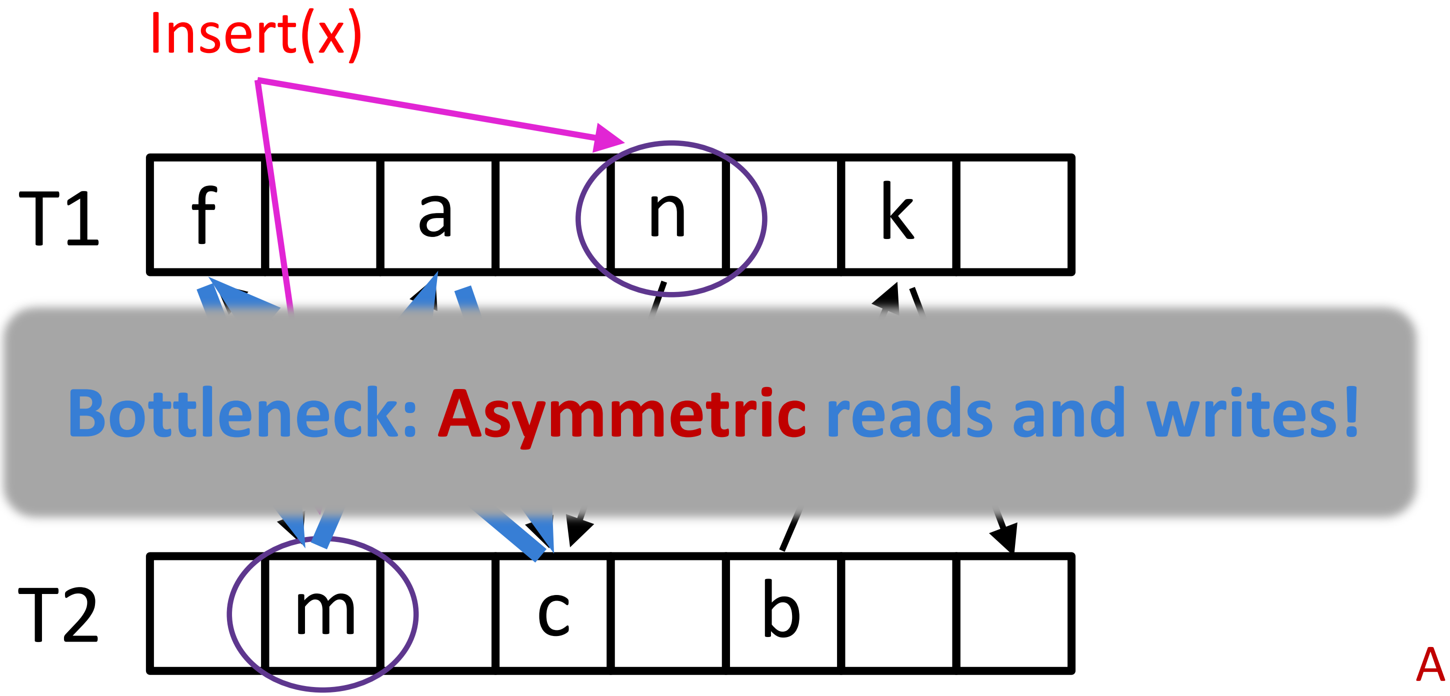




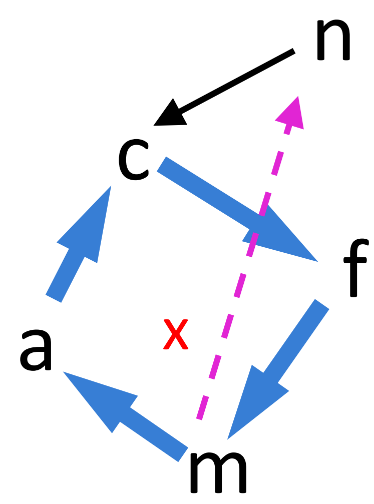
An endless loop occurs!

Cuckoo Hashing

- ➢ **Multi-choice hashing**
- ➢ **Handling hash collisions: kick-out operations**
- ➢ **For lookups, only limited positions are probed => O(1) time complexity** 😄
- ➢ **For insertions, endless loops may occur! => slow-write performance** 😔



Bottleneck: Asymmetric reads and writes!



An endless loop occurs!

Concurrency in Multi-core Systems

- **Existing concurrency strategy for cuckoo hashing**
  - **locking two buckets before each kick-out operation (libcuckoo@EuroSys’14)**

Concurrency in Multi-core Systems

- **Existing concurrency strategy for cuckoo hashing**
  - **locking two buckets before each kick-out operation (libcuckoo@EuroSys’14)**

- **Challenges:**
  - **Poor insertion performance**
  - **Poor scalability**

Concurrency in Multi-core Systems

- **Existing concurrency strategy for cuckoo hashing**
  - **locking two buckets before each kick-out operation (libcuckoo@EuroSys'14)**

- **Challenges:**
  - **Poor insertion performance**
  - **Poor scalability**

- **Design goal:**
  - **A high-throughput and concurrency-friendly cuckoo hash table**

Our Approach: CoCuckoo

- **Pseudoforests to predetermine endless loops**

- **Efficient concurrency strategy**
  - **A graph-grained locking mechanism**
  - **Concurrency optimization to reduce the length of critical path**

- **Higher throughput than state-of-the-art scheme, i.e., libcuckoo**

Mitigating Asymmetric Read and Write Costs in Cuckoo Hashing for Storage Systems

Parallelism & Synchronization, Technical Sessions Track 2
10:25 am – 10:45 am, Thursday, July 11

USENIX ATC 2019