Fast Software Cache Design for Network Appliances

Dong Zhou, Huacheng Yu,
Michael Kaminsky, David G. Andersen

Flow Caching in Open vSwitch

Microflow Cache

Exact Match

Single Hash Table

Flow Caching in Open vSwitch

srcAddr=10.1.2.3, dstAddr=12.4.5.6, srcPort=15213, dstPort=80 → output: 1
srcAddr=12.4.5.6, dstAddr=10.1.2.3, srcPort=80, dstPort=15213 → output: 2
srcAddr=12.4.5.6, dstPort=13.1.2.3, srcPort=80, dstPort=15213 → drop

Microflow Cache
Exact Match
Single Hash Table

Flow Caching in Open vSwitch

Megaflow Cache
Wildcard Match without Priority
Multiple Masked Tables

Miss

Microflow Cache
Exact Match
Single Hash Table

Flow Caching in Open vSwitch

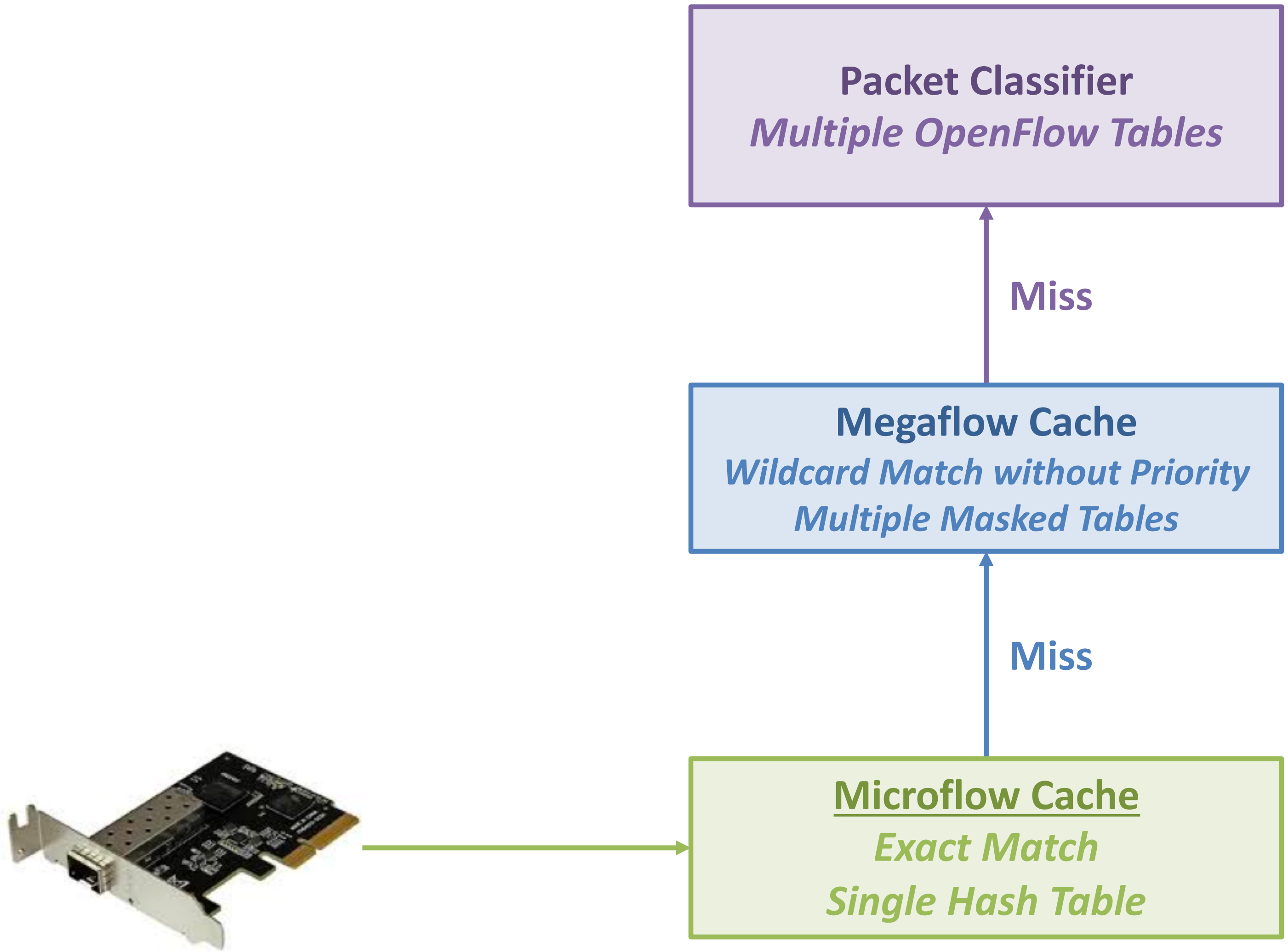
srcAddr=10.0.0.0/8, dstAddr=12.0.0.0/8, srcPort=*, dstPort=* → output: 1
srcAddr=12.0.0.0/8, dstAddr=10.0.0.0/8, srcPort=*, dstPort=* → output: 2
srcAddr=*, dstPort=13.0.0.0/8, srcPort=*, dstPort=* → drop

Megaflow Cache
Wildcard Match without Priority
Multiple Masked Tables

Miss

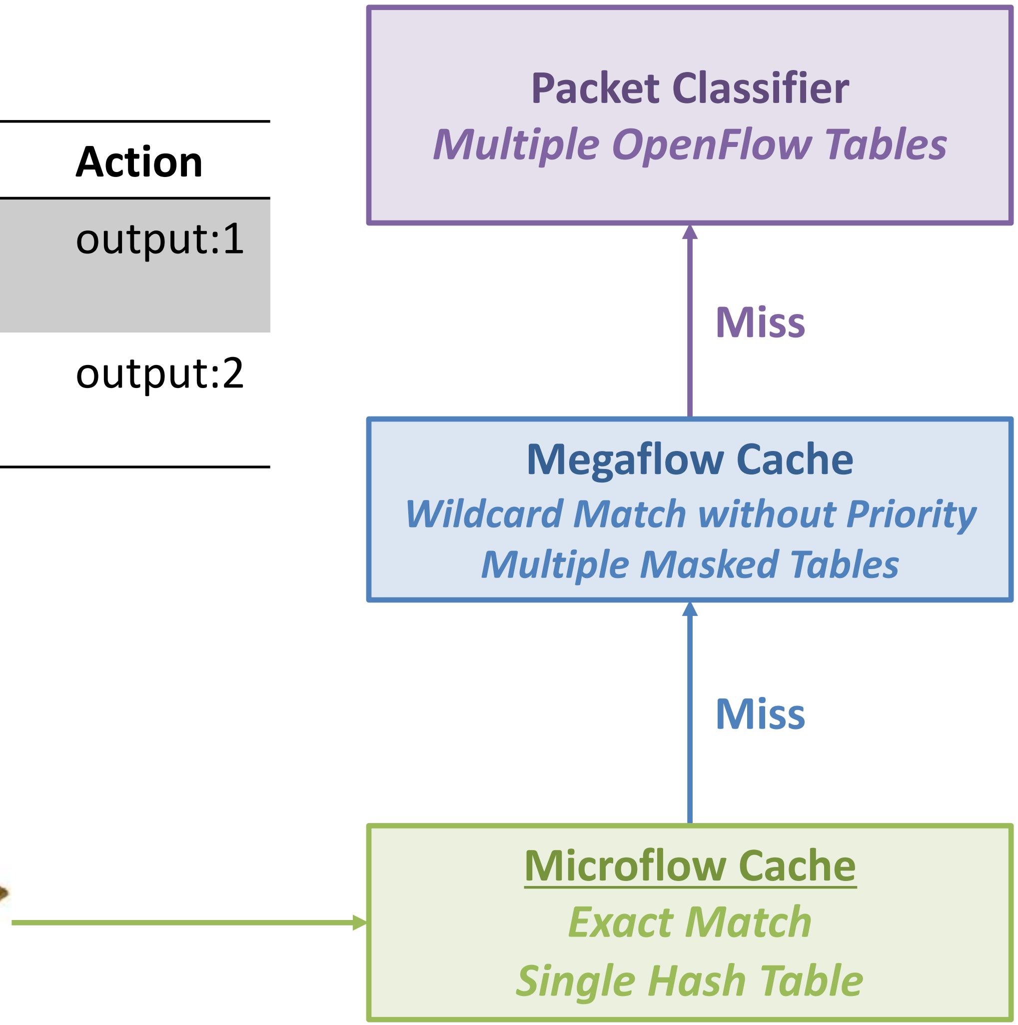
Microflow Cache
Exact Match
Single Hash Table

Flow Caching in Open vSwitch



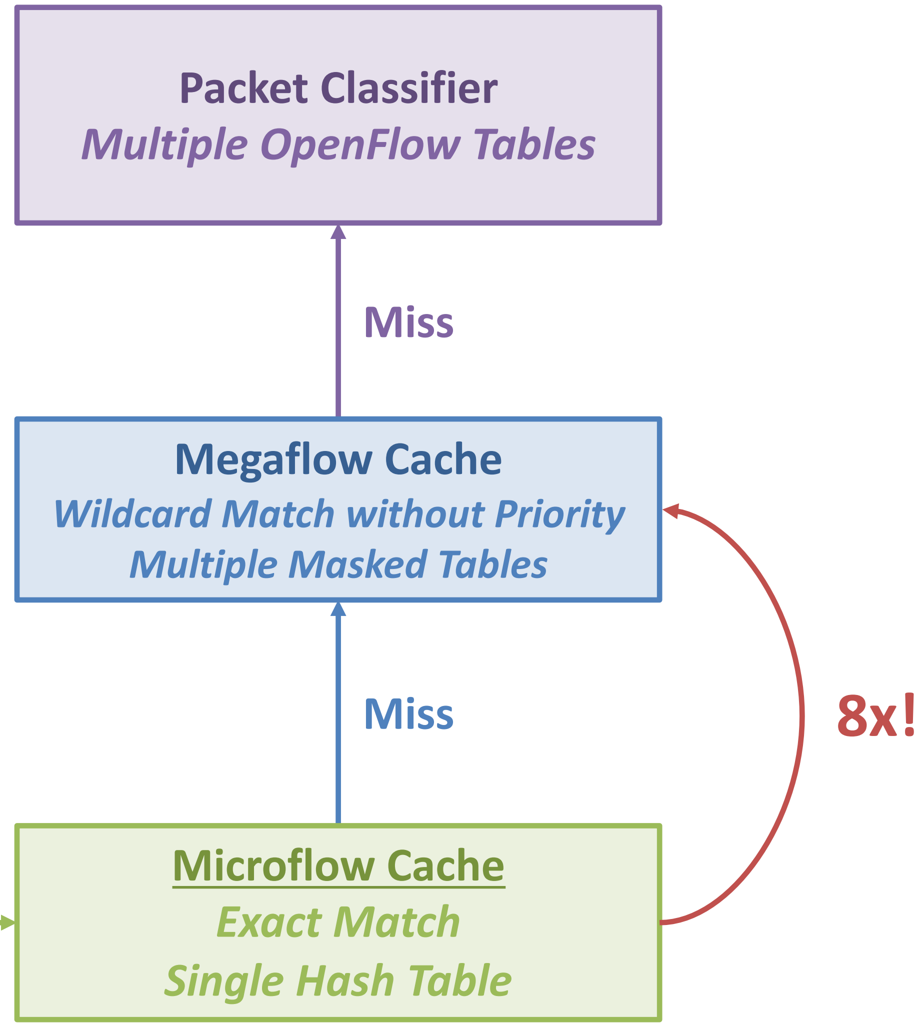
Flow Caching in Open vSwitch

Match	Action
srcAddr==10.0.0.0/8, dstAddr==12.0.0.0/8	output:1
srcAddr==12.0.0.0/8, dstAddr==10.0.0.0/8	output:2

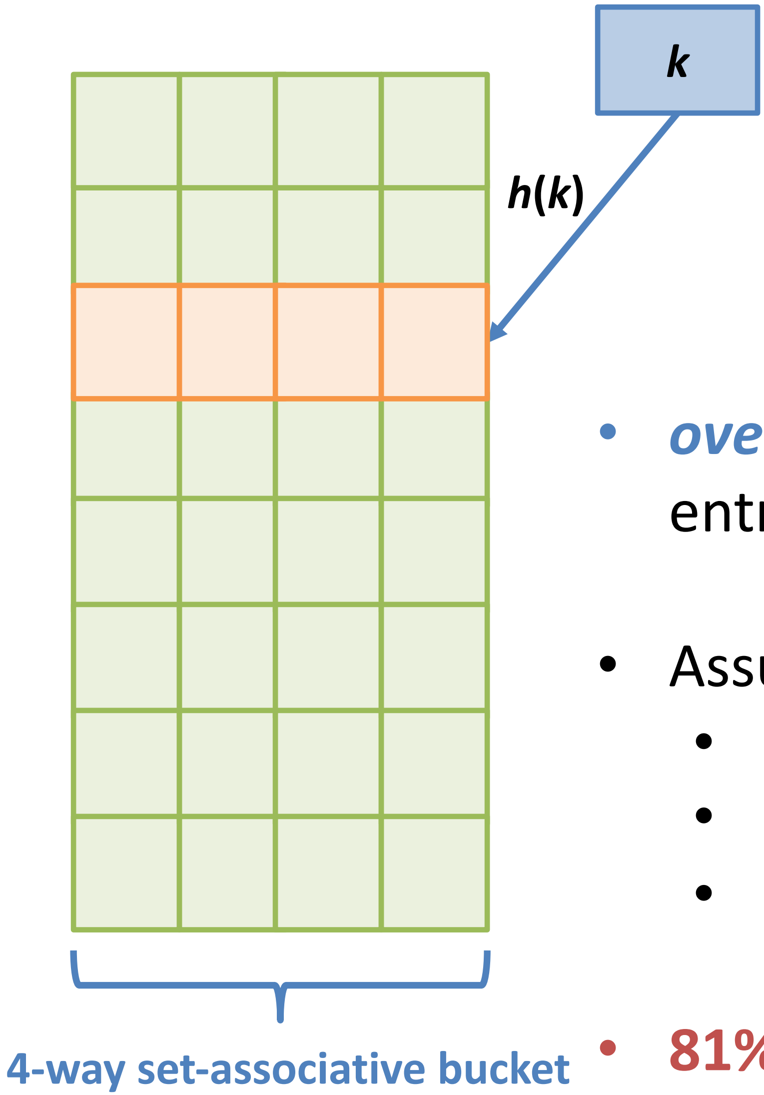


Flow Caching in Open vSwitch

- **Cache Hit Rate**
- **Lookup Latency**



Basic Cache Design

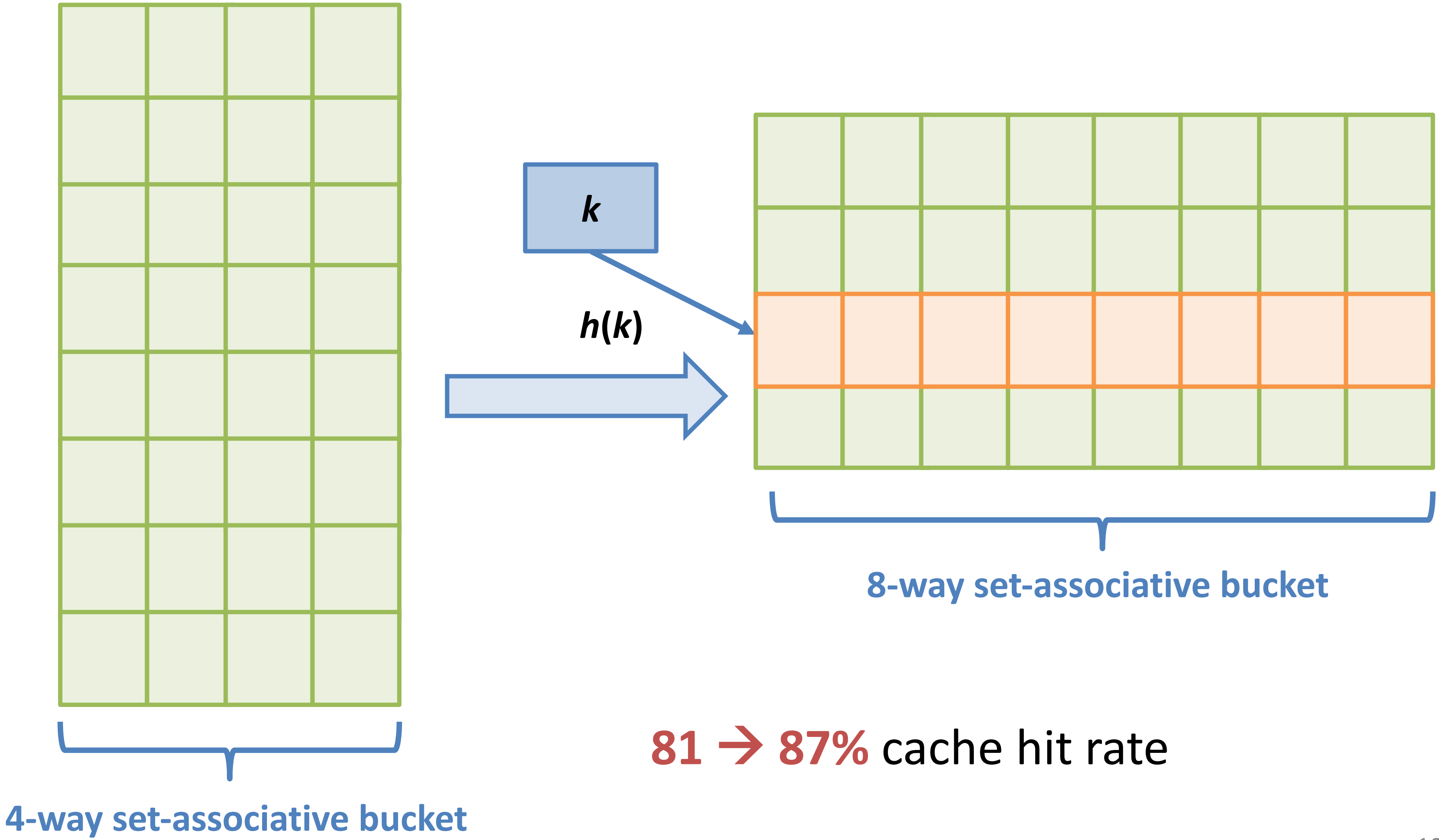


- ***oversubscription factor*** α = # keys / # entries

- Assumption
  - uniform workload
  - random eviction
  - $\alpha = 0.95$

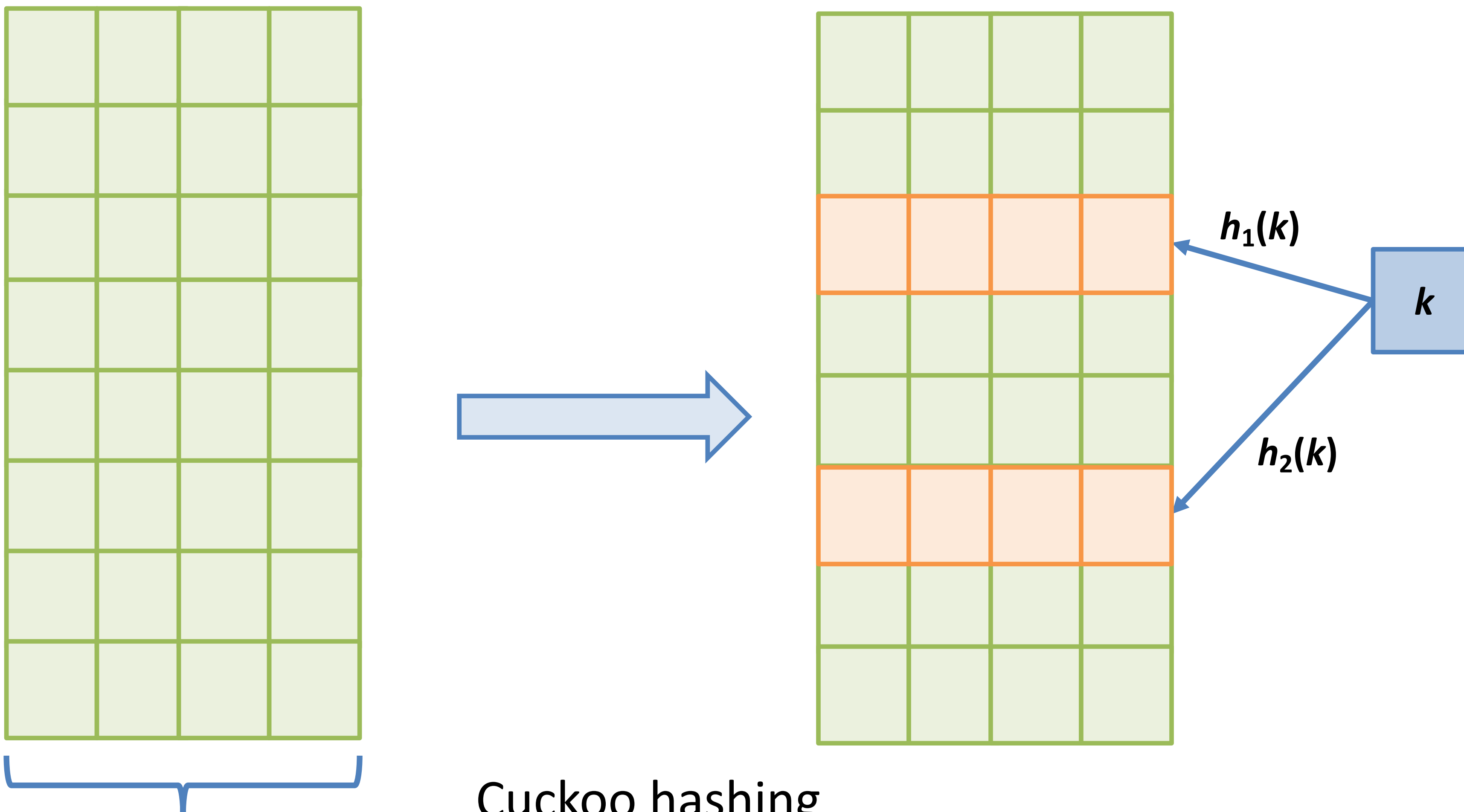
- **81%** cache hit rate

Cache Design: Increase Set-Associativity



81 → 87% cache hit rate

Cache Design:
More Candidate Buckets

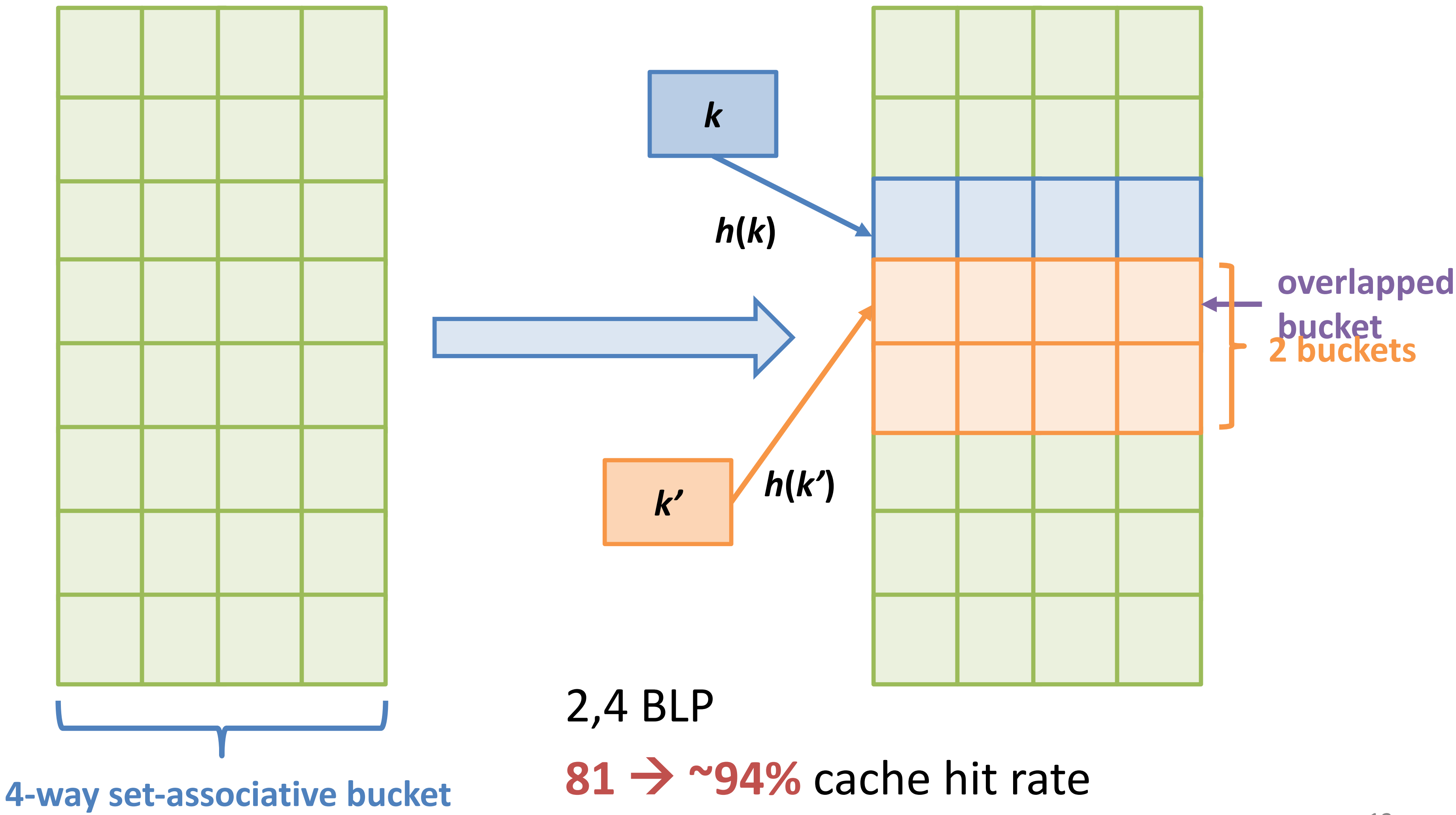


4-way set-associative bucket

Cuckoo hashing

81 → ~99% cache hit rate

Our Solution: Bounded Linear Probing (BLP)



2,4 BLP

81 → ~94% cache hit rate

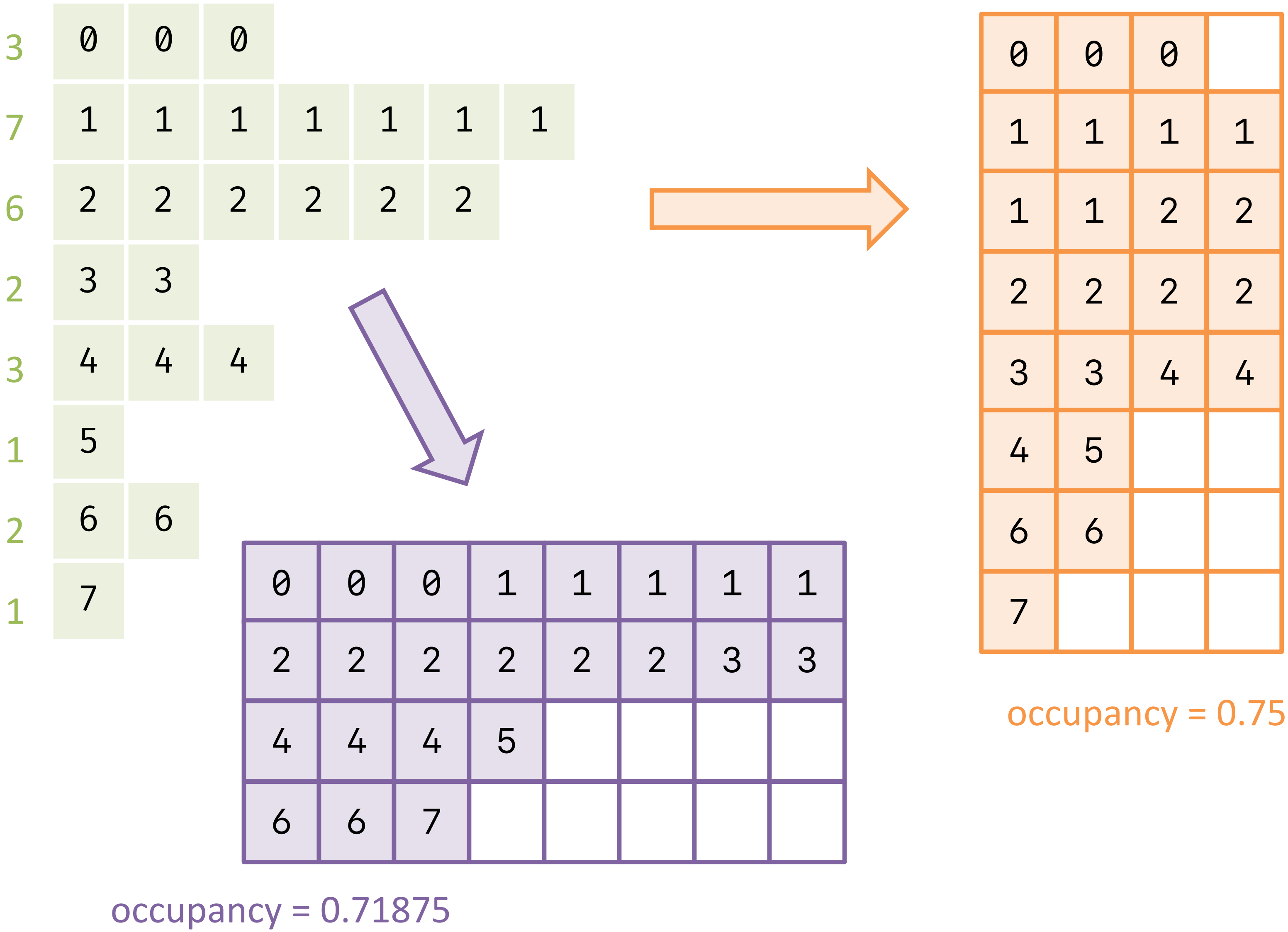
Qualitative Comparison

Design	Lookup Speed (cache line reads)	Hit Rate
4-way set-assoc.	1	~ 81%
8-way set-assoc.	1	~ 87%
2-4 cuckoo	2 random	~ 99%
2-4 BLP	1.5 consecutive	~ 94%

Qualitative Comparison

Design	Lookup Speed (cache line reads)	Hit Rate
4-way set-assoc.	1	~ 81%
8-way set-assoc.	1	**~ 87%**
2-4 cuckoo	2 random	~ 99%
2-4 BLP	1.5 consecutive	**~ 94%**

Why BLP is Better Than Set-Assoc.?



Qualitative Comparison

Design	Lookup Speed (cache line reads)	Hit Rate
4-way set-assoc.	1	~ 81%
8-way set-assoc.	1	~ 87%
2-4 cuckoo	2 random	~ 99%
2-4 BLP	1.5 consecutive	~ 94%

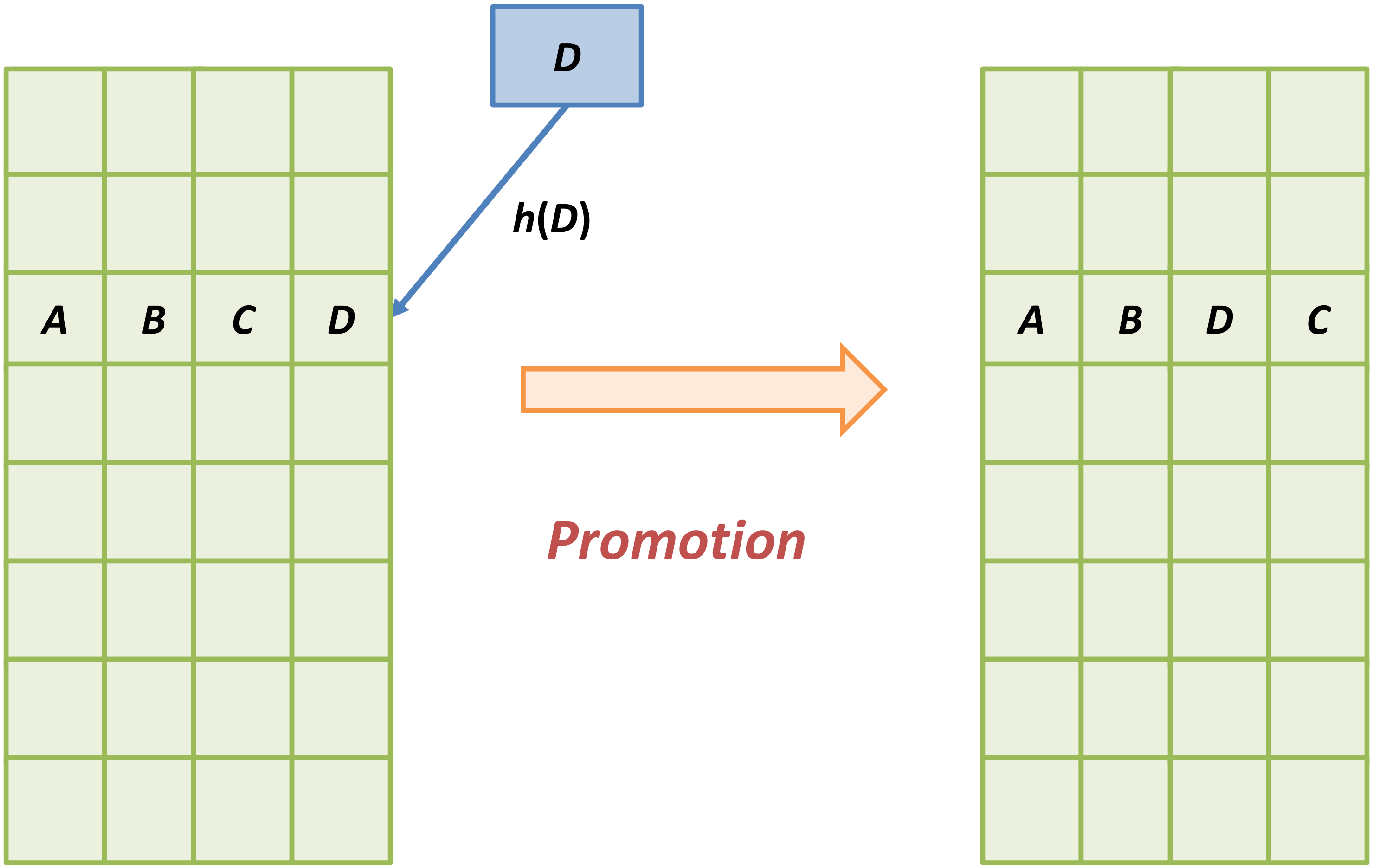
Qualitative Comparison

Design	**Lookup Speed (cache line reads)**	**Hit Rate**
4-way set-assoc.	1	~ 81%
8-way set-assoc.	1	~ 87%
2-4 cuckoo	2 ***random***	~ 99%
2-4 BLP	1.5 ***consecutive***	~ 94%

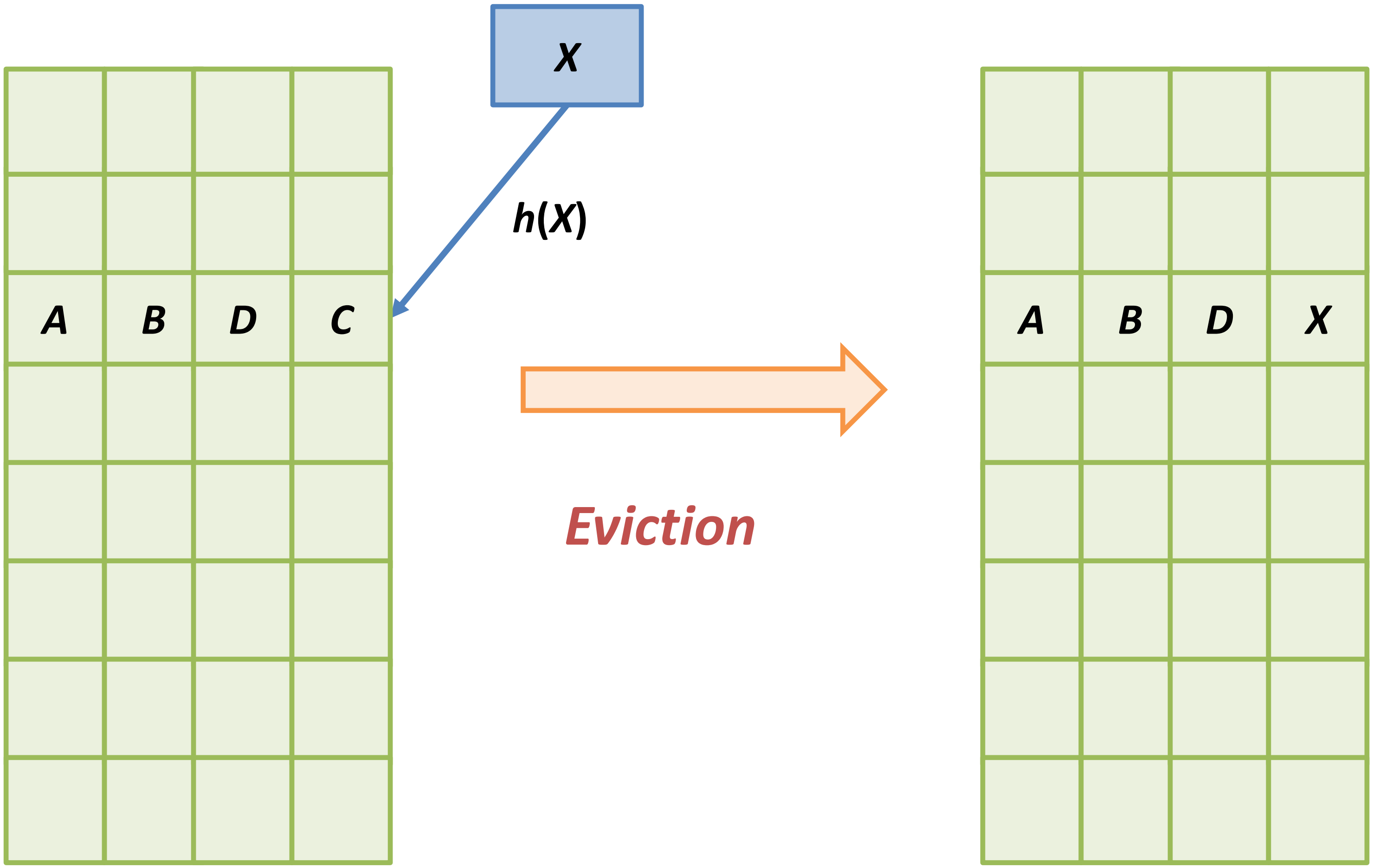
Better Cache Replacement

- Traditional LRU
  - High space overhead
  - CLOCK: 1 bit / key
- Our Solution: Probabilistic Bubble LRU (PBLRU)

PBLRU: Bubbling



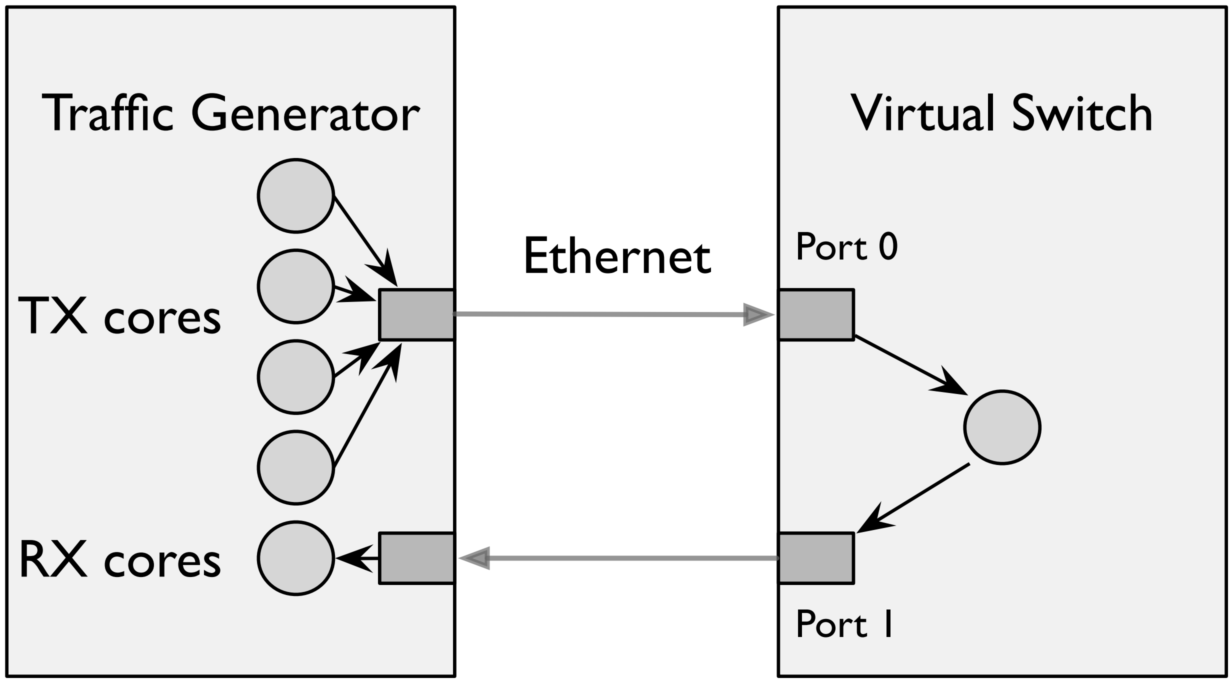
PBLRU: Bubbling



PBLRU

- Basic bubbling
  – Combines both recency and frequency information
- Probabilistic bubbling
  – We only promote every *n*-th cache hit to reduce the number of memory writes
- Applying to 2-4 BLP
  – We choose a random bucket to apply bubbling

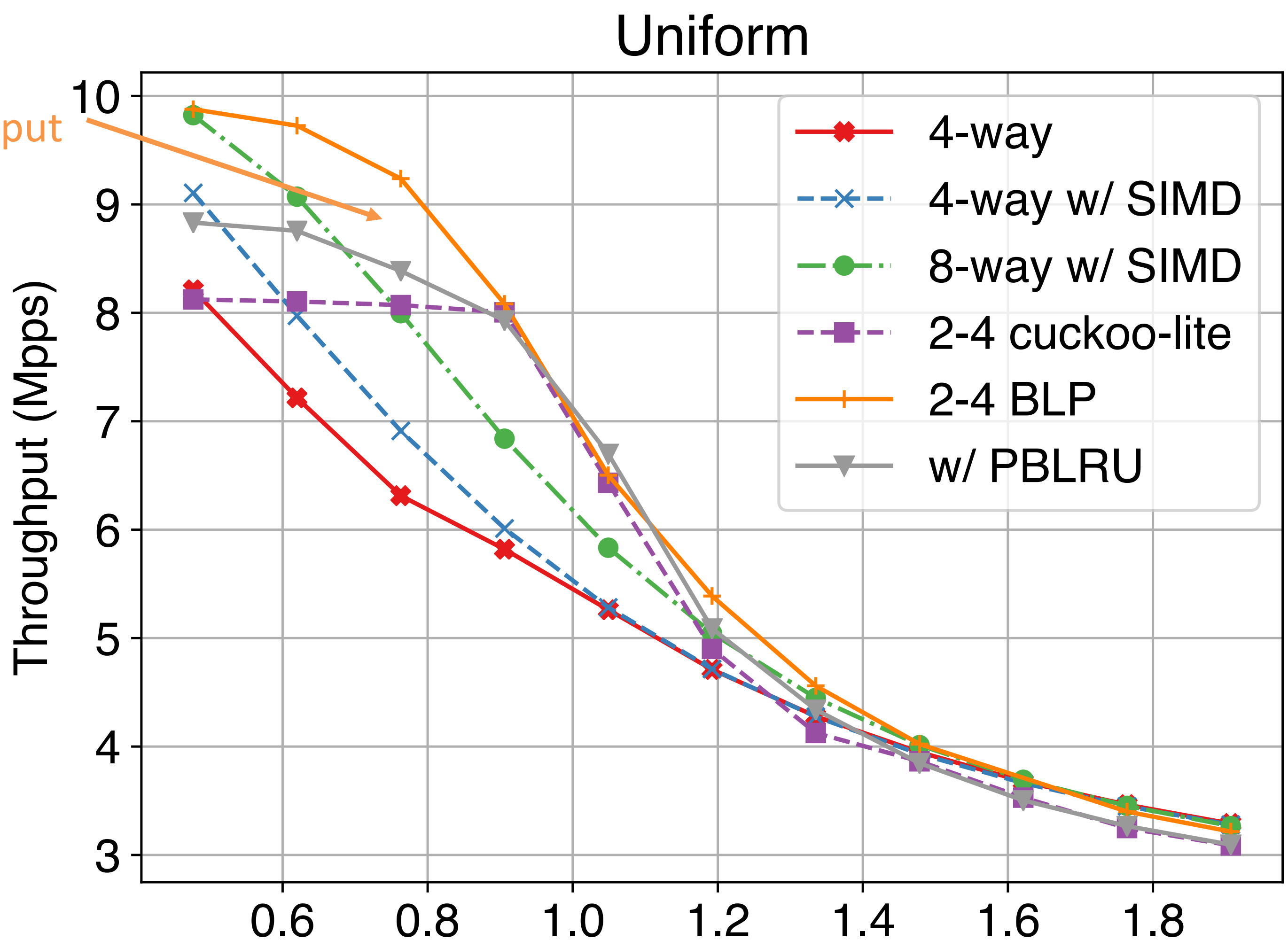
Evaluation



Hardware	Description
CPU	2× Intel Xeon E5-2660v3 CPUs (2.60GHz)
DRAM	160 GiB DDR4 Memory
L3 Cache	2× 24 MiB
NIC	Intel X520 dual-port 10GbE

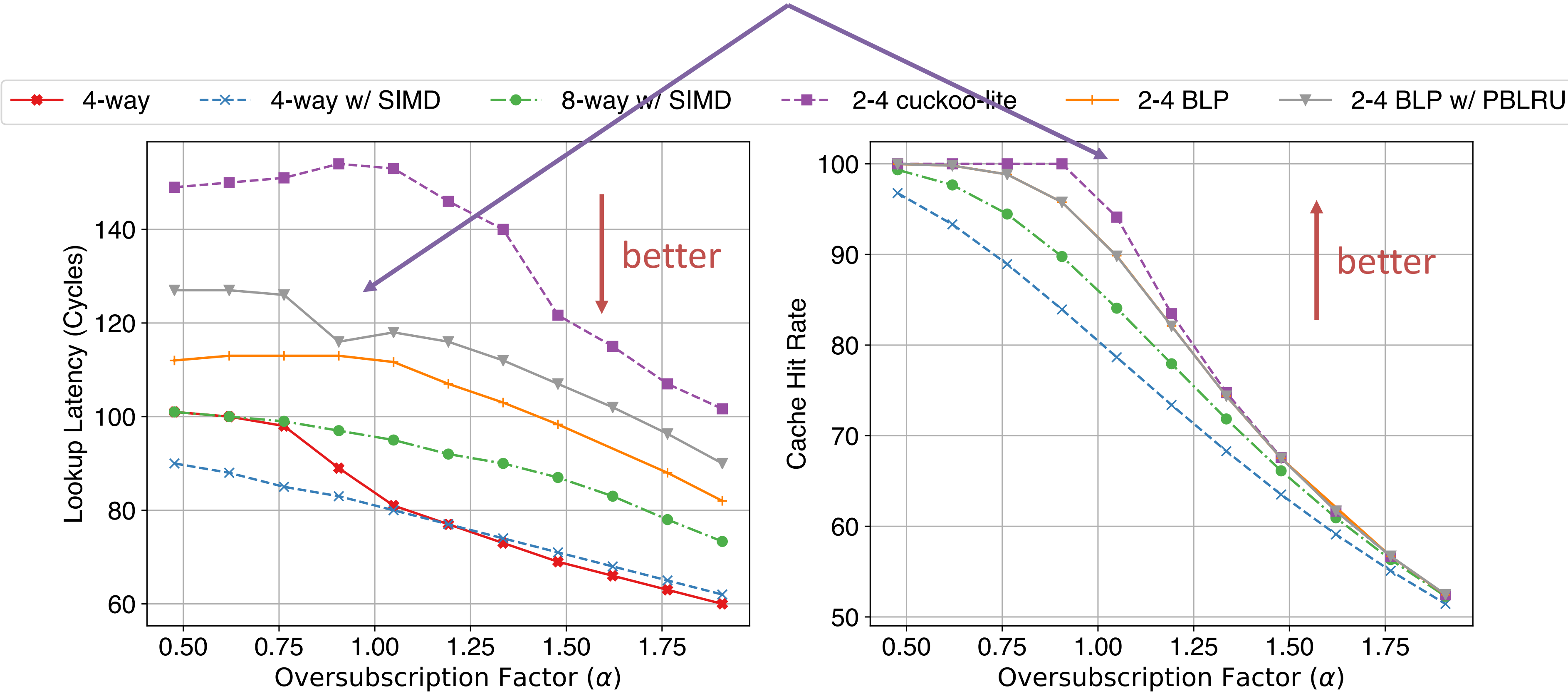
Throughput (Uniform)

15% higher tput

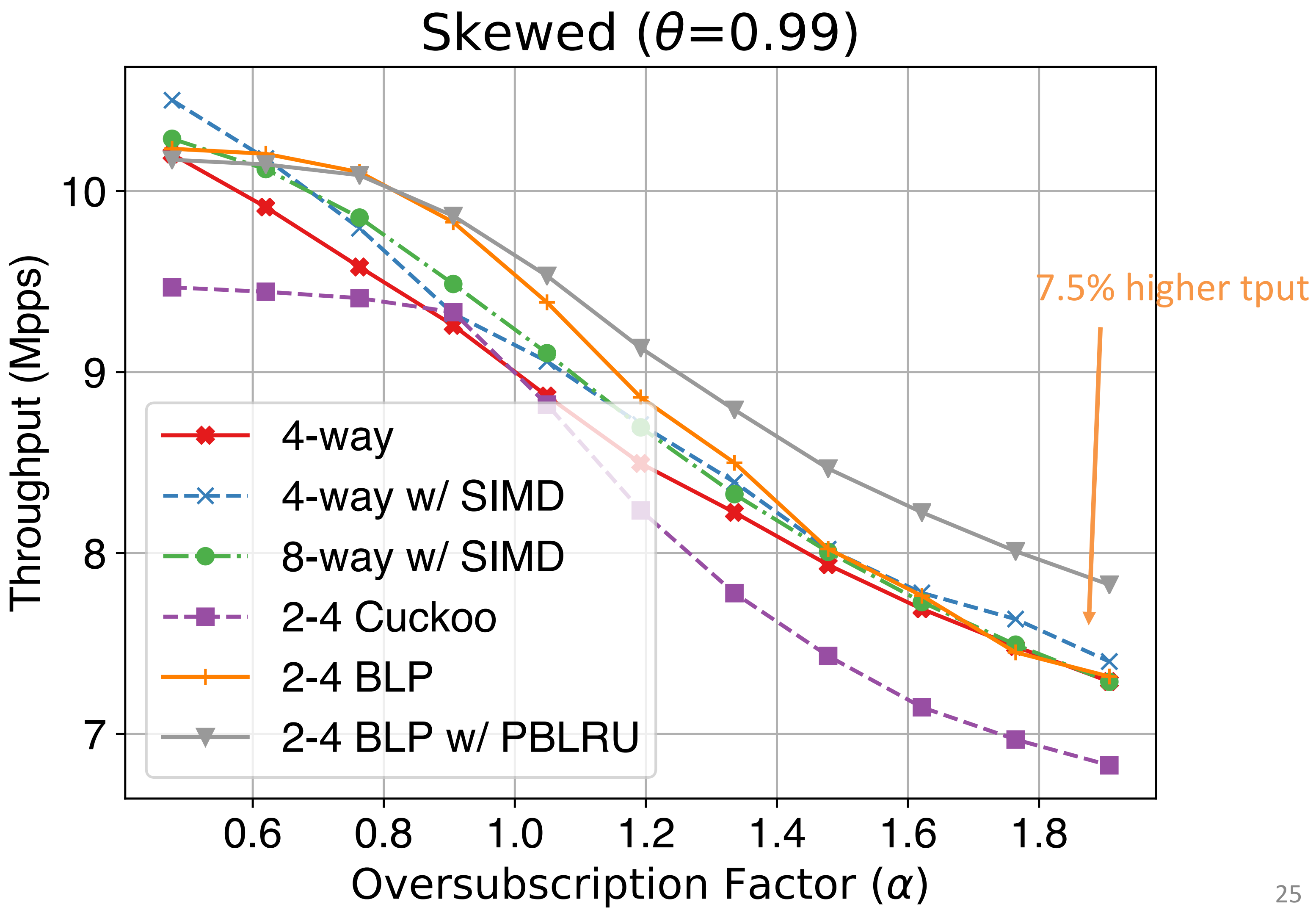


Lookup Latency and Hit Rate

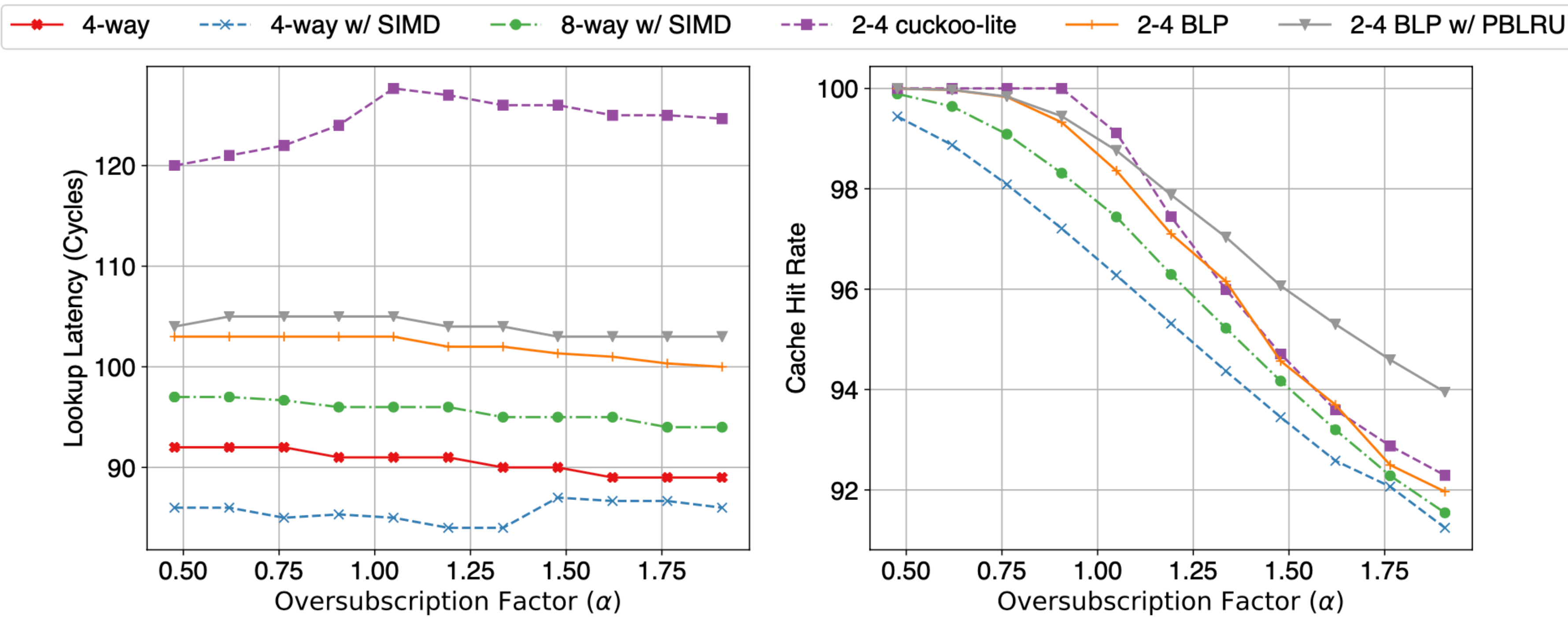
cache hit rate improvement is not enough to compensate for its higher lookup latency



Throughput (Skewed)



Lookup Latency and Hit Rate



Summary

- Bounded Linear Probing
- Probabilistic Bubble LRU
- Balance between Cache Hit Rate and Lookup Latency

Thank You!