Jumanji: The Case for Dynamic NUCA in the Datacenter

Brian C. Schwedock, Nathan Beckmann

MICRO 2020

Electrical & Computer ENGINEERING

Carnegie Mellon University Computer Science Department

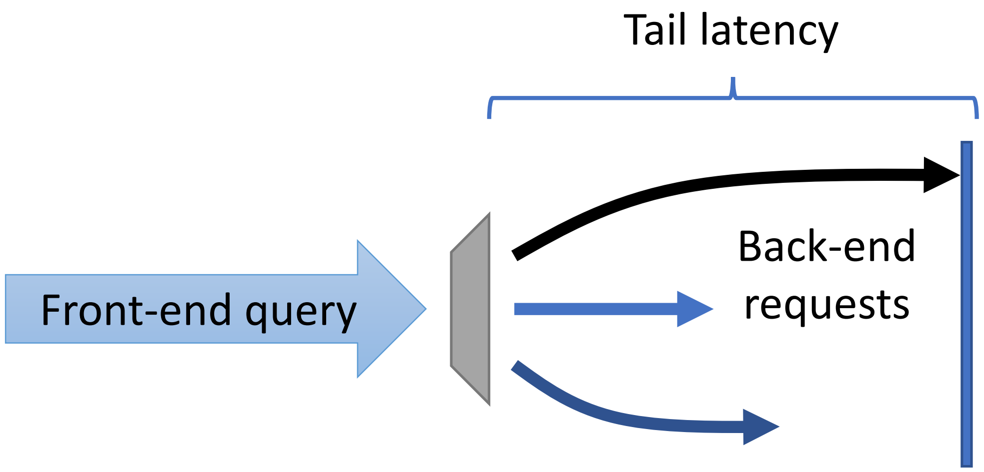
Datacenters care about security and tail latency

- **Security:** data and performance protection among untrusted users (e.g., VMs)
- **Tail latency:** execution time of slowest application requests

Meltdown

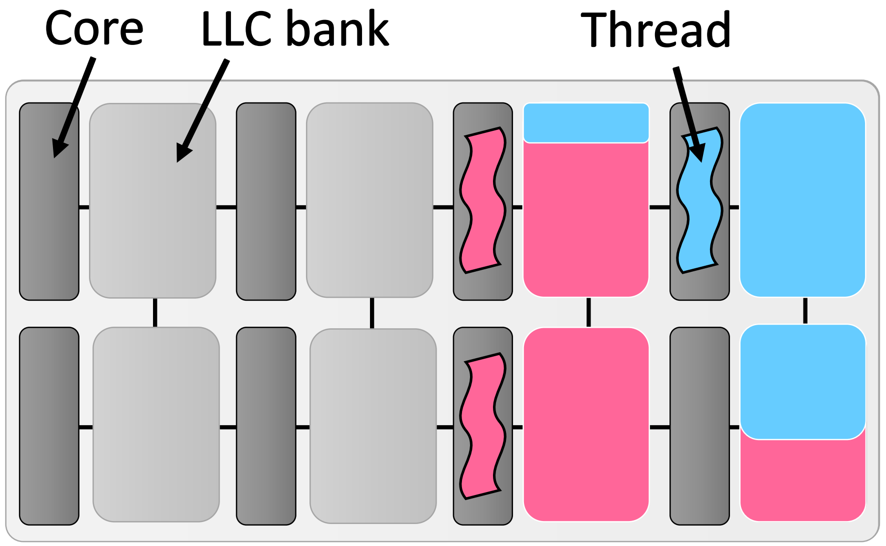
Spectre

Computer systems must be redesigned to efficiently enforce these goals!



Prior D-NUCAs do not work for datacenter applications!

- D-NUCAs improve energy efficiency > 50%!
- **Problem**: After 20 years of research, all D-NUCAs care only about data movement, making them unreliable for datacenter apps



8-core chip multiprocessor (CMP)

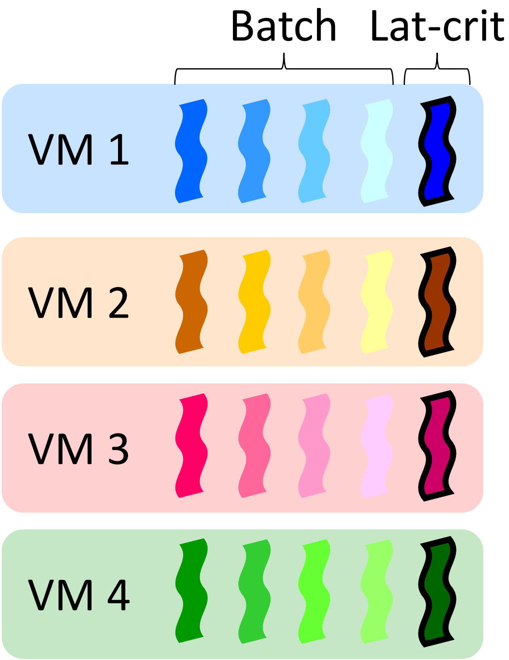
Dynamic NUCA (D-NUCA)
Static NUCA (S-NUCA)

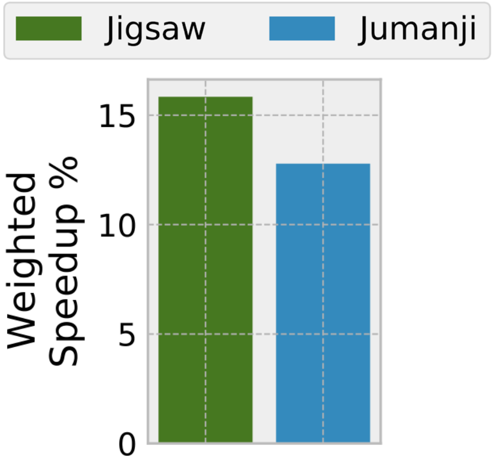
Jumanji is a new D-NUCA that improves security, tail latency, ***and*** energy efficiency!

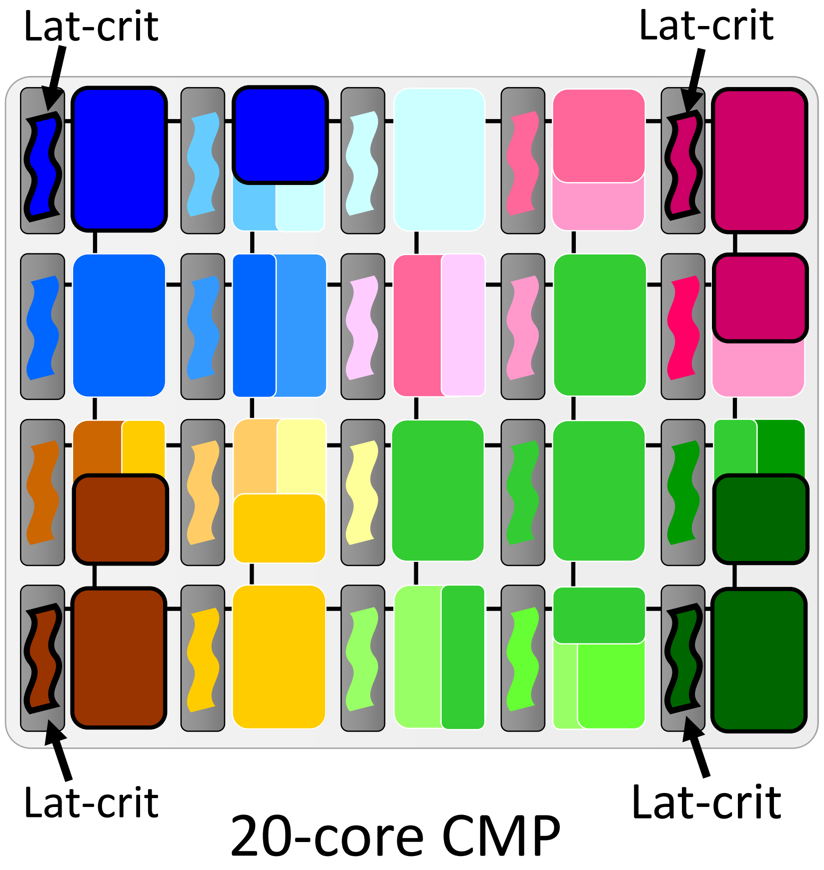
High-level overview of Jumanji

Jumanji: places apps' data in the LLC to meet apps' high-level goals

1. Reserves space to meet tail-latency deadlines
2. Isolates VMs across banks to defend against LLC attacks
3. Optimizes batch data placement within each VM





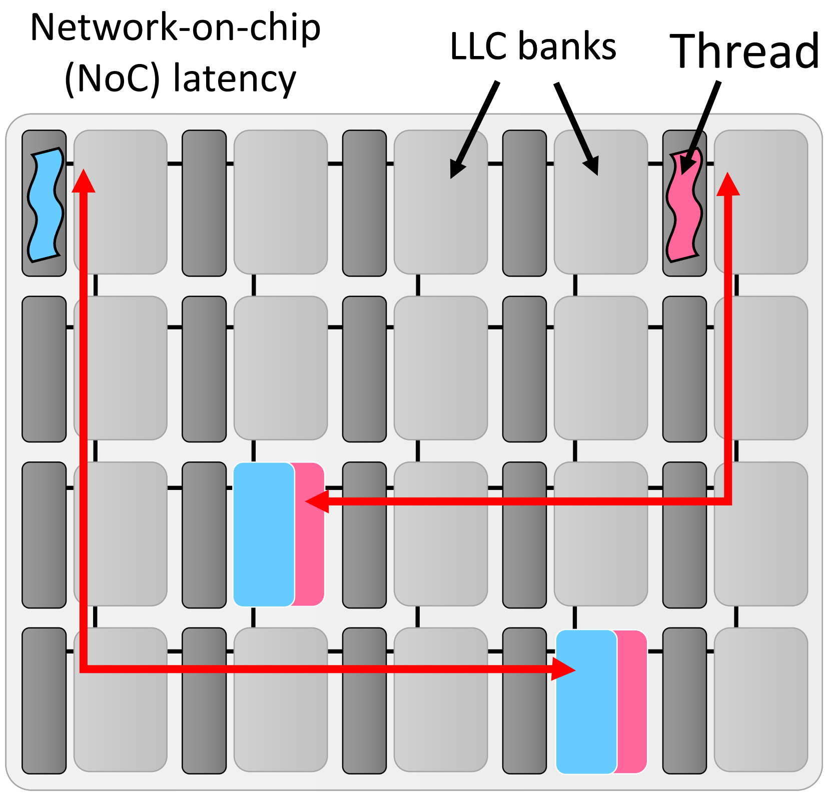


Agenda

- Motivation
  - Security
  - Tail latency
  - Prior D-NUCAs
- Jumanji's design
- Evaluation
- Conclusion

LLC has a big impact on security and tail latency

- Data movement within the LLC **exposes side-channel attacks** and **determines tail latencies**
- Many recently discovered side channels occur at the LLC [1]
- Larger LLC allocations greatly reduce tail latency [2,3,4]
- **These works ignore NUCA**, and by doing so, miss additional security vulnerabilities and harm efficiency



20-core CMP
w/ NUCA

[1] Liu et al., LLC Side-Channel Attacks, S&P 2015
[2] Chen et al., PARTIES, ASPLOS 2019
[3] Kasture et al., Ubik, ASPLOS 2014
[4] Lo et al., Heracles, ISCA 2015

Prior work

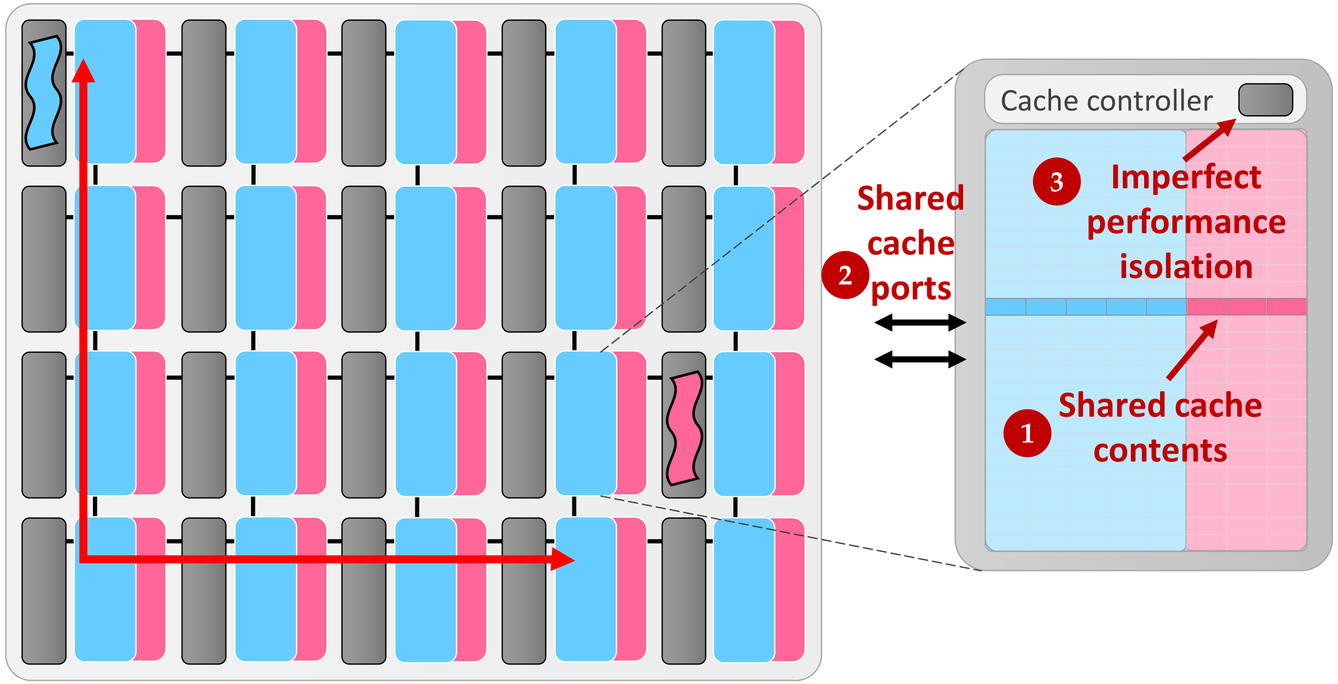
Security: sharing LLC banks is unsafe

Tail latency: ignoring NUCA wastes cache space

D-NUCA: ignoring application goals is harmful

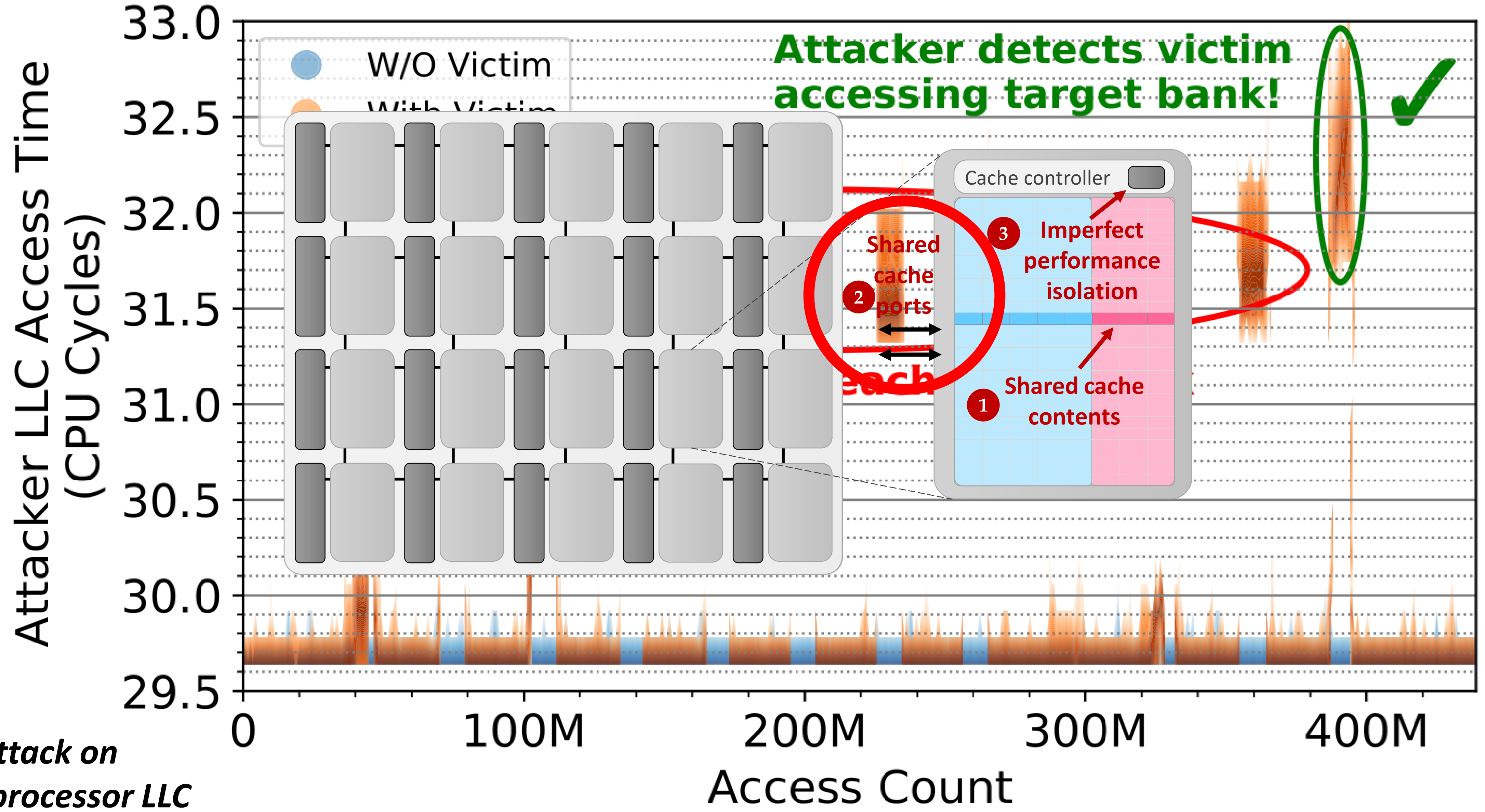
Prior LLC designs are **insecure** and **inefficient**

- Prior LLC designs focus on defending conflict attacks (e.g., prime + probe); **way-partitioning** is the most common defense
- **Insecure:** Limited LLC associativity prevents defending all processes
- **Insecure:** We demonstrate new *port* and *replacement-policy* attacks on prior designs
- **Wasteful:** Ignoring NUCA ➔ lots of unnecessary data movement



20-core CMP

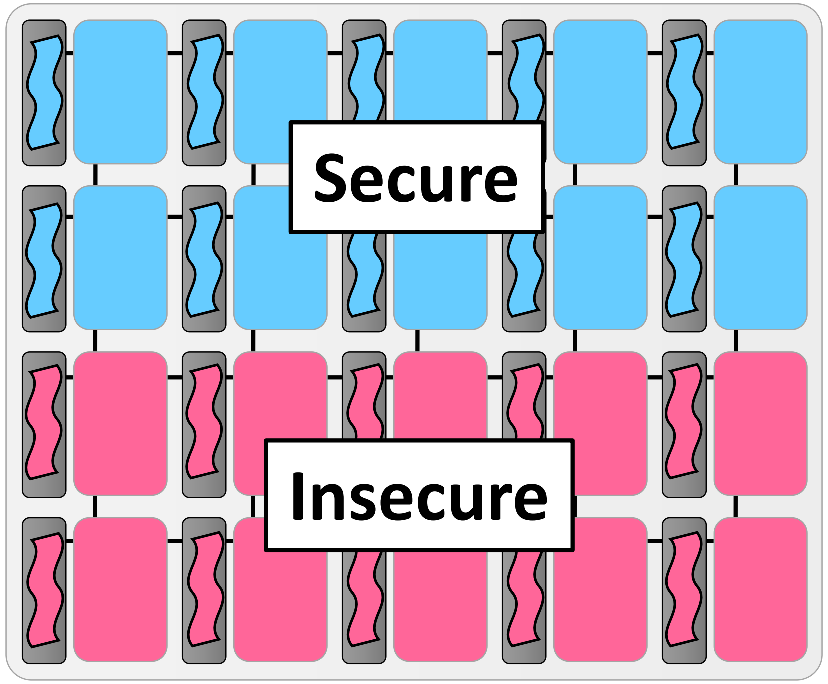
Demonstration: Sharing LLC banks is unsafe



Port attack on Xeon processor LLC

Only prior LLC defense: IRONHIDE

- IRONHIDE is the only prior solution which defends all LLC attacks
- It isolates applications across LLC banks, creating two regions: *secure* and *insecure*
- Although it also defends non-LLC attacks, it only supports one secure region at a time
- Additionally, IRONHIDE **does not address tail latency** and **does not minimize data movement** as well as D-NUCAs



20-core CMP

Omar et al., IRONHIDE, HPCA 2020

Prior work

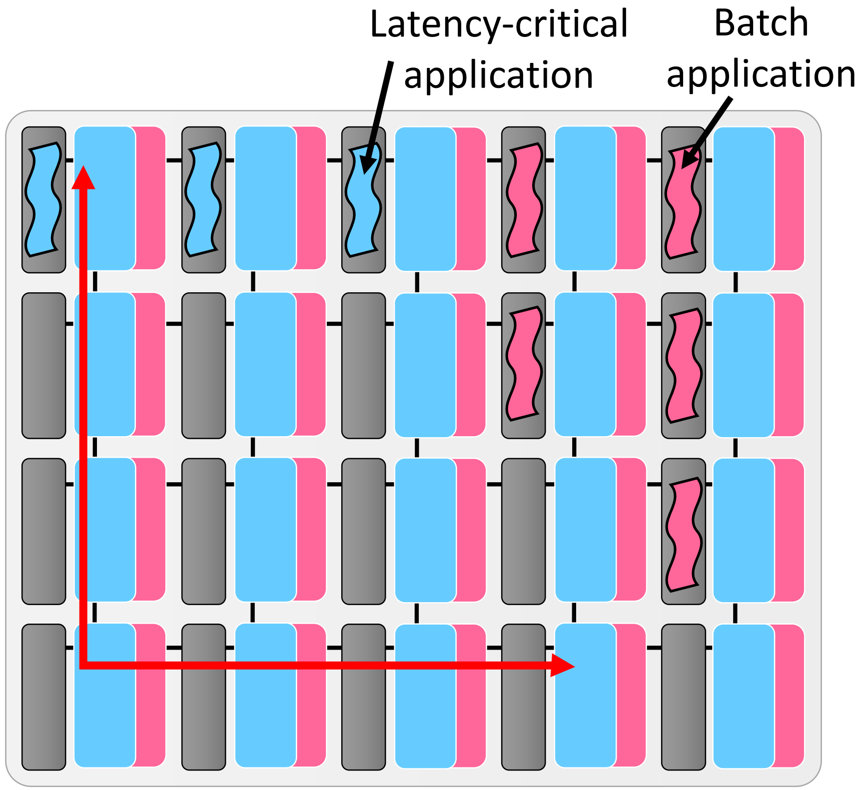
Security: sharing LLC banks is unsafe

Tail latency: ignoring NUCA wastes cache space

D-NUCA: ignoring application goals is harmful

Prior LLC designs for **tail latency** are **inefficient**

- Prior LLC designs for tail latency dynamically allocate cache space, but ignore NUCA
- **Wasteful:** Ignoring NUCA ➔ lots of unnecessary data movement for latency-critical applications
- **Wasteful:** Latency-critical applications thus need more cache space to meet deadlines, *harming co-running batch applications*
- (And are still insecure)



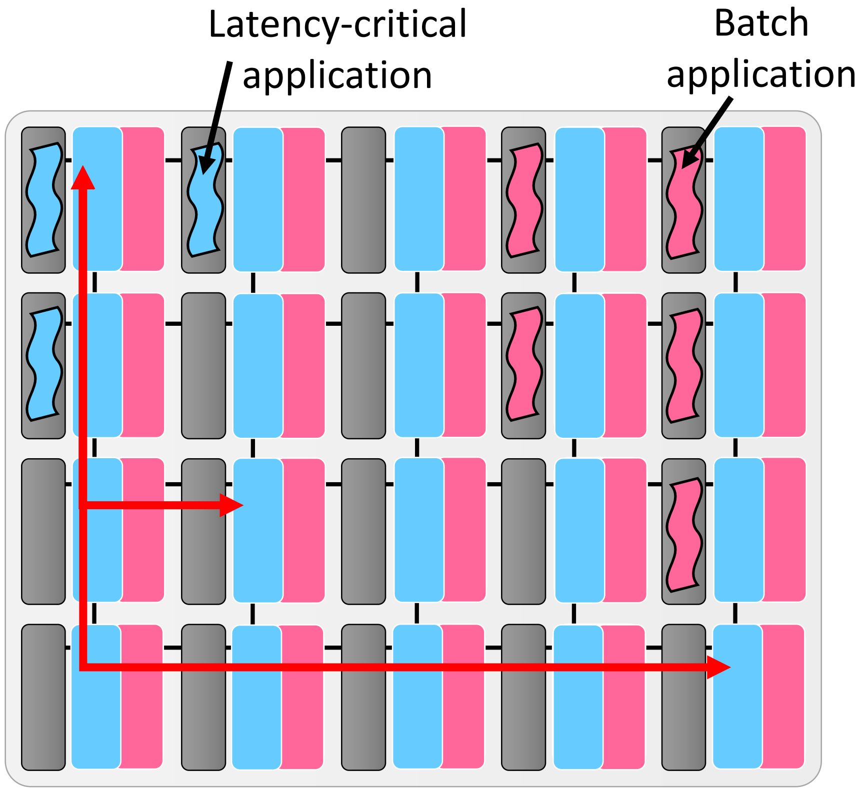
20-core CMP

Prior LLC designs for **tail latency** are **inefficient**

With S-NUCA, LLC accesses have high latency!

Placing data closer lowers avg access latency…

… so less space is needed to maintain tail latency



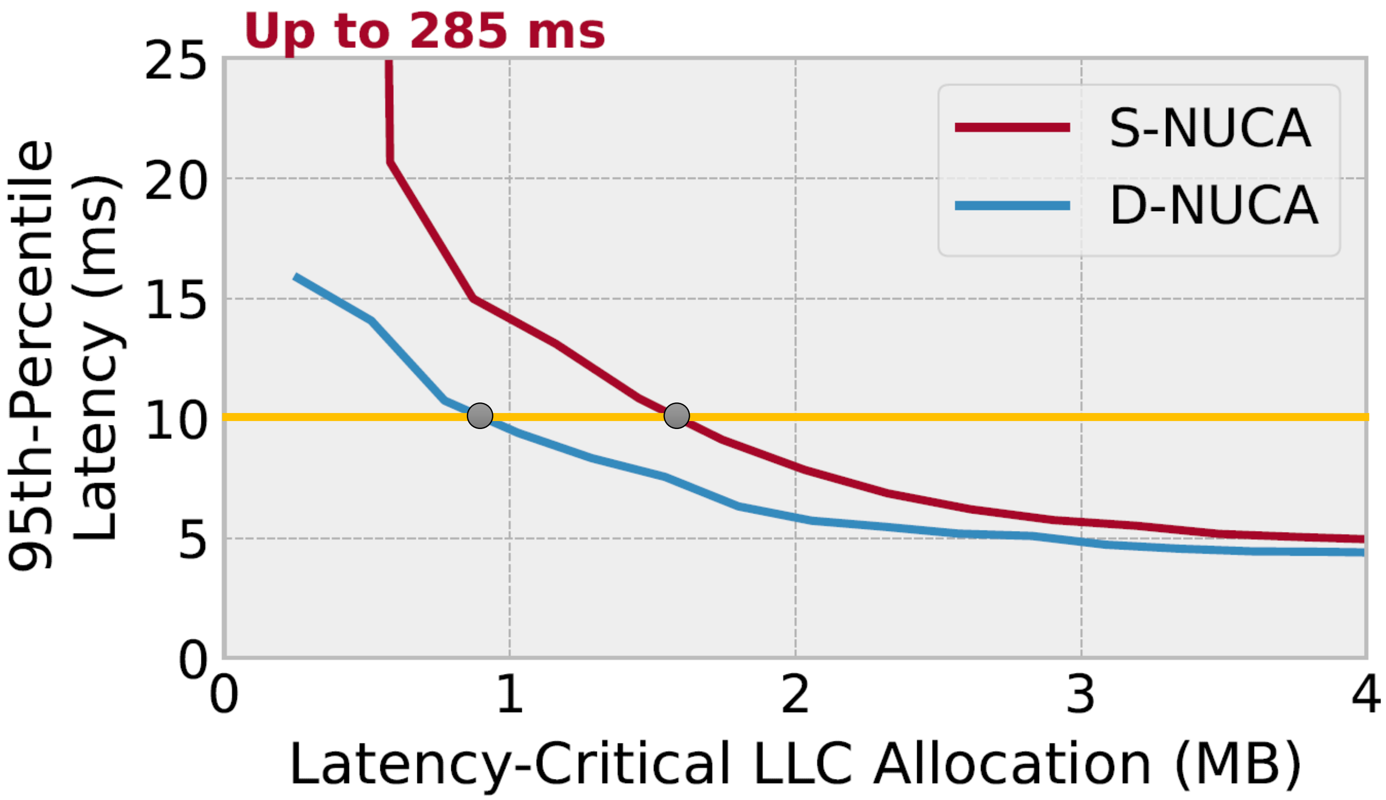
20-core CMP

Which leaves more space for co-running batch applications to improve throughput

D-NUCA meets tail-latency deadlines much more efficiently!

D-NUCA meets deadlines with less LLC space

- Simulated 20-core CMP
- Running latency-critical application *Xapian* in isolation
- Measured tail latency with different allocation sizes using way-partitioning (S-NUCA) and nearby data placement (D-NUCA)



S-NUCA requires 75% more space!

Prior work

Security: sharing LLC banks is unsafe

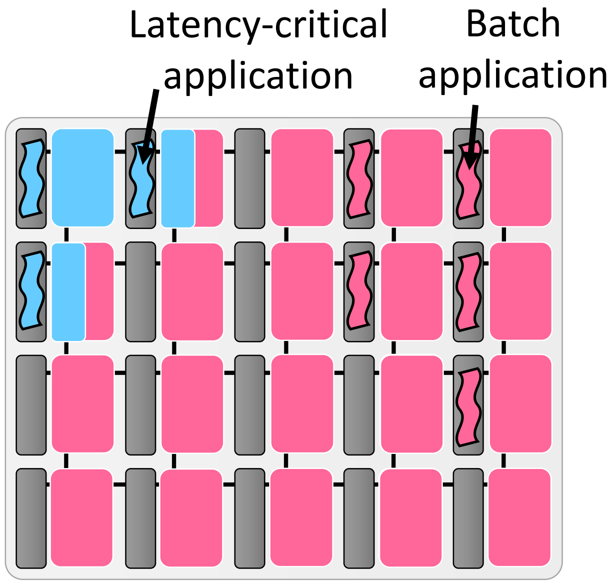
Tail latency: ignoring NUCA wastes cache space

D-NUCA: ignoring application goals is harmful

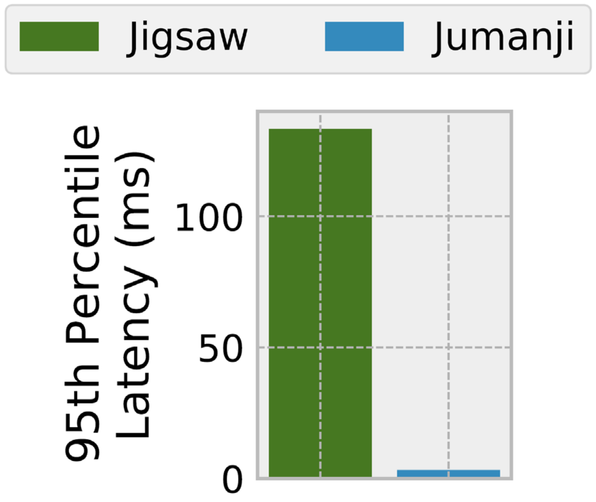
Unfortunately, prior D-NUCAs fail in the datacenter

- Dynamic non-uniform cache access (D-NUCA) architectures place data in LLC banks to **minimize data movement**
- *… but data movement != security and tail latency*

➔ Jigsaw performs well on throughput-oriented batch applications, but poorly for all other goals



Jigsaw: State-of-the-art D-NUCA



Tail latency for Jigsaw vs Jumanji

Beckmann et al., Jigsaw, PACT 2013

Jumanji is the solution!

Defends all applications against more LLC attacks

Meets tail-latency deadlines with minimal data movement

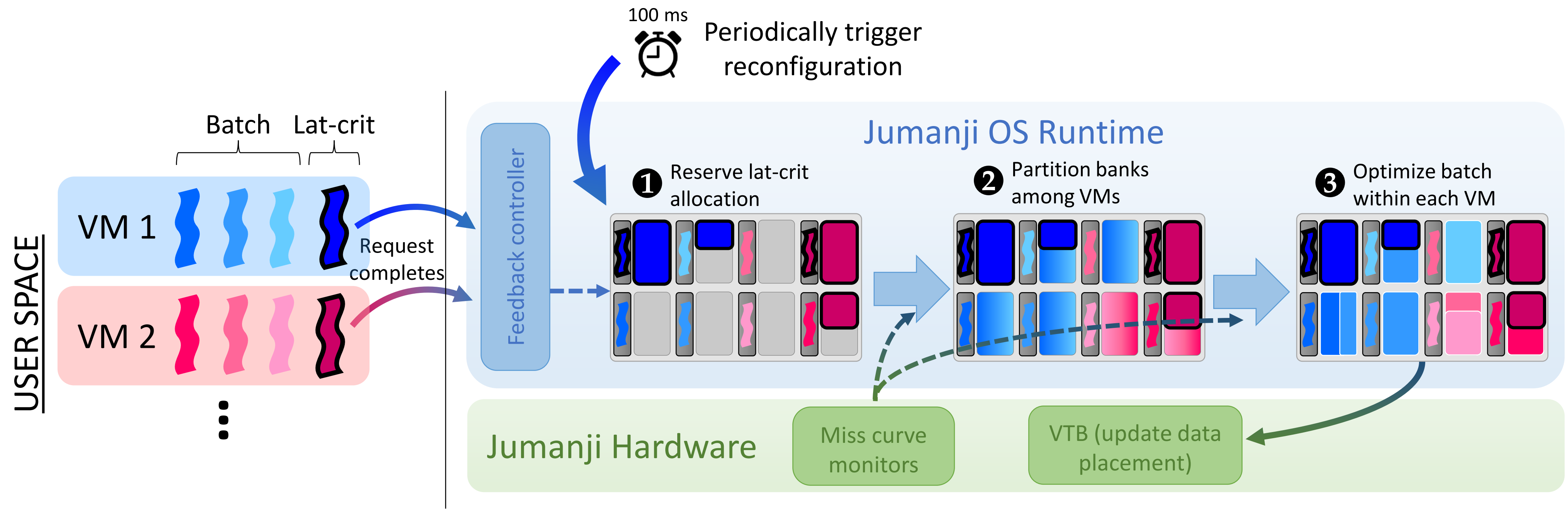
Maximizes throughput of co-running batch applications

Simple design and small software changes to Jigsaw

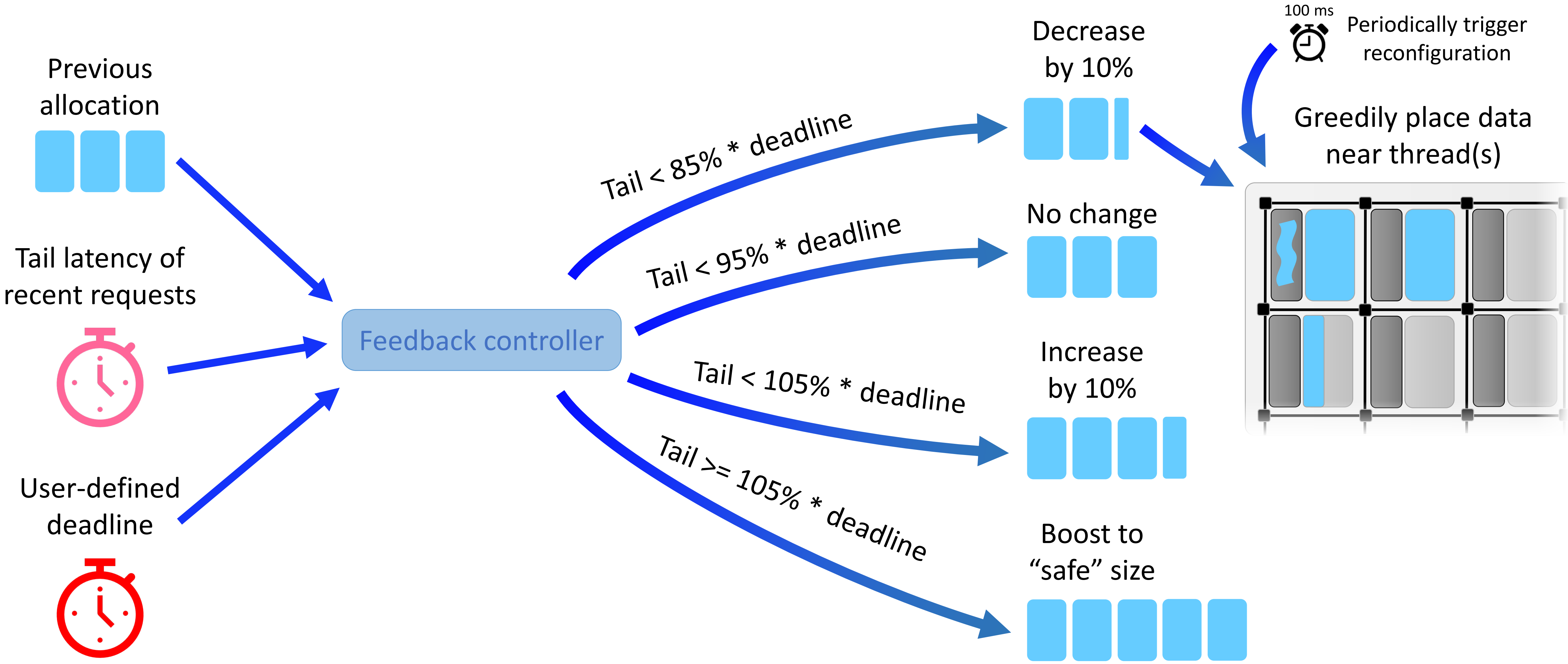
Agenda

- Motivation
  - Security
  - Tail latency
  - Prior D-NUCAs
- **Jumanji's design**
- Evaluation
- Conclusion

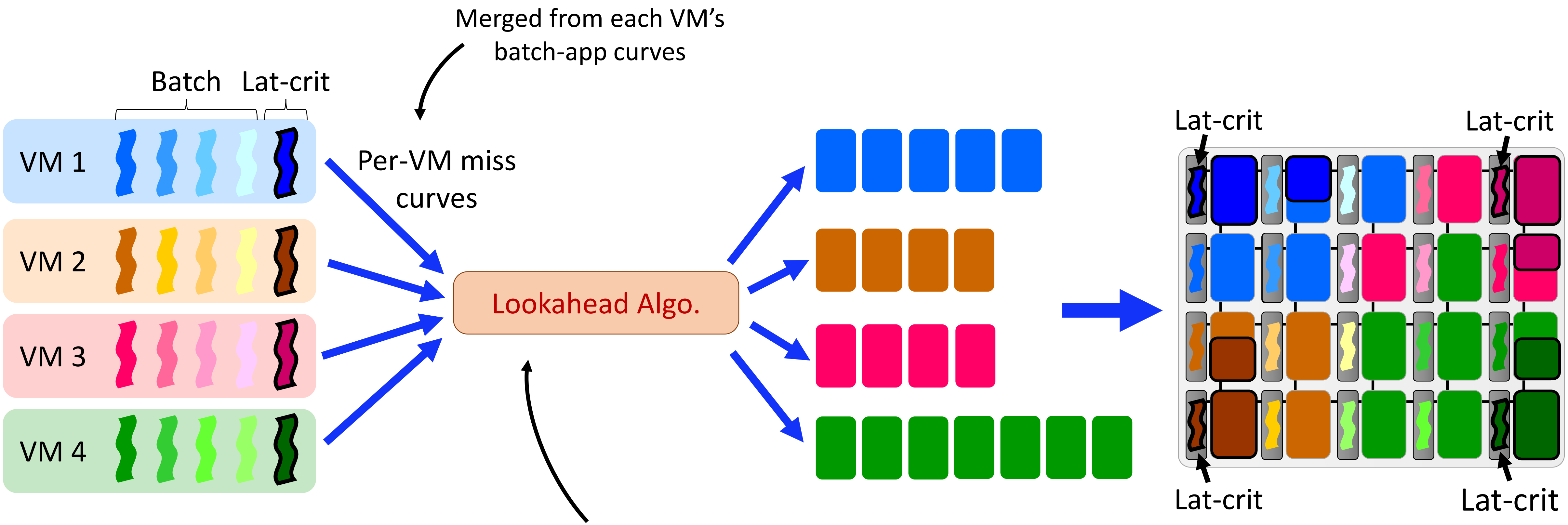
Jumanji Software



Step 1: Meeting tail-latency deadlines

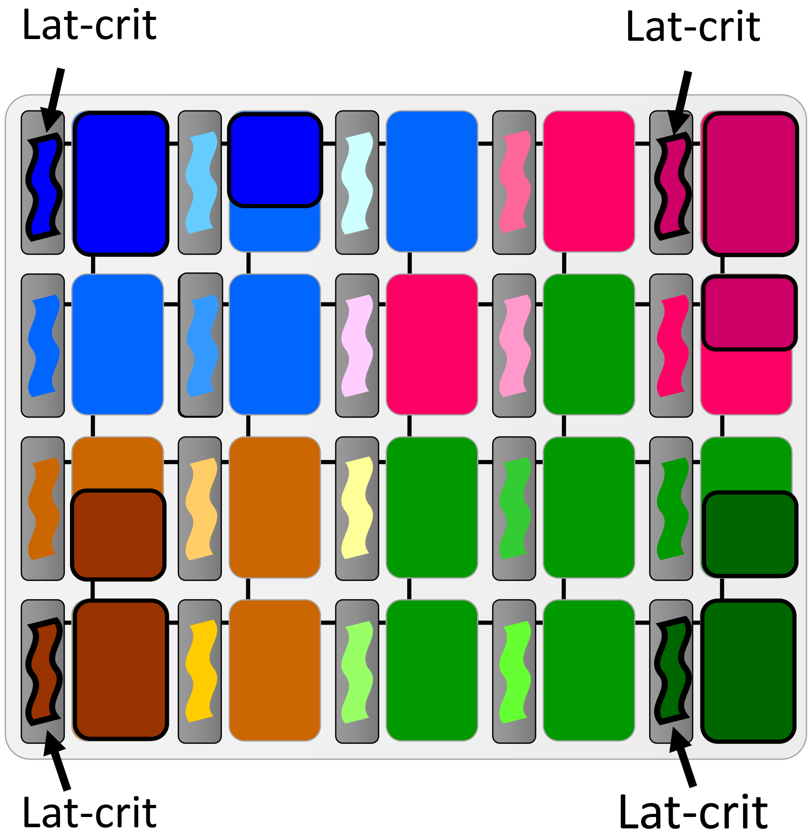


Step 2: Defending LLC attacks

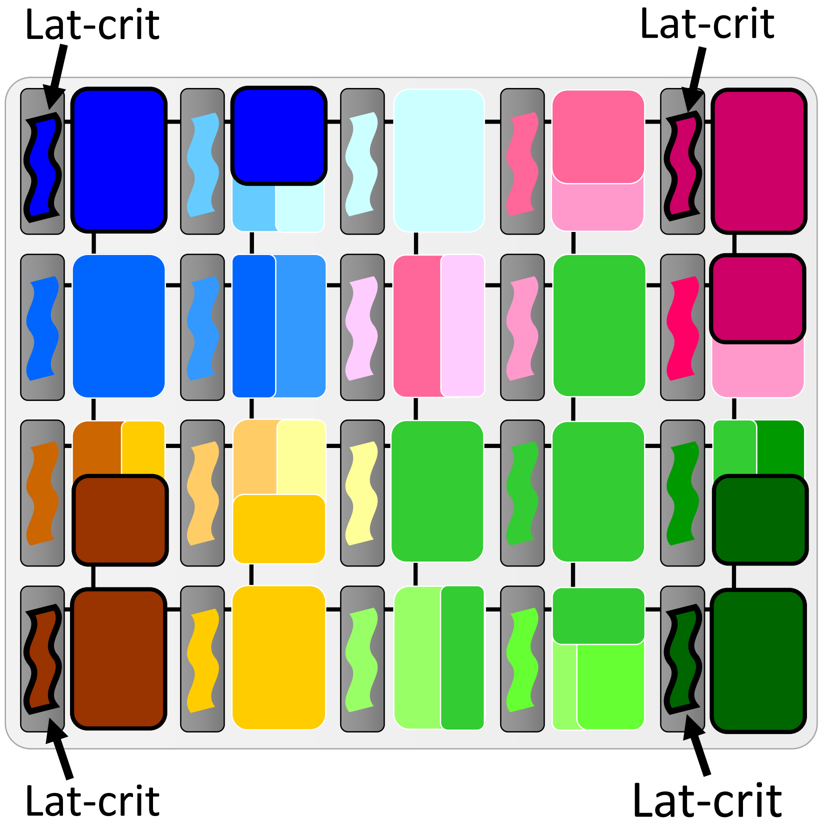


Modified to allocate at bank granularity to prevent VMs from sharing banks. **This is how Jumanji defends LLC bank attacks.**

Step 3: Optimizing for batch performance



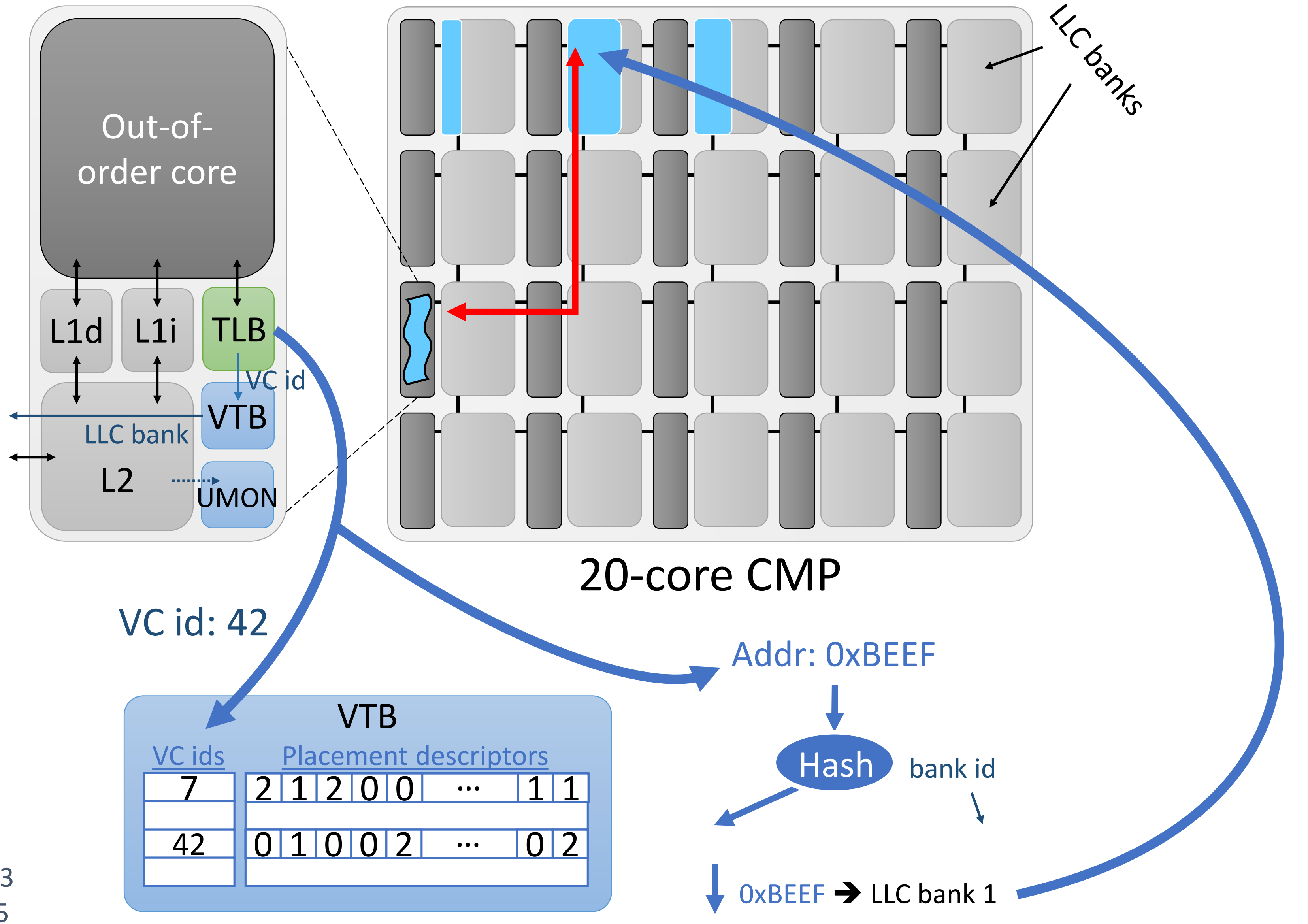
Execute Jigsaw on batch apps within each VM



Jumanji Hardware (borrowed from Jigsaw)

Modified: TLB

Added: VTB, UMON



Beckmann et al., Jigsaw, PACT 2013

Beckmann et al., CDCS, HPCA 2015

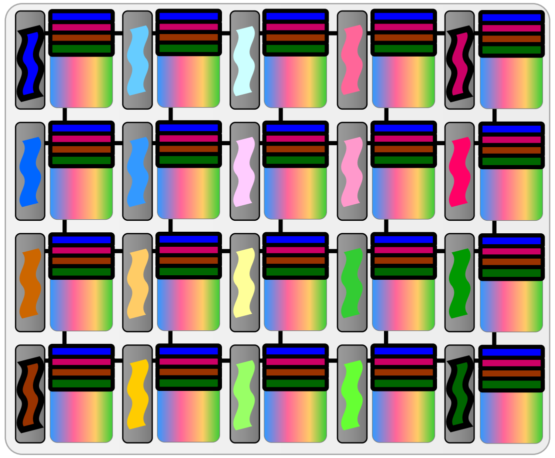
Agenda

- Motivation
  - Security
  - Tail latency
  - Prior D-NUCAs
- Jumanji’s design
- **Evaluation**
- Conclusion

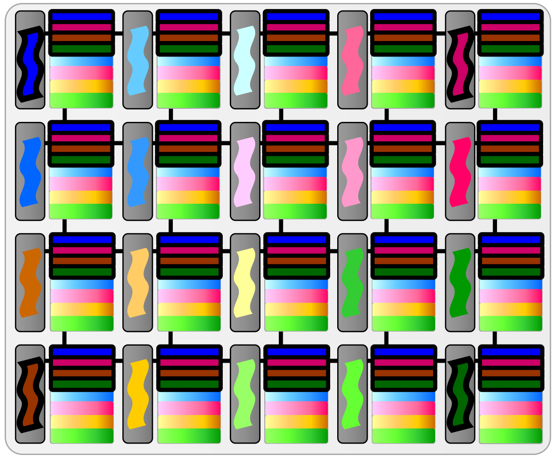
Evaluation Methodology

- Simulate 20-core CMP based on Nehalem using ZSim
- 20 single-threaded applications split into 4 VMs
- Each VM has
  - 1 latency-critical app (from Tailbench)
  - 4 batch apps (from SPEC CPU2006)
- Latency-critical workloads
  - 4 copies of the same latency-critical app
  - Random mixes of latency-critical apps
- Batch workloads
  - 40 random mixes of batch apps for each latency-critical workload

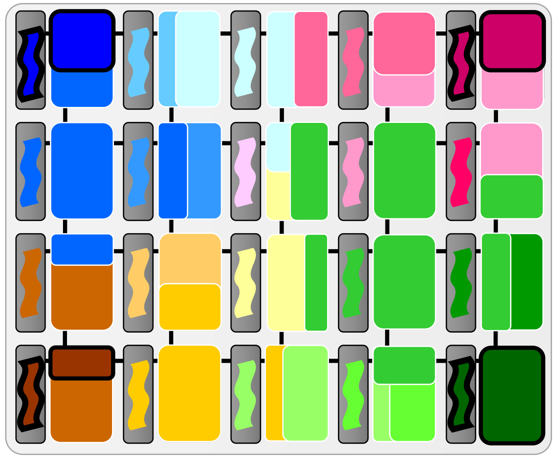
Evaluation Methodology – LLC Designs



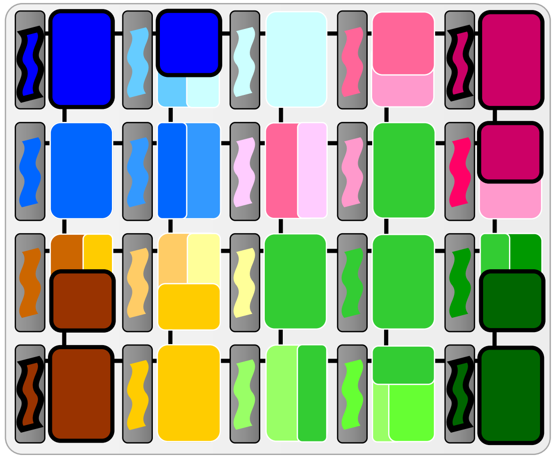
Adaptive:
Meets deadlines using way-partitioning and feedback controller



VM-Part:
Also way-partitions across VMs to defend conflict attacks (but not our new attacks)

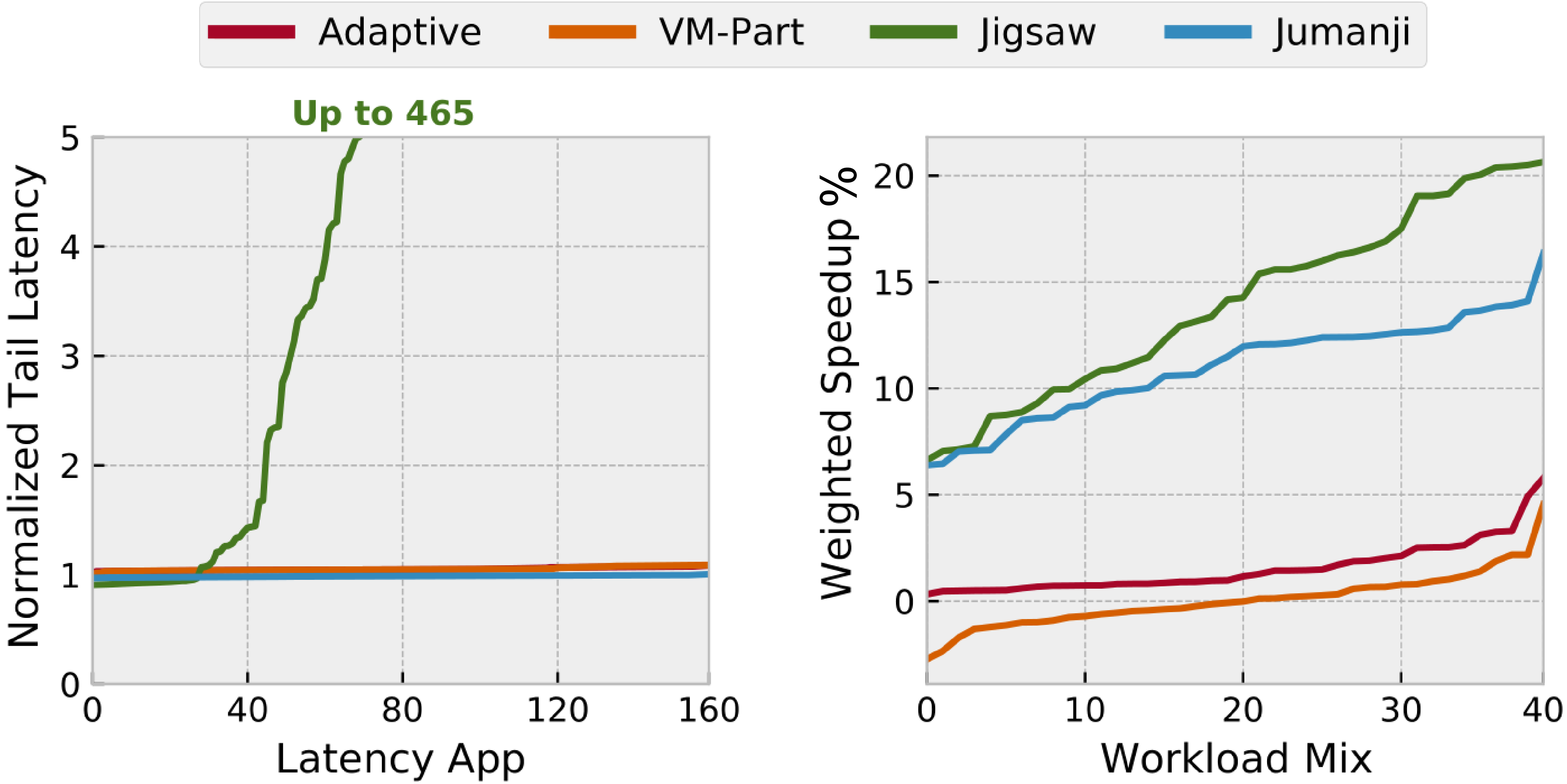


Jigsaw



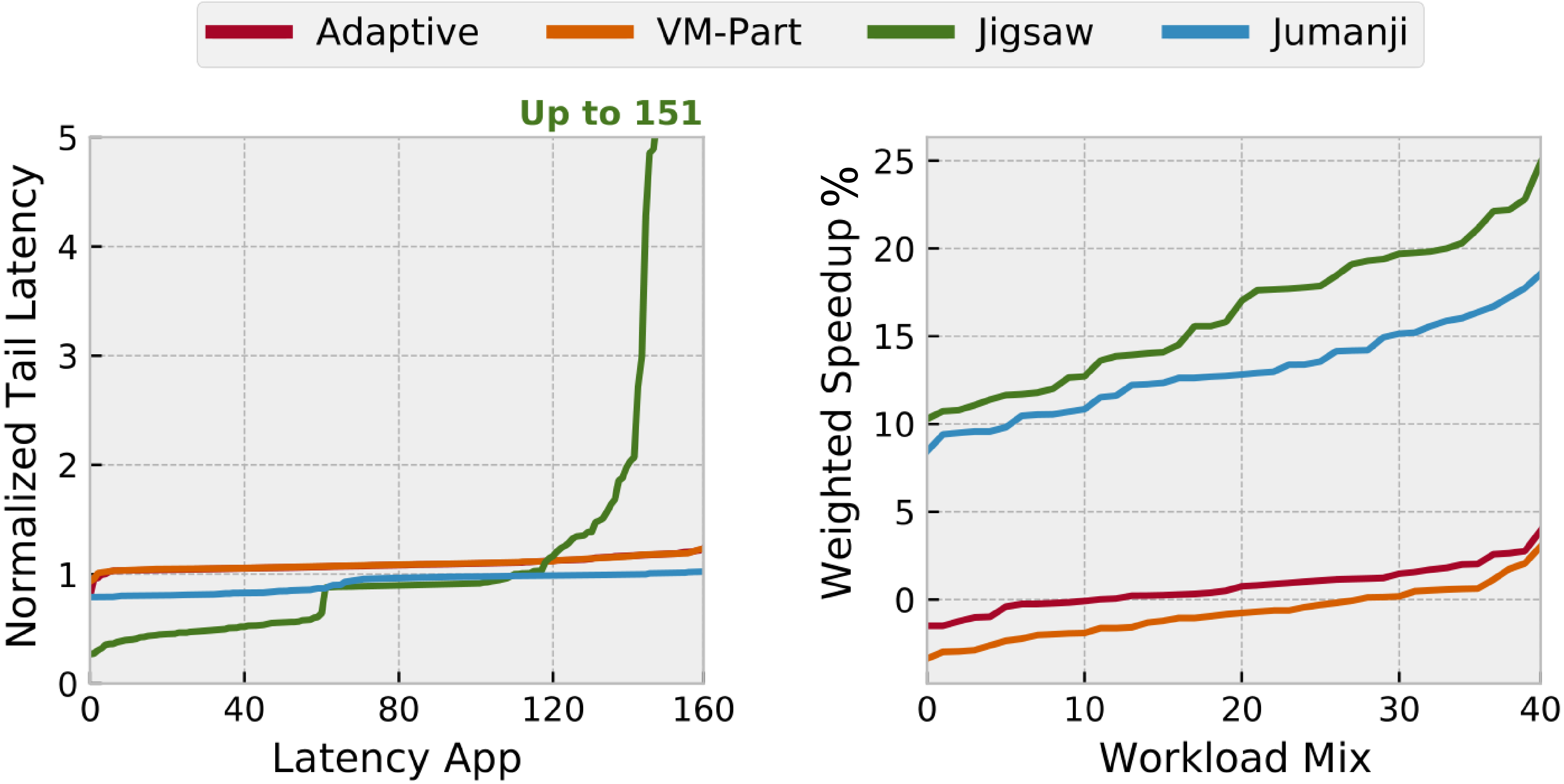
Jumanji

Jumanji meets deadlines and speeds up batch apps



Latency-critical apps: Xapian x 4

Jumanji meets deadlines and speeds up batch apps



Latency-critical apps: random mixes from Tailbench

See the paper for more results!

- Jumanji's data placement is nearly ideal
- Jumanji scales well with VM size

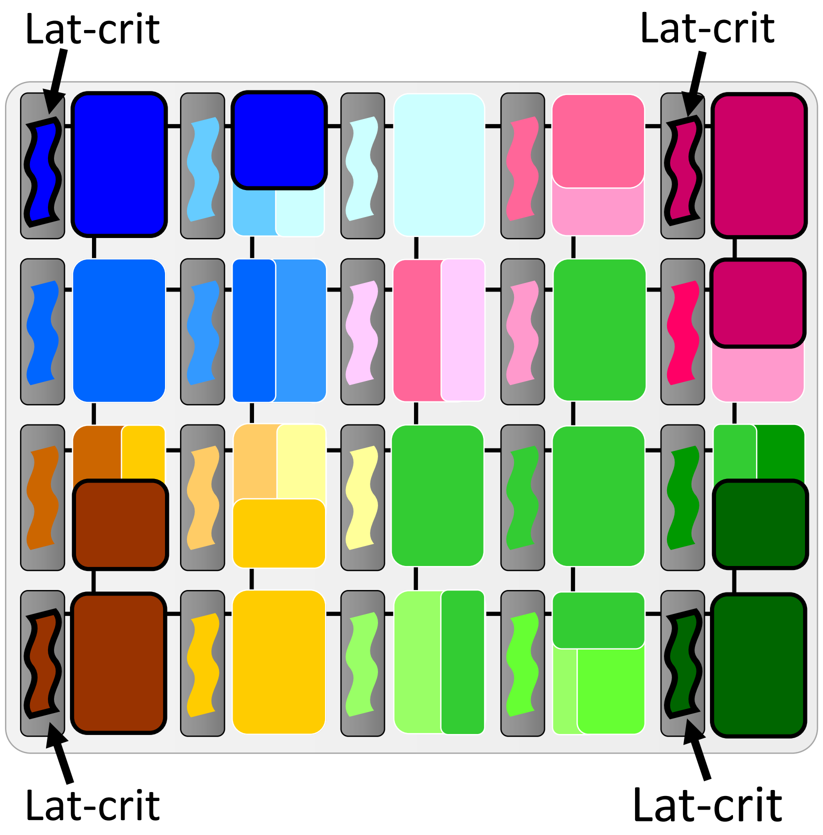
- Also…
  - Energy savings
  - Security analysis
  - System sensitivity study

Agenda

- Motivation
  - Security
  - Tail latency
  - Prior D-NUCAs
- Jumanji's design
- Evaluation
- **Conclusion**

Jumanji makes D-NUCA viable in the datacenter

- Jumanji recognizes the advantages D-NUCA provides for security and tail latency
- Isolating untrusted VMs across LLC banks provides stronger security than prior designs
- Placing latency-critical data near cores saves cache space for co-running batch applications
- The overall design makes D-NUCA work for modern application goals



Brian Schwedock *bschwedo@andrew.cmu.edu*
Nathan Beckmann *beckmann@cs.cmu.edu*