EE 691: R&D Project

# Secure LLC

Presenter: Rishabh Ravi

Roll No: 200260041

Supervisor: Prof. Virendra Singh


Computer Architecture & Dependable Systems Laboratory,

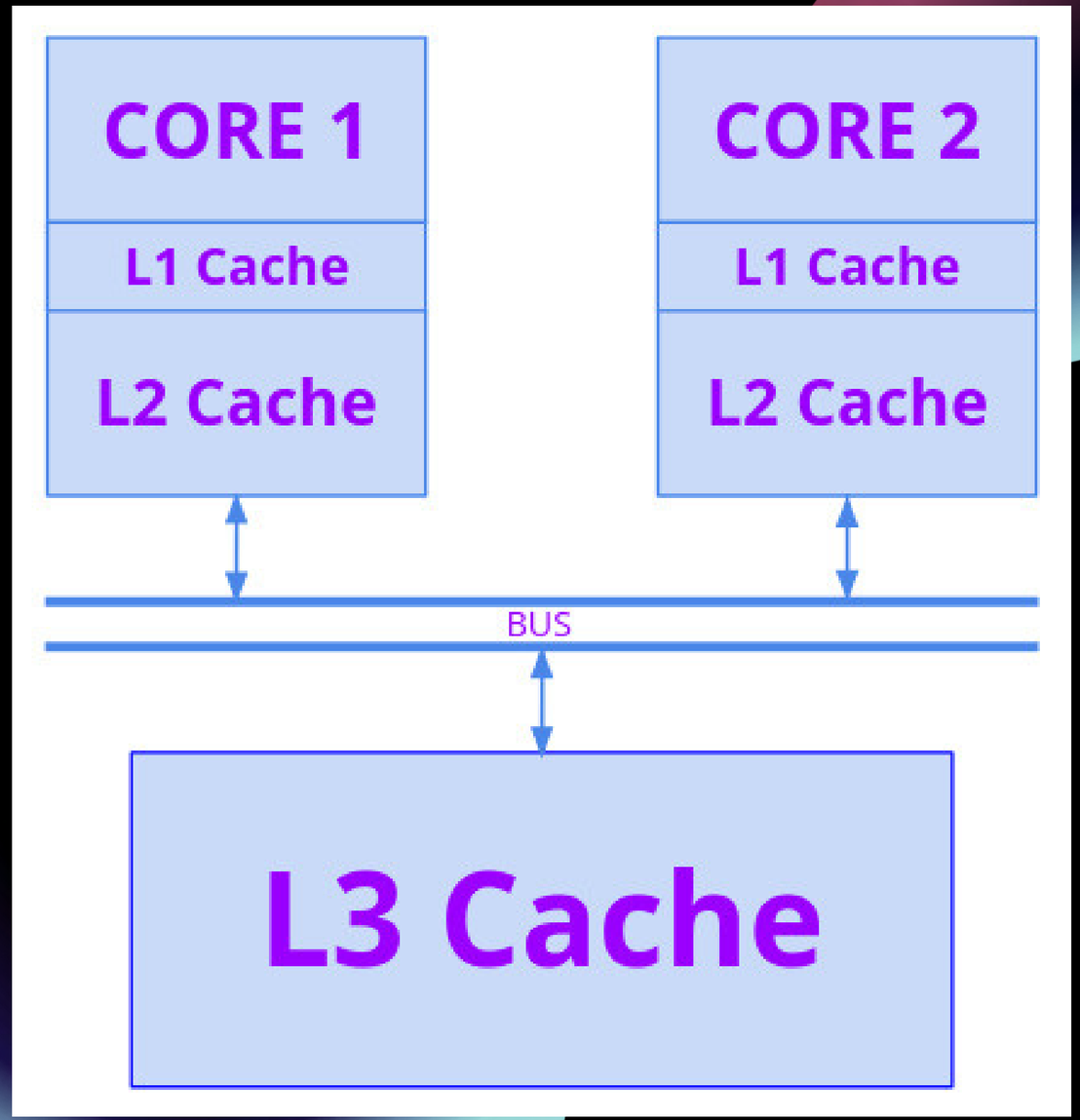
Department of Electrical Engineering, Indian Institute of Technology Bombay

# Outline

- Introduction
- Literature Survey
- Proposed Idea
- Simulation Results
- Conclusion
- Future Work
- Refernces

# Introduction

- The primary benefit of cache partitioning is that it provides protection from one core evicting another



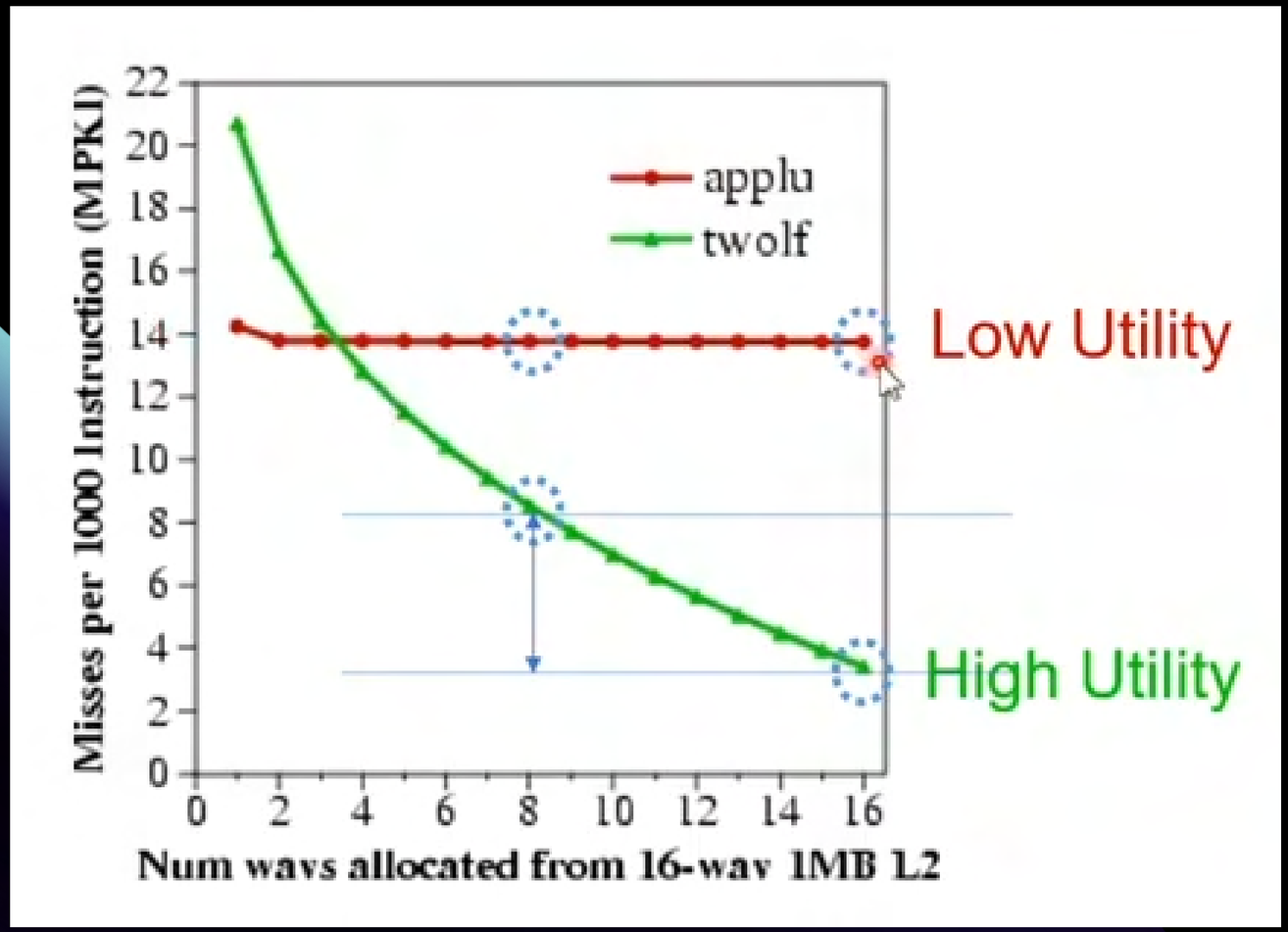
# Introduction

- Static Partitioning divides it equally between each core once and never again.
- UCP is an algorithm that dynamically partitions the cache based on their utility
- This creates side channels and the risk of hardware attacks

# Literature Survey

## UCP

A core with demand greater than another is given an extra line takenthe other core.

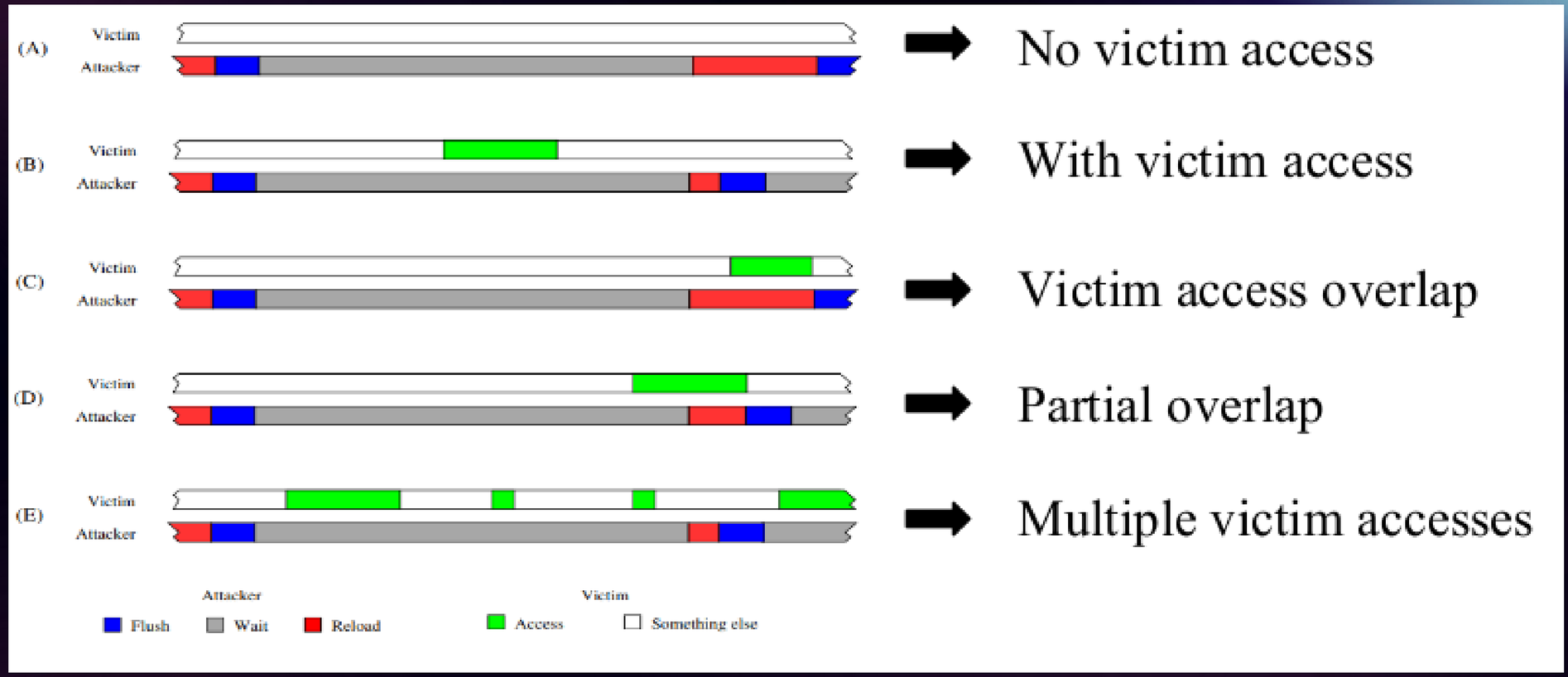



M. K. Qureshi and Y. N. Patt, "Utility-Based Cache Partitioning: A Low-Overhead, High-Performance, Runtime Mechanism to Partition Shared Caches," 2006 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'06), 2006, pp. 423-432, doi: 10.1109/MICRO.2006.49.

# UCP vs Static

- Utiltiy-based partitioning performs significantly better
- This however introduces the risk of hardware attacks
- A line reallocated from one core to another, provides information
- Several attacks have target the side channel so formed

# Flush + Reload




Y. Yarom and K. Falkner, “Flush+reload: A high resolution, low noise, l3 cache side-channel attack,” in Proceedings of the 23rd USENIX Conference on Security Symposium, SEC’14, (USA), p. 719–732, USENIX Association, 2014

# Prime + Probe

- The attacker fills cache sets with its own data
- It then waits for a time interval while the victim executes and utilizes the cache.
- The attacker continues execution and measures the time to load each set of his data

F. Liu, Y. Yarom, Q. Ge, G. Heiser, and R. B. Lee, “Last-level cache side-channel attacks are practical,” in 2015 IEEE Symposium on Security and Privacy, pp. 605–622, 2015.

# Mitigations

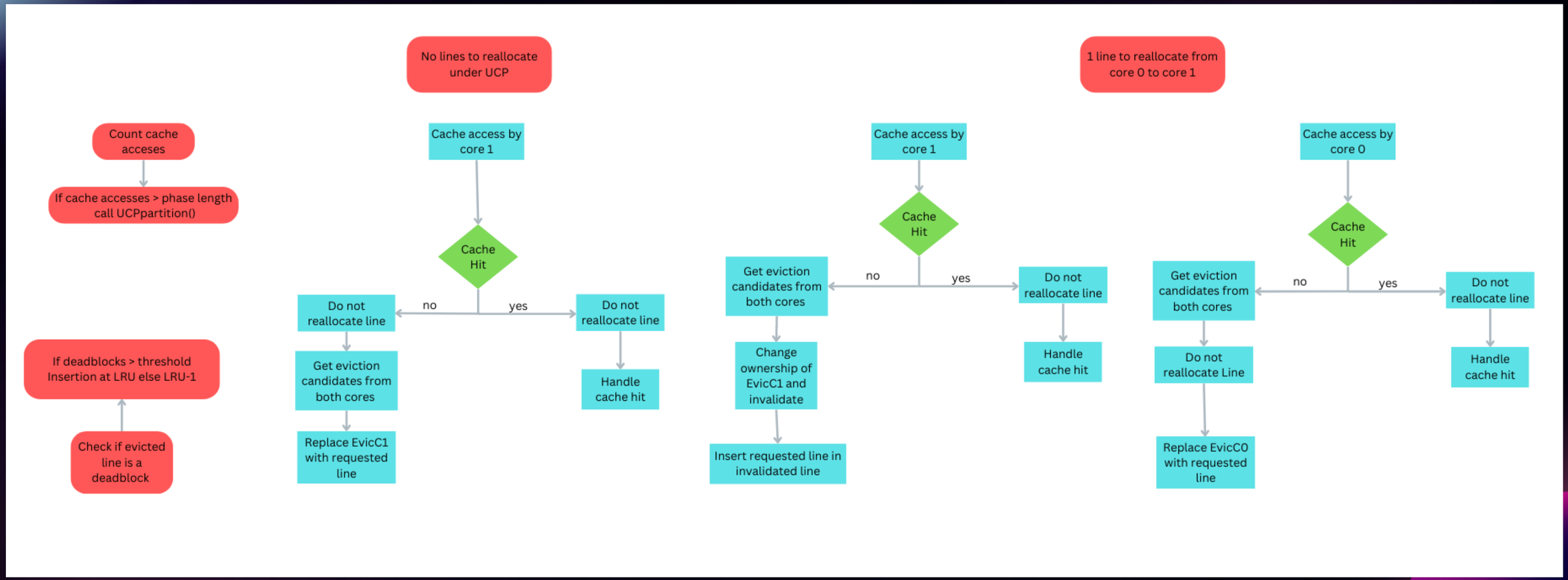
- These attacks which target the LLC side channel need to be mitigated
- One way is to revert back to static partitioning
- This however decreases performance
- Another could be to implement the PASS-P algorithm

# PASS-P

- It mitigates attacks where the attacker tries to analyze the memory accesses made by the victim
- PASS-P invalidates all cache lines that are reallocated from one process to another.

N. Boran, P. Joshi, and V. Singh, “Pass-p: Performance and security sensitive dynamic cache partitioning,” in Proceedings of the 19th International Conference on Security and Cryptography - Volume 1: SECRYPT,, pp. 443–450, INSTICC, SciTePress, 2022.

# Proposed Idea



# Proposed Idea

- Partitioning could be calculated independently for each set or done globally
- We refer independent partitioning to as local partitioning and the latter as global partitioning
- The former results in each set having a different partition

# Simulation Results

# Simulation Setup

- **Simulator**: Sniper Multicore Simulator
- **Baseline Processor**: x86 Nehalem microarchitecture, 2.67 GHz, 4-wide fetch,128-entry ROB
- **Main Memory Access Latency**: 175 cycles

# Simulation Setup

- **Memory Organization:** Private L1, L2 caches and shared L3 cache
- **LLC Size**: 4MB
- **Cores:** 2 core

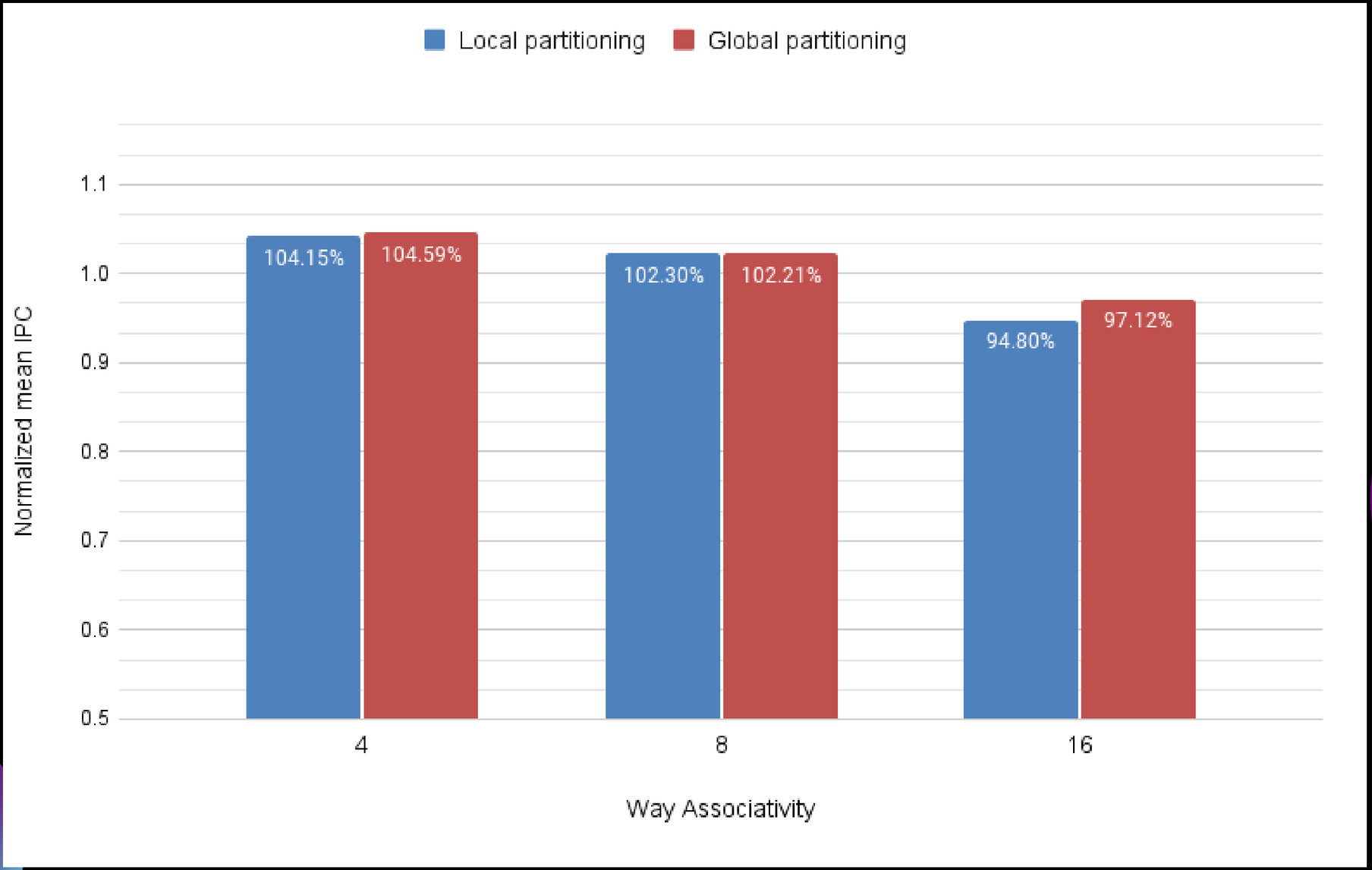
# Simulation Setup

- **Benchmarks:** 25 pairs of SPEC* 2006 benchmarks, categorized as memory-memory intensive (18) and memory-compute intensive(7)

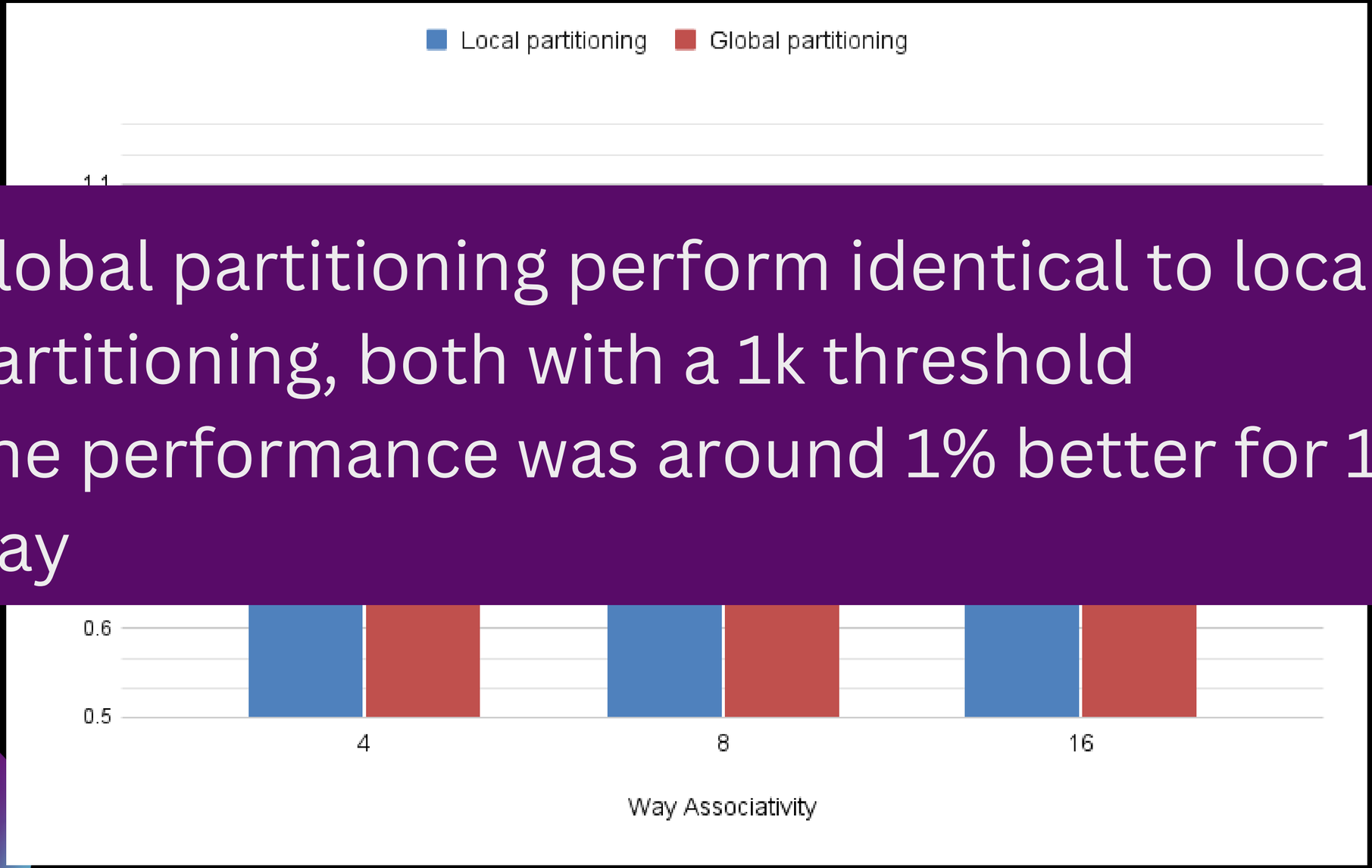
- IPC normalized with respect to static partitioning

John L. Henning. 2006. SPEC CPU2006 benchmark descriptions. SIGARCH Comput. Archit. News 34, 4 (September 2006), 1–17. https://doi.org/10.1145/1186736.1186737

# IPC comparison for different partitioning



# IPC comparison for different partitioning




- Global partitioning perform identical to local partitioning, both with a 1k threshold
- The performance was around 1% better for 16 way

# IPC comparison for different partitioning

- This could be due to
  - considerably larger computation associated with local partitioning
  - Each set is partitioned a lesser number of times
  - This leads to each set incorrectly modeling the utility

# IPC comparison for different set associativies

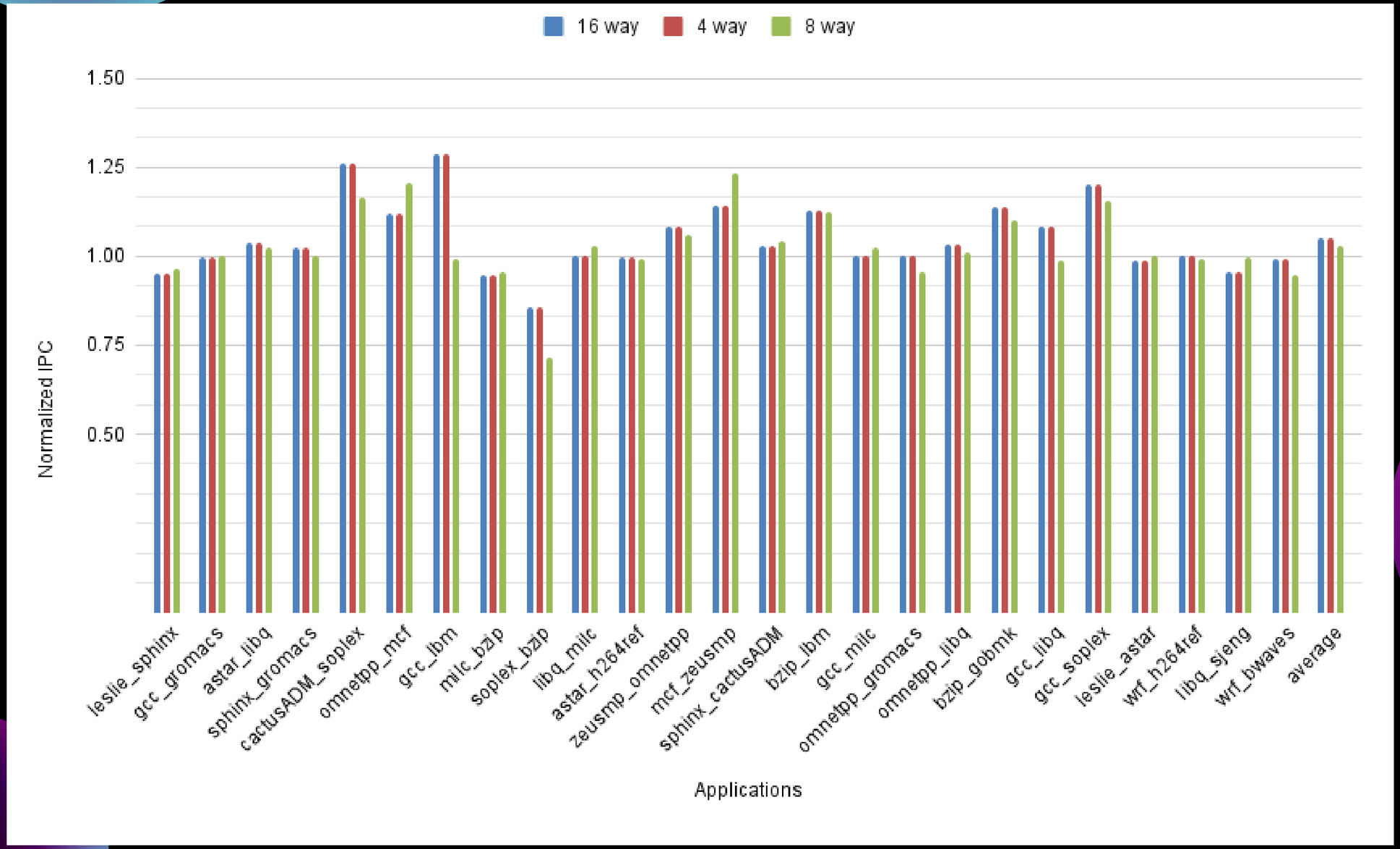
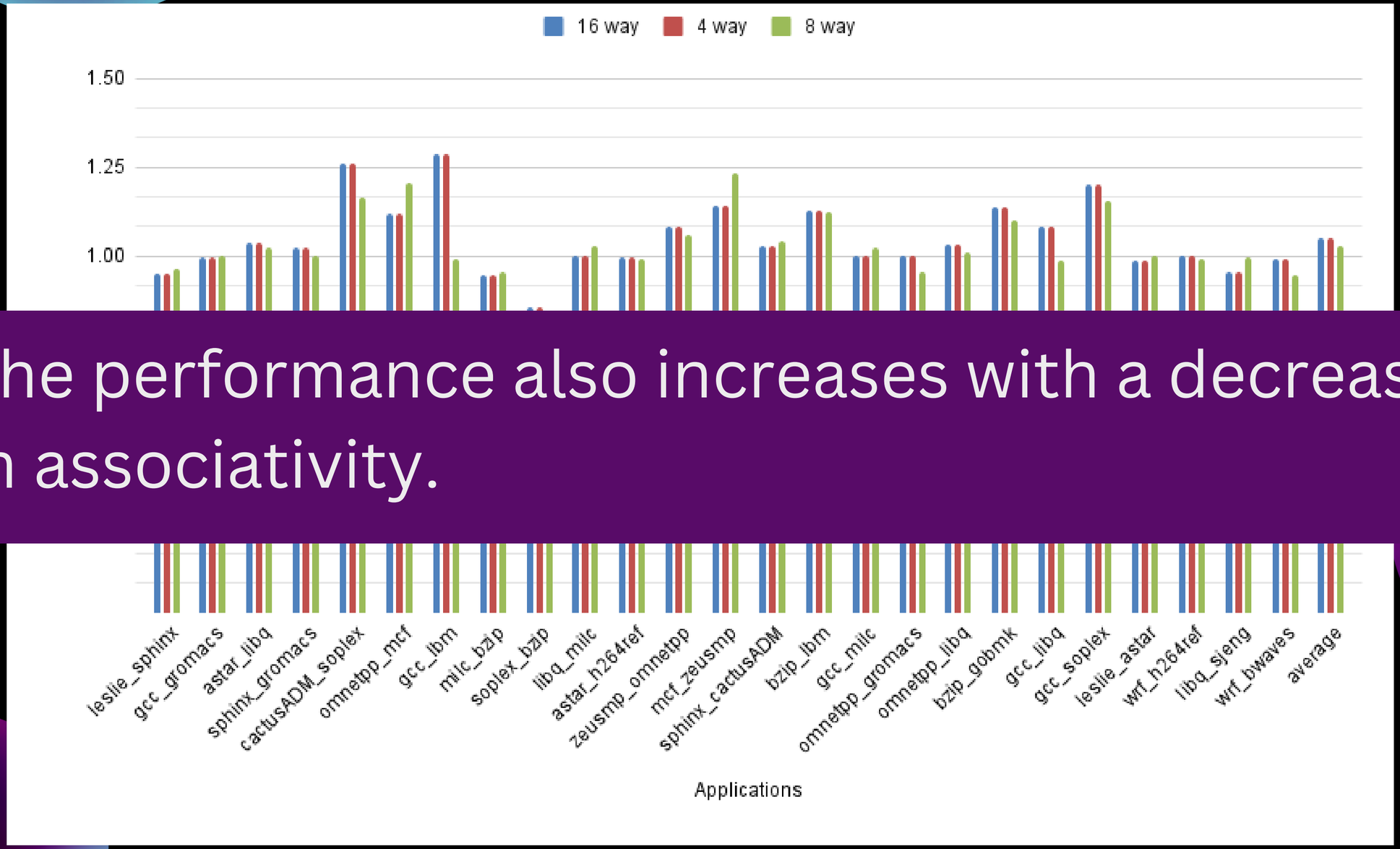



Fig: IPC comparison for **global** partitioning with different set associativites

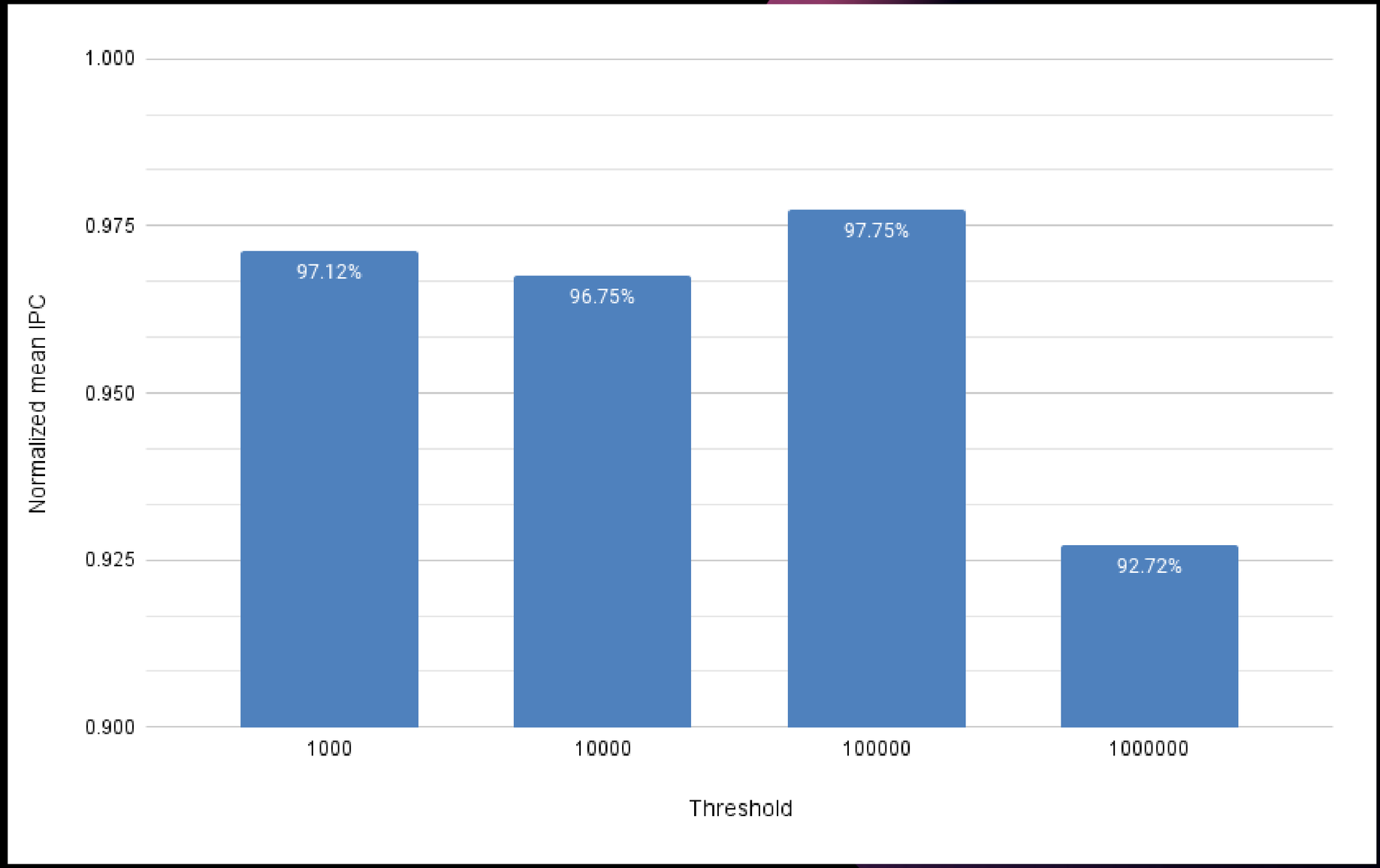
# IPC comparison for different set associativies



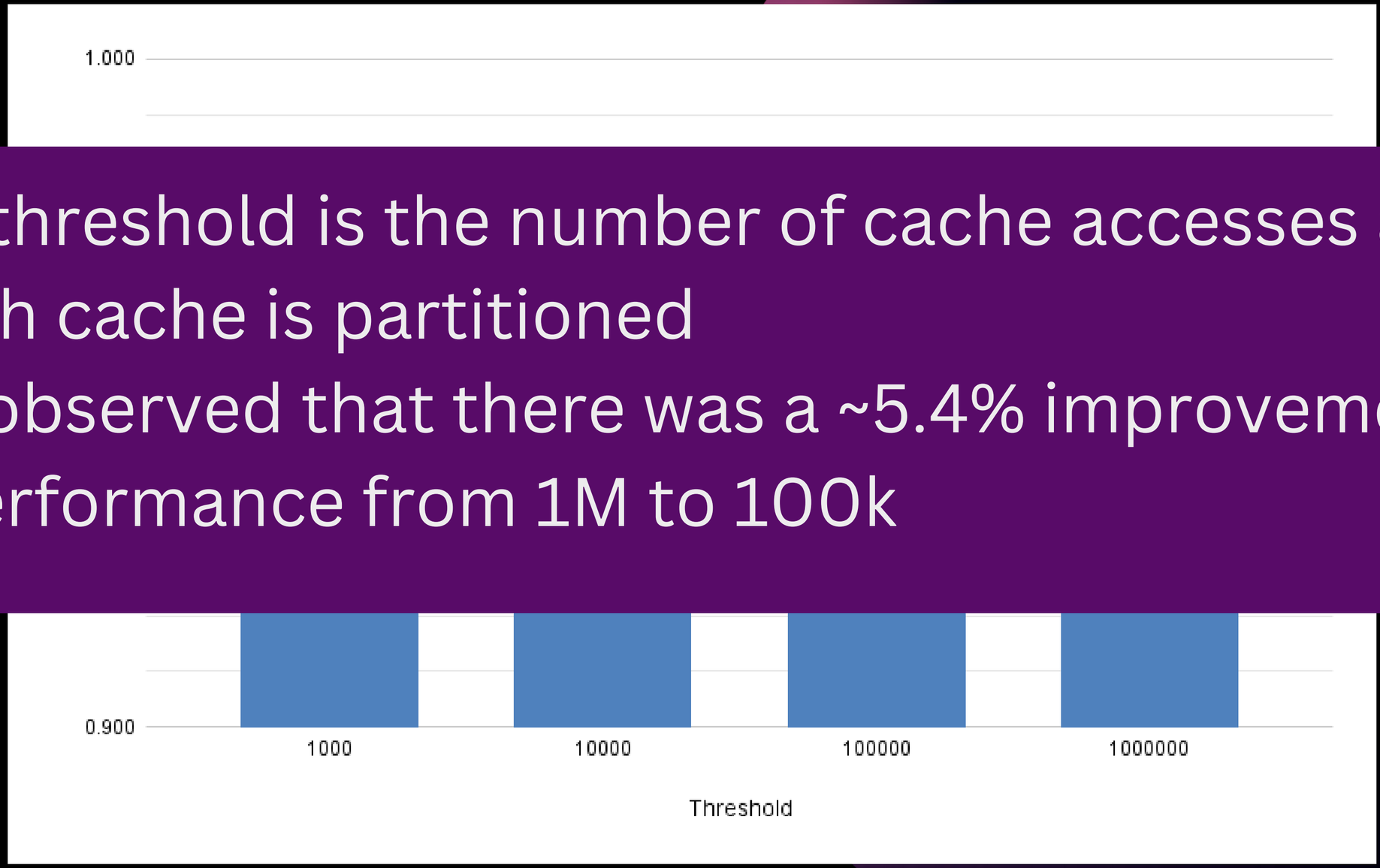

- The performance also increases with a decrease in associativity.

Fig: IPC comparison for **global** partitioning with different set associativites

# IPC comparison for different thresholds

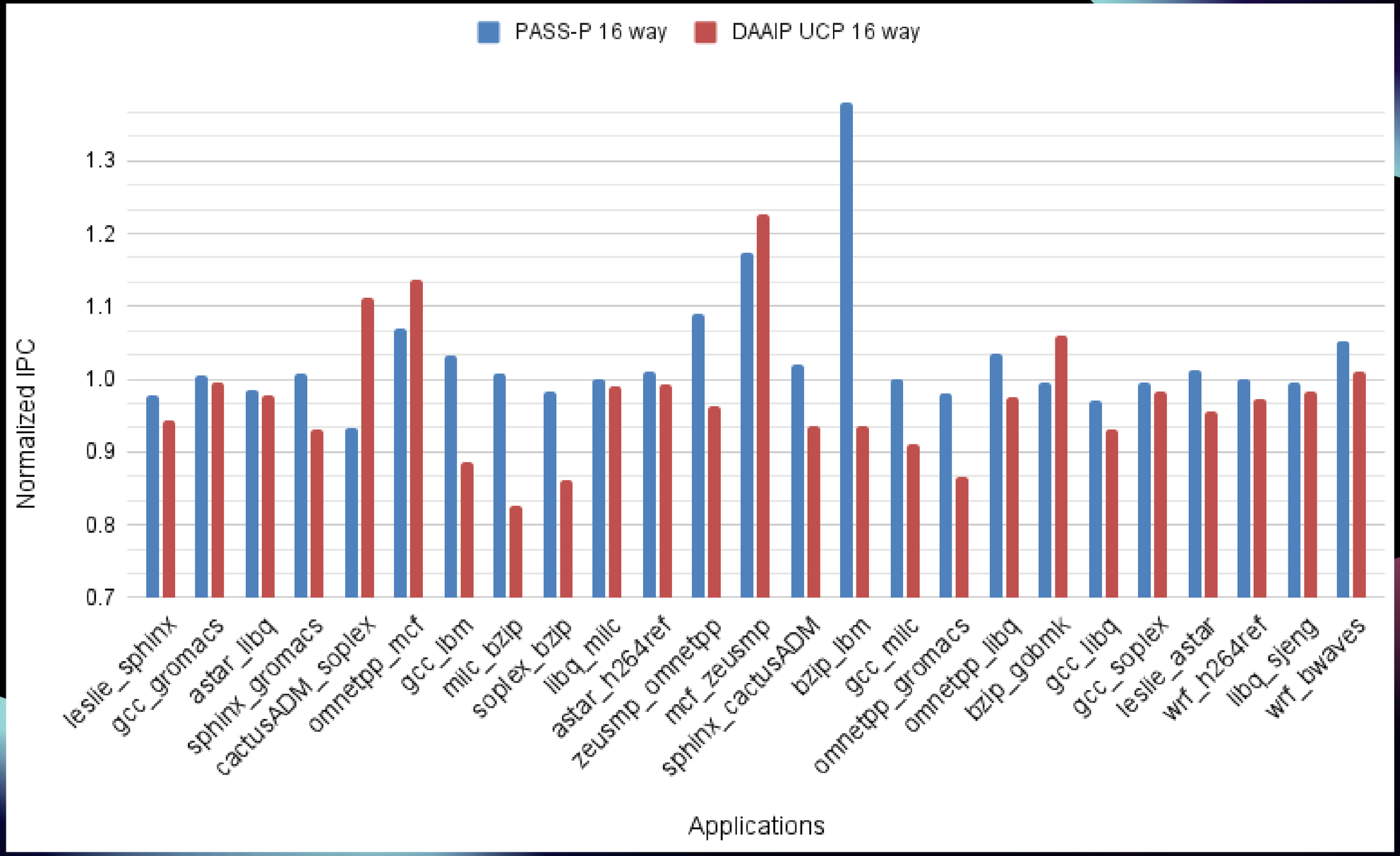


# IPC comparison for different thresholds

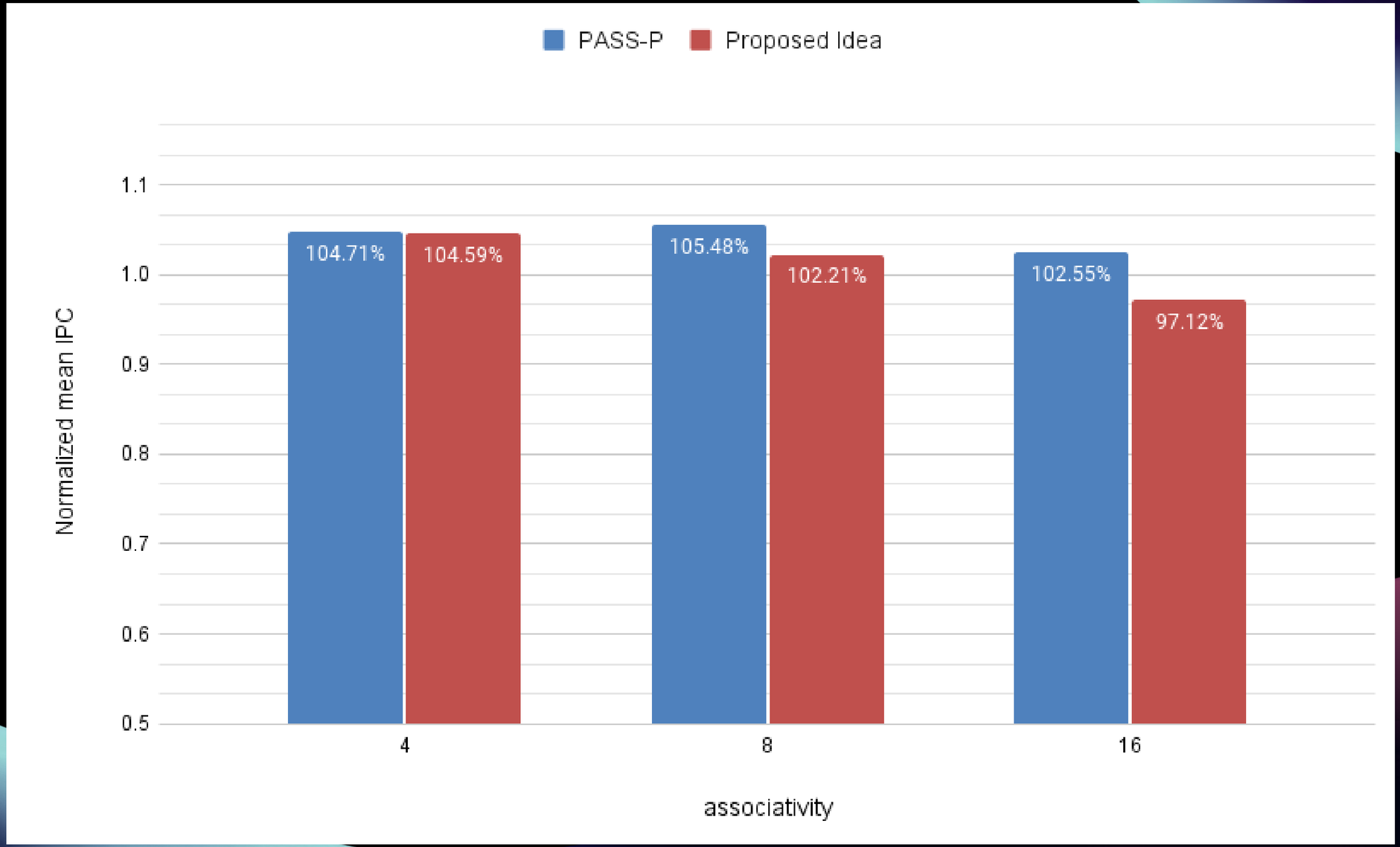



- The threshold is the number of cache accesses after which cache is partitioned
- It is observed that there was a ~5.4% improvement in performance from 1M to 100k

# Comparison with PASS-P



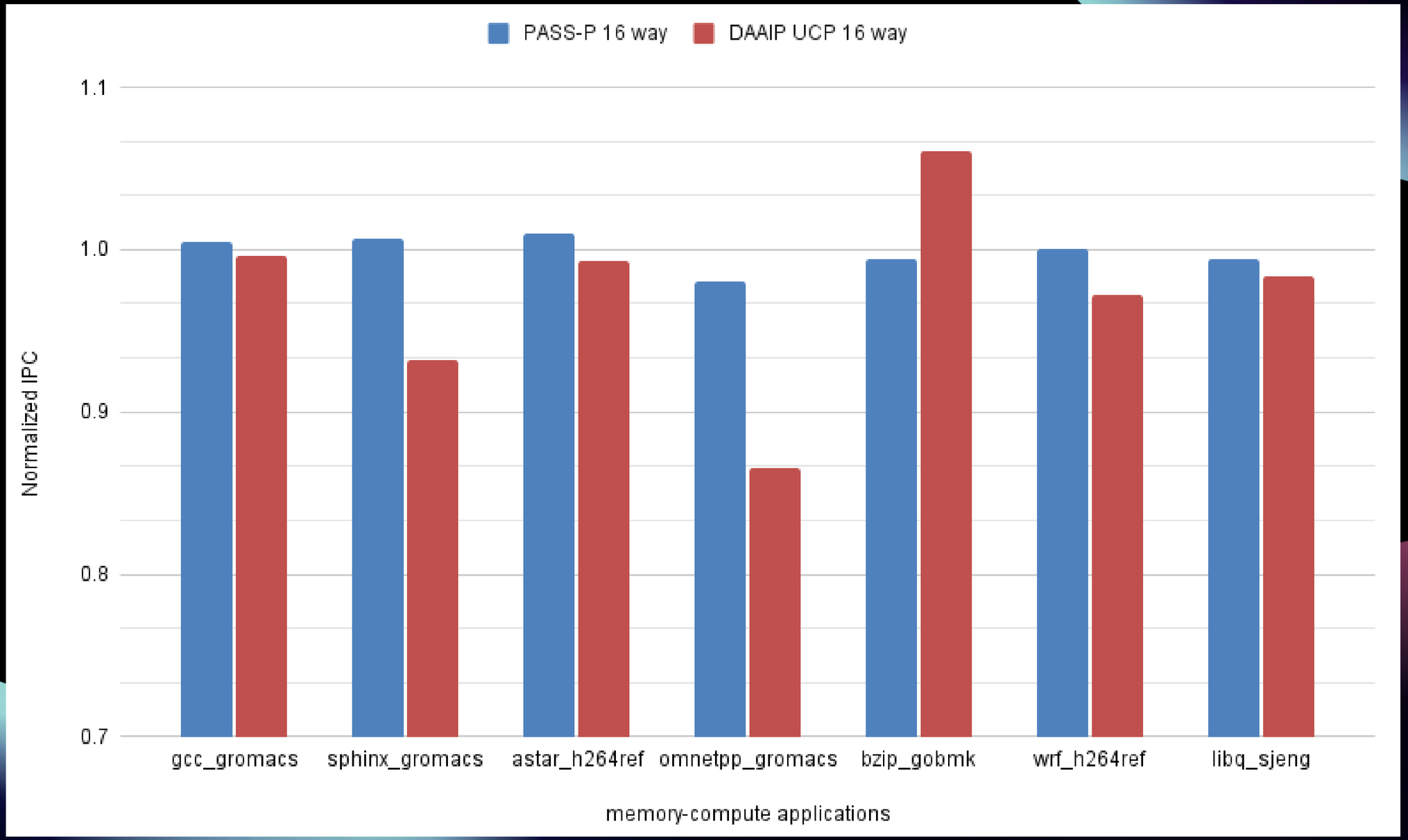
# Comparison with PASS-P



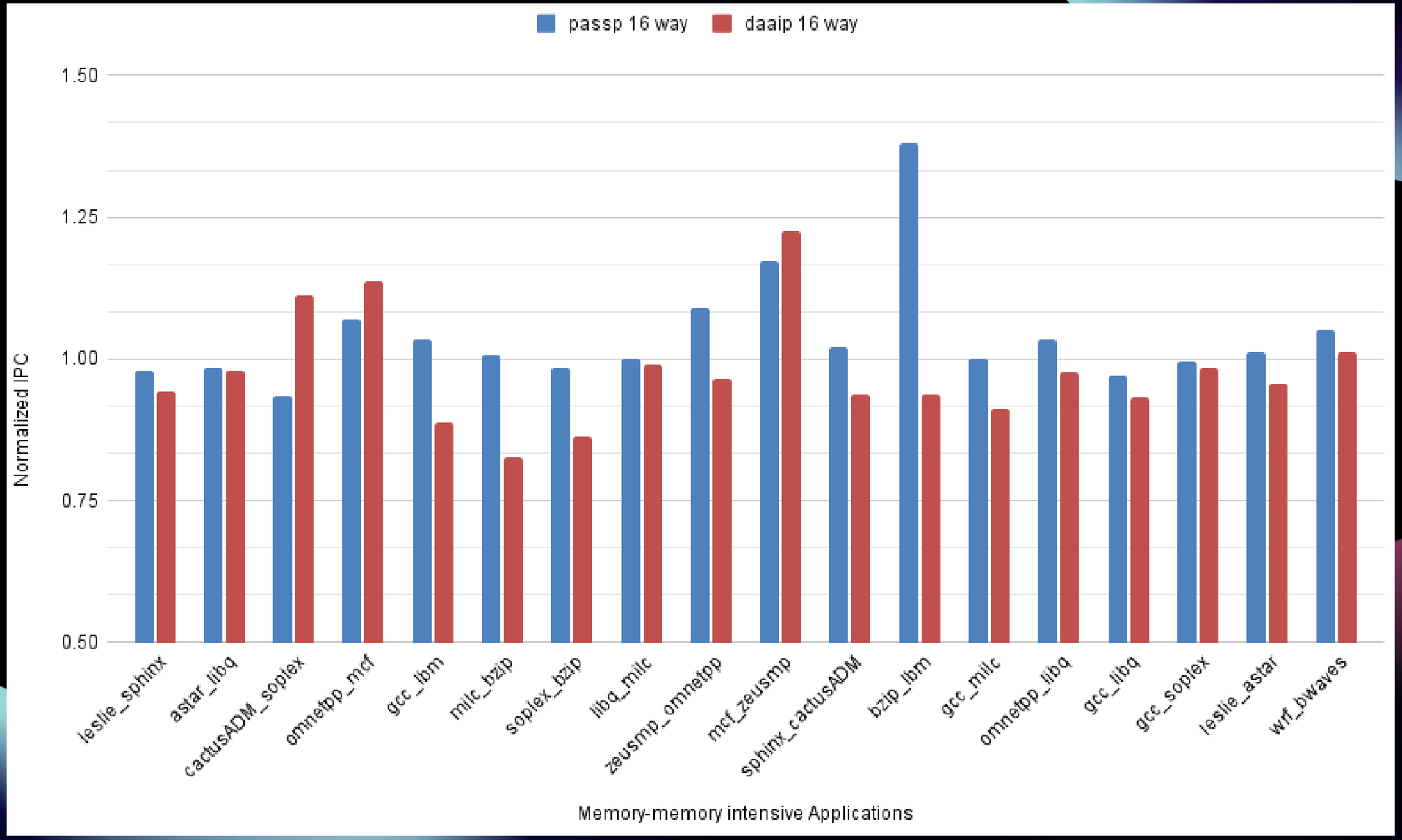
# Comparison with PASS-P

- PASS-P showed a ~5% better performance with 16 way, ~3% for 8 way and ~0.2% for 4 way
- This could arise due to PASS-P selectively reallocating clean lines
- ~41% of the lines evicted or reallocated were dirty
- Thus selecting only clean lines could avoid latencies due to writeback

# Comparison with PASS-P



# Comparison with PASS-P



# Future Work

- Extend the proposed idea to invalidate only clean lines
- Incorporate replacement policies like Mockingjay that have high prediction accuracy.

# Conclusion

- A partition that closely models the core's utility can improve performance by ~10%
- The hardware overhead for the proposed idea is only 256kB and 3 counters above SRRIP
- The proposed idea did not perform better than PASS-P as expected

# DAAIP

- It is a cache replacement algorithm that dynamically updates the insertion policy
- Based on the number of deadblocks encountered, new lines are inserted at LRU or LRU-1.

Newton, S. K. Mahto, S. Pai, and V. Singh, “Daaip: Deadblock aware adaptive insertion policy for high performance caching,” in 2017 IEEE International Conference on Computer Design (ICCD), pp. 345–352, 2017.

# References


1. N. Boran, P. Joshi, and V. Singh, “Pass-p: Performance and security sensitive dynamic cache partitioning,” in Proceedings of the 19th International Conference on Security and Cryptography - Volume 1: SECRYPT,, pp. 443–450, INSTICC, SciTePress, 2022.
2. Newton, S. K. Mahto, S. Pai, and V. Singh, “Daaip: Deadblock aware adaptive insertion policy for high performance caching,” in 2017 IEEE International Conference on Computer Design (ICCD), pp. 345–352, 2017.
3. Y. Yarom and K. Falkner, “Flush+reload: A high resolution, low noise, l3 cache side-channel attack,” in Proceedings of the 23rd USENIX Conference on Security Symposium, SEC’14, (USA), p. 719–732, USENIX Association, 2014
4. F. Liu, Y. Yarom, Q. Ge, G. Heiser, and R. B. Lee, “Last-level cache side-channel attacks are practical,” in 2015 IEEE Symposium on Security and Privacy, pp. 605–622, 2015.

# THANK YOU!