# Analyzing and Addressing Performance Overheads of Confidential Computing on GPU-based Systems

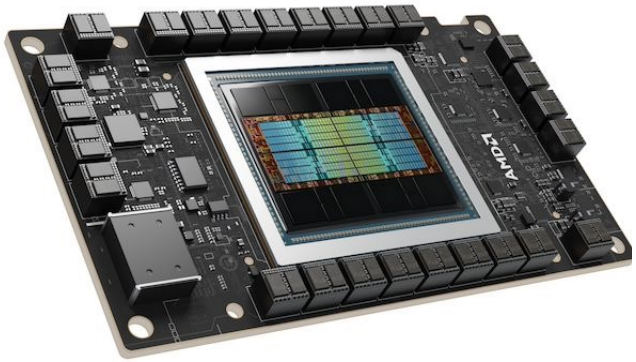
**CC Summit 2025 | June 18, 2025**

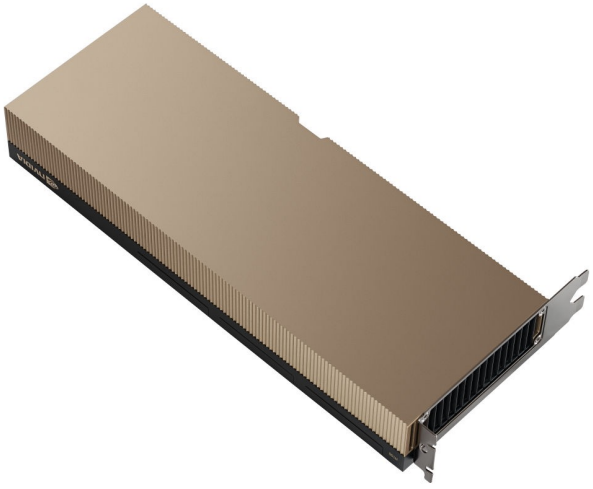
**Adwait Jog**
**Associate Professor & Anita Jones Faculty Fellow**
**Computer Science Department**
**University of Virginia**
**Contact: ajog@viriginia.edu**

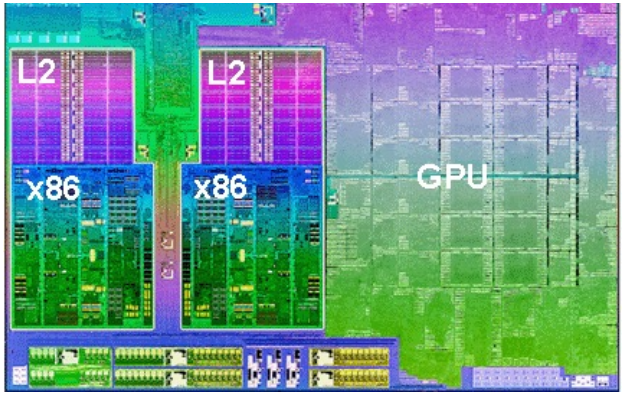
# Applications + Big Data
# + Graphics Processing Units (GPUs)

DL/ML/AI | Graphics | HPC | Crypto | Graphs | AVs

**Abundant Data and Task-level Parallelism**








**High Throughput + Energy Efficiency**

Credit/Source: AMD, NVIDIA, Wiki

# A Decade of GPU Research

**Performance & Energy Efficiency** ⟷ **Security, Privacy & Reliability**

**New Cache and Interconnect Design**
(ISCA'25, HPCA'21, PACT'20, PACT'19)

**Memory Hierarchy Optimizations**
(DSN'19, ICS'19, HPCA'18a)

**Improving Compute/Memory Utilization**
(SIGMETRICS'23, ASPLOS'20, MICRO'18a)

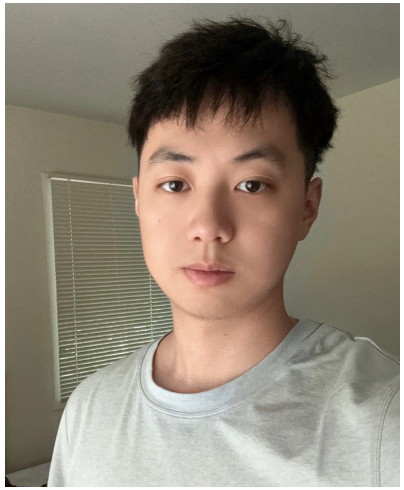
**Simulation and Hardware-Software Co-design**
(ISCA'25, MICRO'24, ISCA'23, MICRO'23)

**Confidential Computing**
(ISPASS'25)

**Side-channel Mitigation**
(HPCA'20, HPCA'18b)

**Fast and Accurate Reliability Analysis**
(CLUSTER'24, SIGMETRICS'21, TC'21, MICRO'18b)

**Low-overhead Protection against Faults**
(ISSRE'24, DSN'21, ICSE'21)



**Yang Yang**
**(Ph.D. student)**

**Mohammad Sonji**
**(Ph.D. student)**

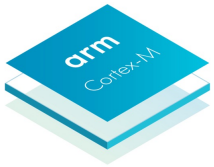
**Adwait Jog**
**(Faculty)**

This presentation is based on the following IEEE publication:

Yang Yang, Mohammad Sonji, Adwait Jog
**Dissecting Performance Overheads of Confidential Computing on GPU-based Systems,**
In the Proceedings of IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), Ghent, Belgium, May 2025 [© 2025 IEEE]

Preprint: https://adwaitjog.github.io/docs/pdf/dissectcc-ispass25.pdf
Code: https://github.com/insight-cal-uva/hcc-ispass25-artifact

# Trusted Execution Environments

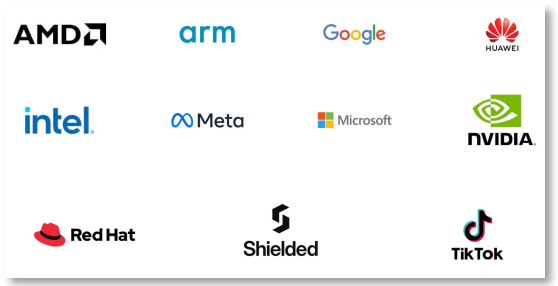


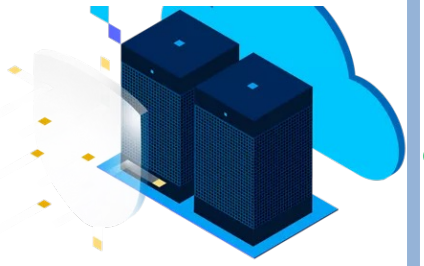

ARM TrustZone **2002**


Intel SGX **2015**


AMD SEV **2016**


CC Consortium **2019**



Intel TDX **2020**

AMD SEV-SNP **2020**

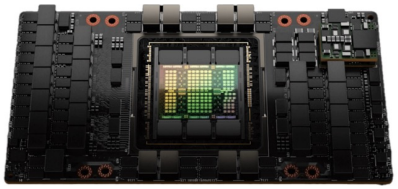
ARM CCA **2021**

**VM-based Isolation**

RISC-V KeyStone **2019**

RISC-V Penglai **2020**

RISC-V®

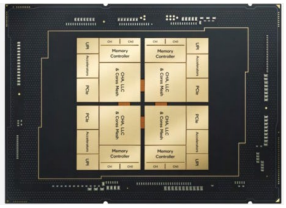


**The first commercial GPU TEE**

NVIDIA HCC **2023**

**CPUs With TDX**

Intel SPR **2023** Intel EMR **2023**



**GPU-based CC**

Credit/Source: ARM/INTEL/AMD/CCC/RISC-V/NVIDIA/Wiki

# Our Short/Long-term Goals

- To advance the understanding of the performance implications of GPU-based confidential computing
- Help bridge any performance gap between CC=on and CC=off
  - …and then do so under different scenarios (e.g., multi-GPUs, CPU-GPU architectures)
- Complement existing and new technological advances in CC through academic research

# Inspired by 2023 NVIDIA paper

- *~40% drop in training performance*
- *As we understand, the main issues are:*
  - Software Encryption at CPU | Bounce Buffers
  - Performance Overheads w/Unified Virtual Memory

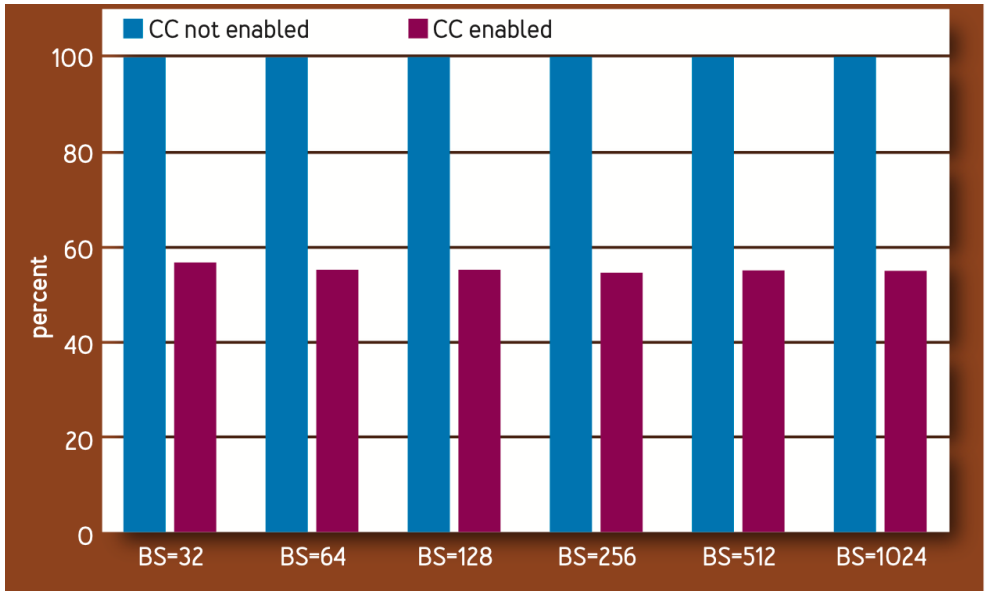



Fig.1 ResNetv1.5 training results*

*Creating the First Confidential GPUs, 2023 © ACM

# Academic Research at UVA

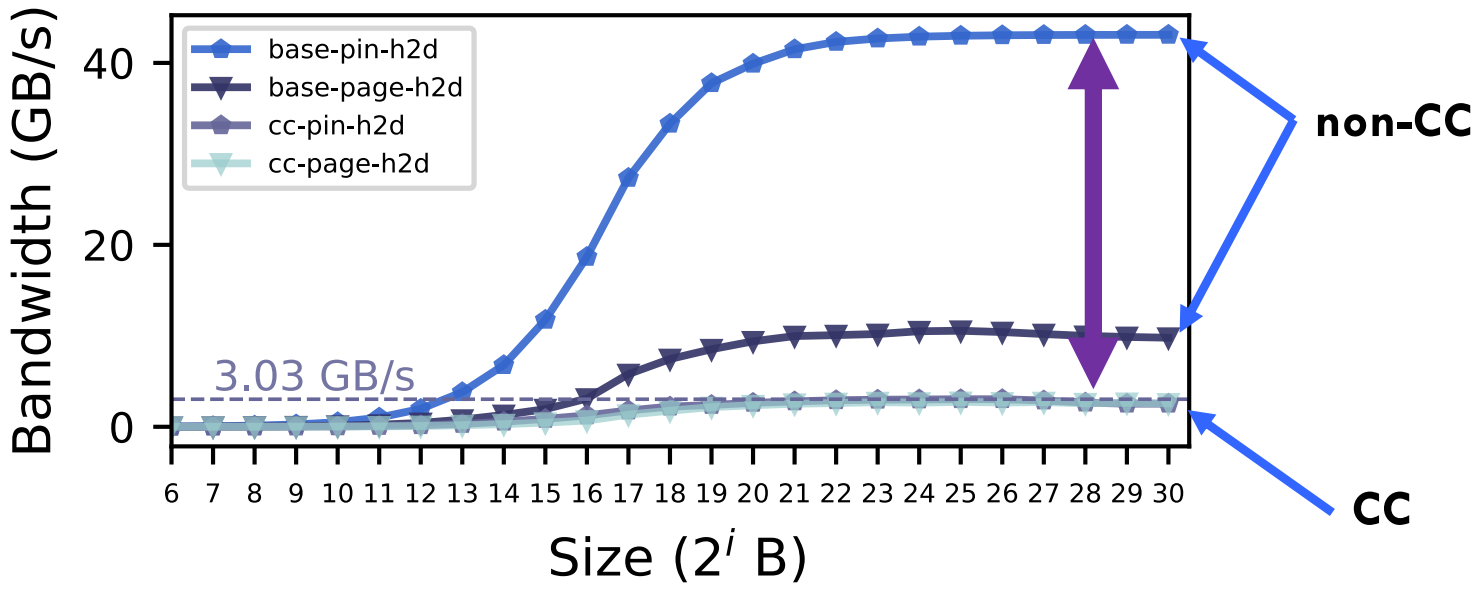
System Configuration

| Configuration | Details |
|---|---|
| **CPU** | 2× 5th Gen Intel Xeon 6530 Gold @2.1GHz, 32 cores |
| **TME-MK** | Auto bypass enabled |
| **OS** | Ubuntu 22.04.5 LTS (Linux 6.2.0, tdx patched) |
| **Hypervisor** | QEMU 7.2.0 (tdx patched) |
| **TDX Tools** | TDX 1.5 (tag 2023ww15) |
| **GPU** | NVIDIA H100 NVL, 94GB HBM3, PCIe 5.0 ×16<br>CUDA 12.4, Driver 550.127.05 |

**Evaluation Setup:**

- CC-capable CPUs: support TDX
- NVIDIA H100 NVL system
- Code available: https://github.com/insight-cal-uva/hcc-ispass25-artifact

# Transfer Bandwidth

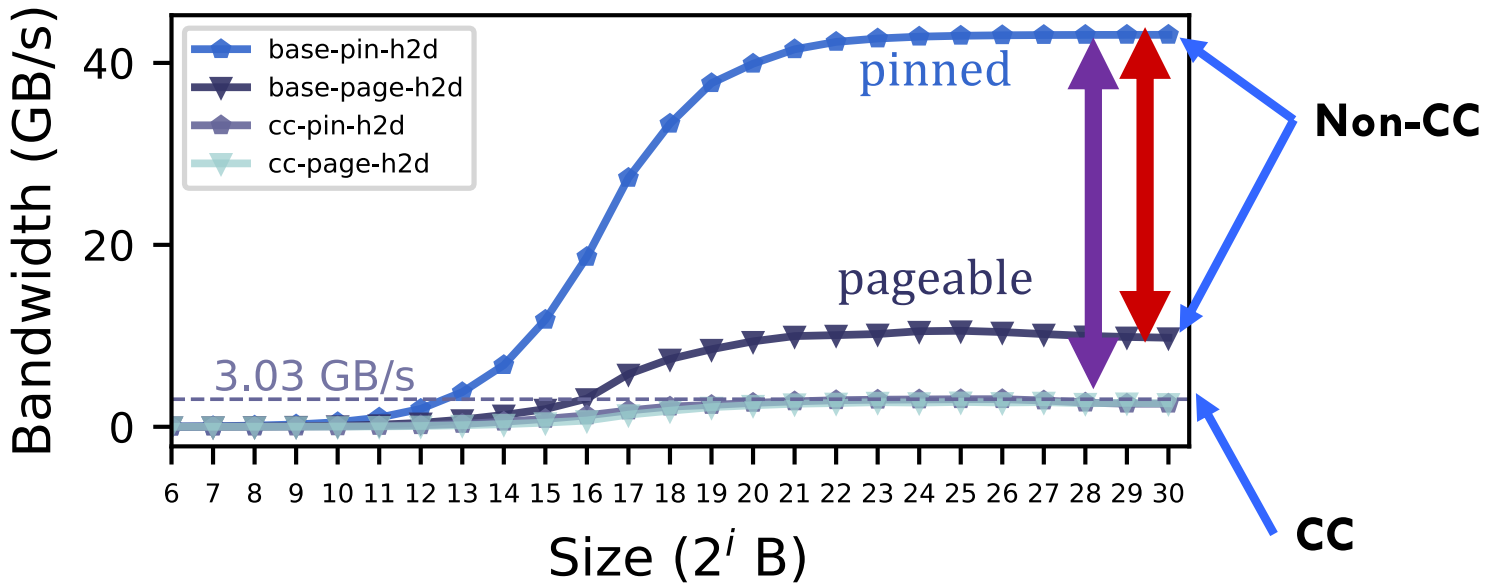



Copy bandwidth

**Observation 1.**

✓ PCIe bandwidth utilization in CC mode **drops** compared to non-CC.

# Transfer Bandwidth

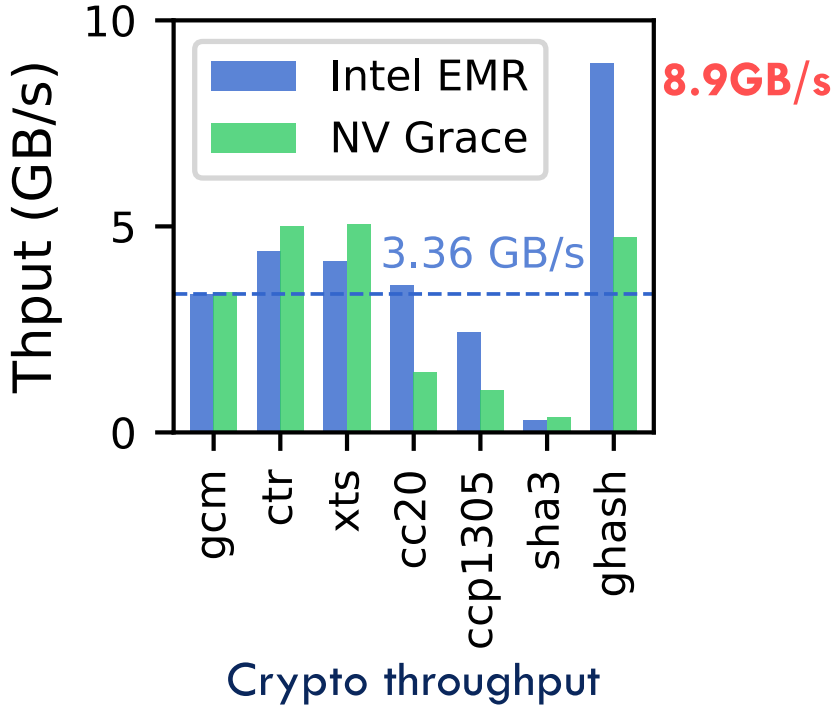



Copy bandwidth

**Observation 1.**

- ✓ PCIe bandwidth utilization in CC mode **drops** compared to non-CC.
- ✓ Bandwidth **gap** between pageable and pinned memory observed in non-CC mode disappears in CC mode, suggesting that pinned memory relies on pageable mechanisms in CC mode.

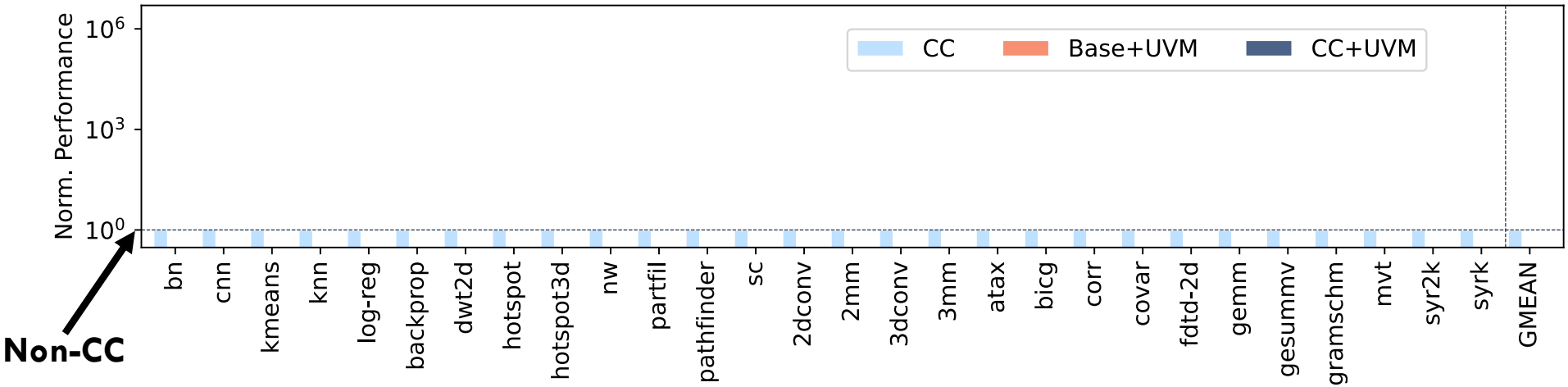
# Crypto Throughput




**Observation 2.**

- ✓ Our findings suggest that absence of **dedicated hardware AES engines** results in low encryption throughput, even when using **AES-NI** acceleration.
- ✓ While *alternative cryptographic algorithms* may offer higher throughput, they may come at the cost of weaker security guarantees.

# Kernel Execution Time



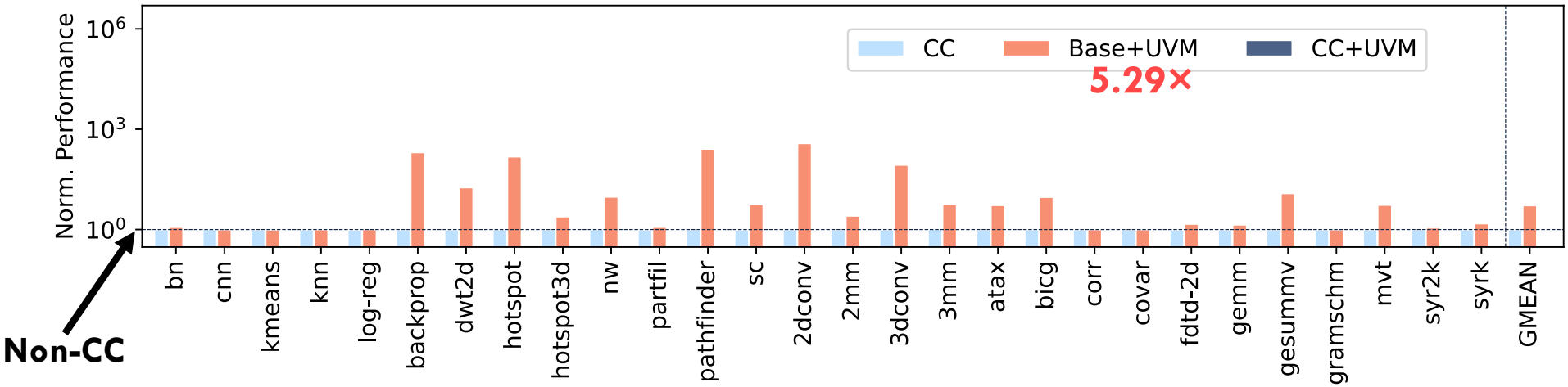

Normalized kernel execution time.

**Observation 3.**

- CC has **minimal** impact on **non-UVM** kernels (**0.48% increase**).

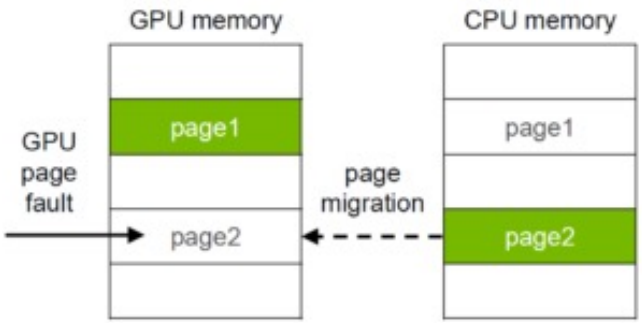
? **Execution is locked inside GPU, no interaction with CPU**

# Kernel Execution Time




Normalized kernel execution time.

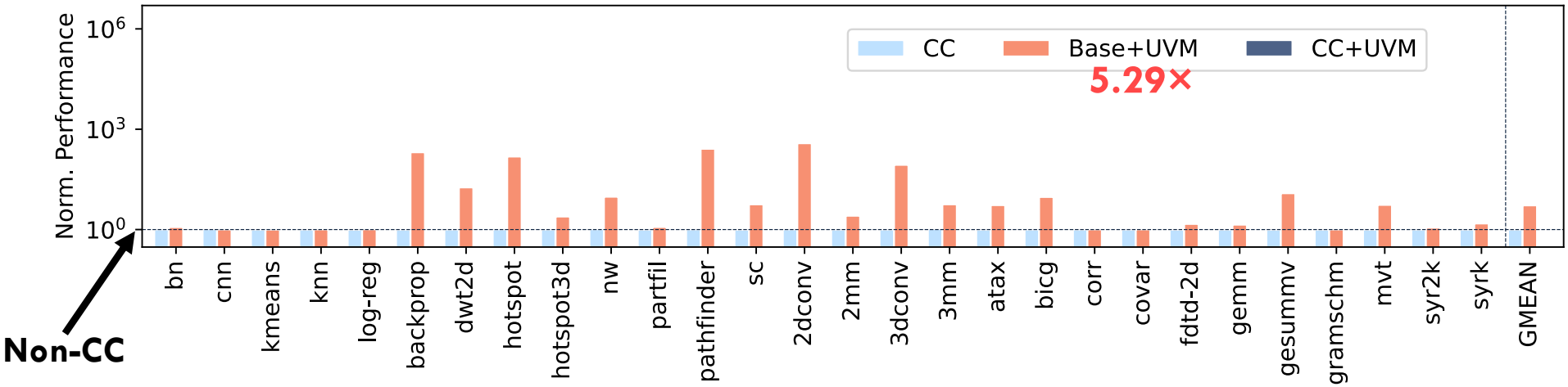
Single Address Space




Source: NVIDIA

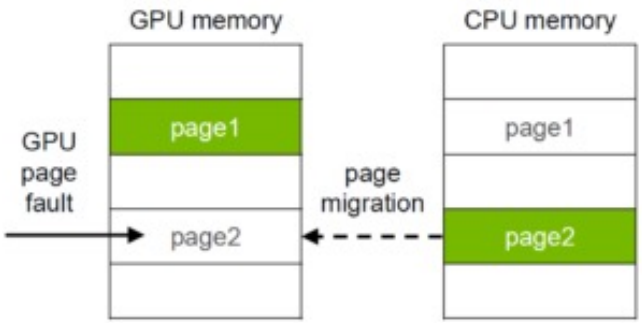
**Frequent CPU-GPU interaction!**

# Kernel Execution Time



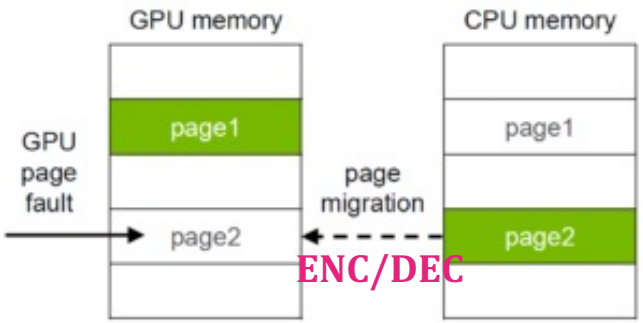

Normalized kernel execution time.

Single Address Space



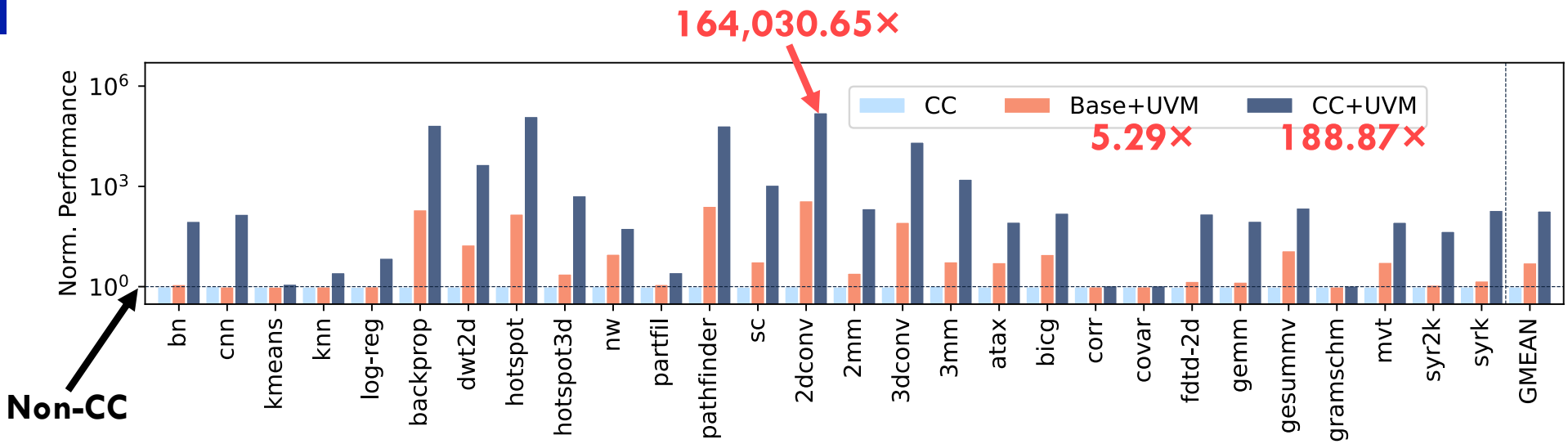

Source: NVIDIA

**Frequent CPU-GPU interaction!**




With CC ON

# Kernel Execution Time

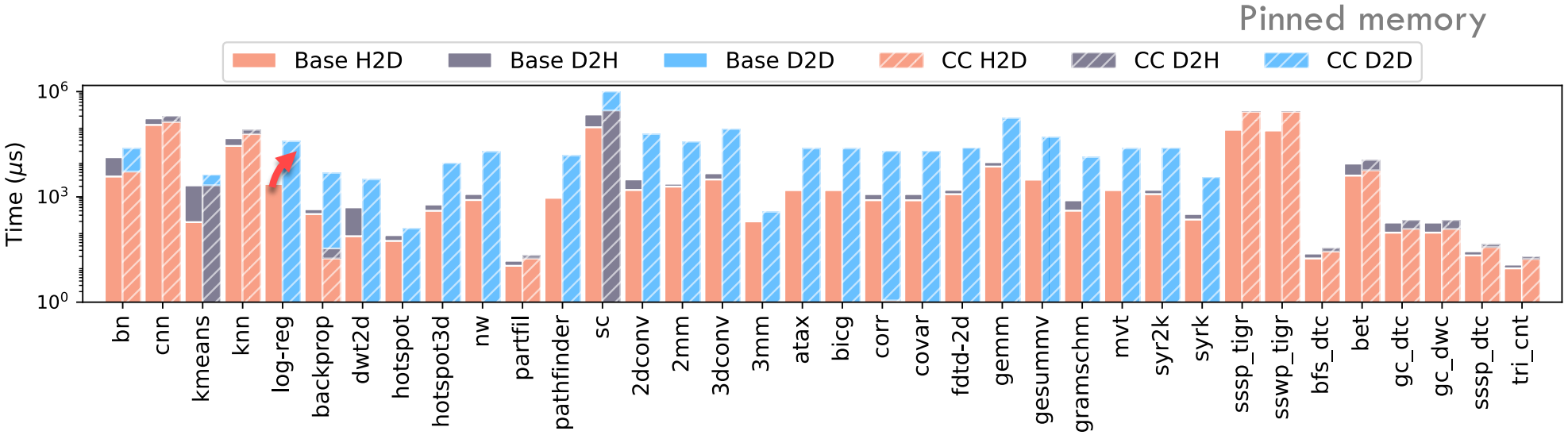



Normalized kernel execution time.

**Observation 3.**

- CC has **minimal** impact on **non-UVM** kernels (**0.48% increase**).
- **UVM** in CC mode incurs an average slowdown of **188.87×** across the GPGPU benchmarks we studied.

# Memory Transfer Time

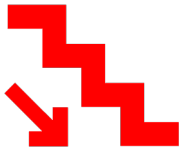


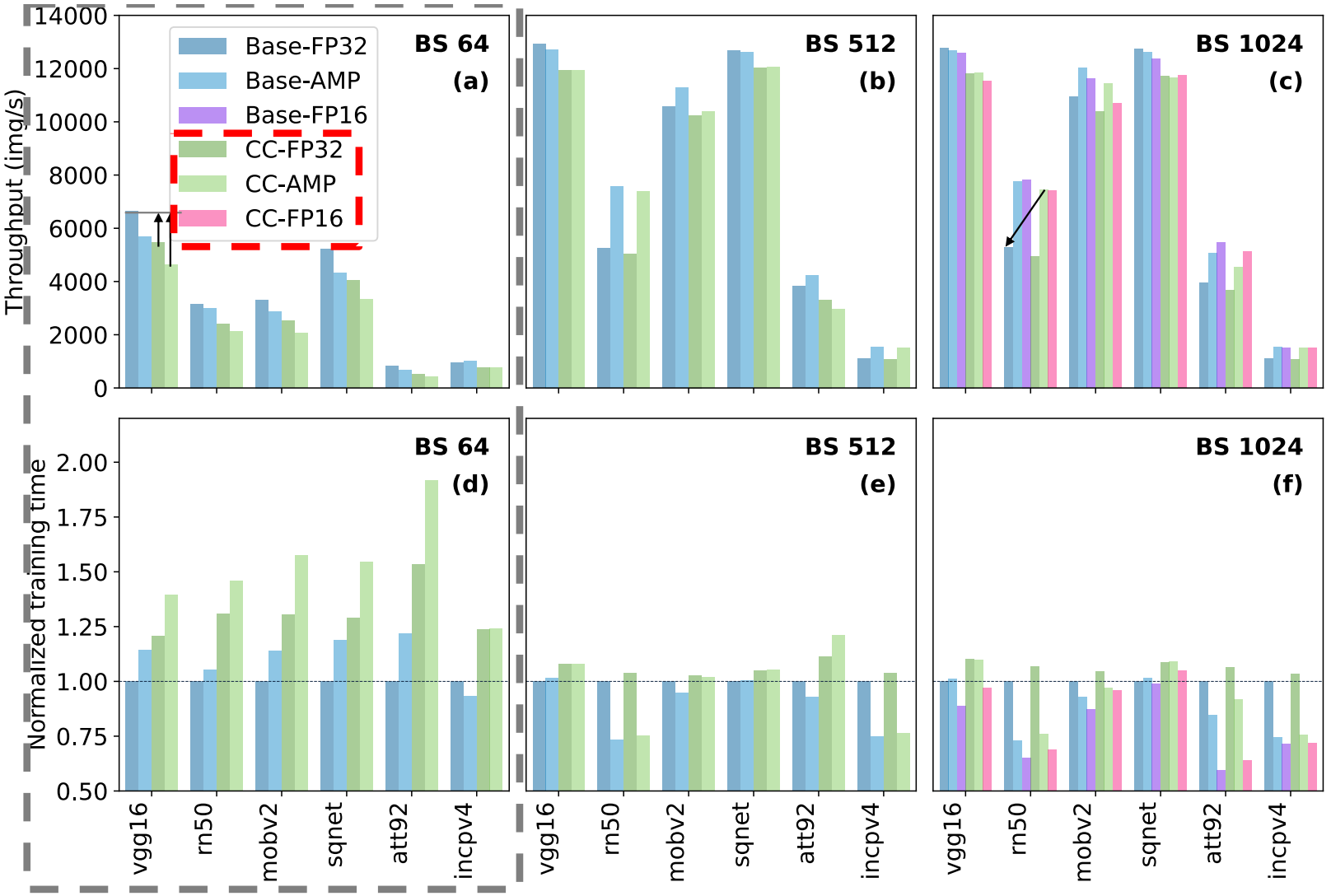

Time ($\mu s$) spent on memory copy.

**Observation 4.**

- ✓ On average, copy operations in CC mode takes **5.80×** longer compared to non-CC mode for the GPGPU benchmarks we studied.
- ✓ We find that pinned memory is converted to UVM encrypted paging in CC mode.

# CNNs



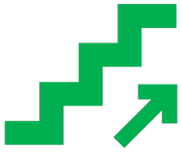


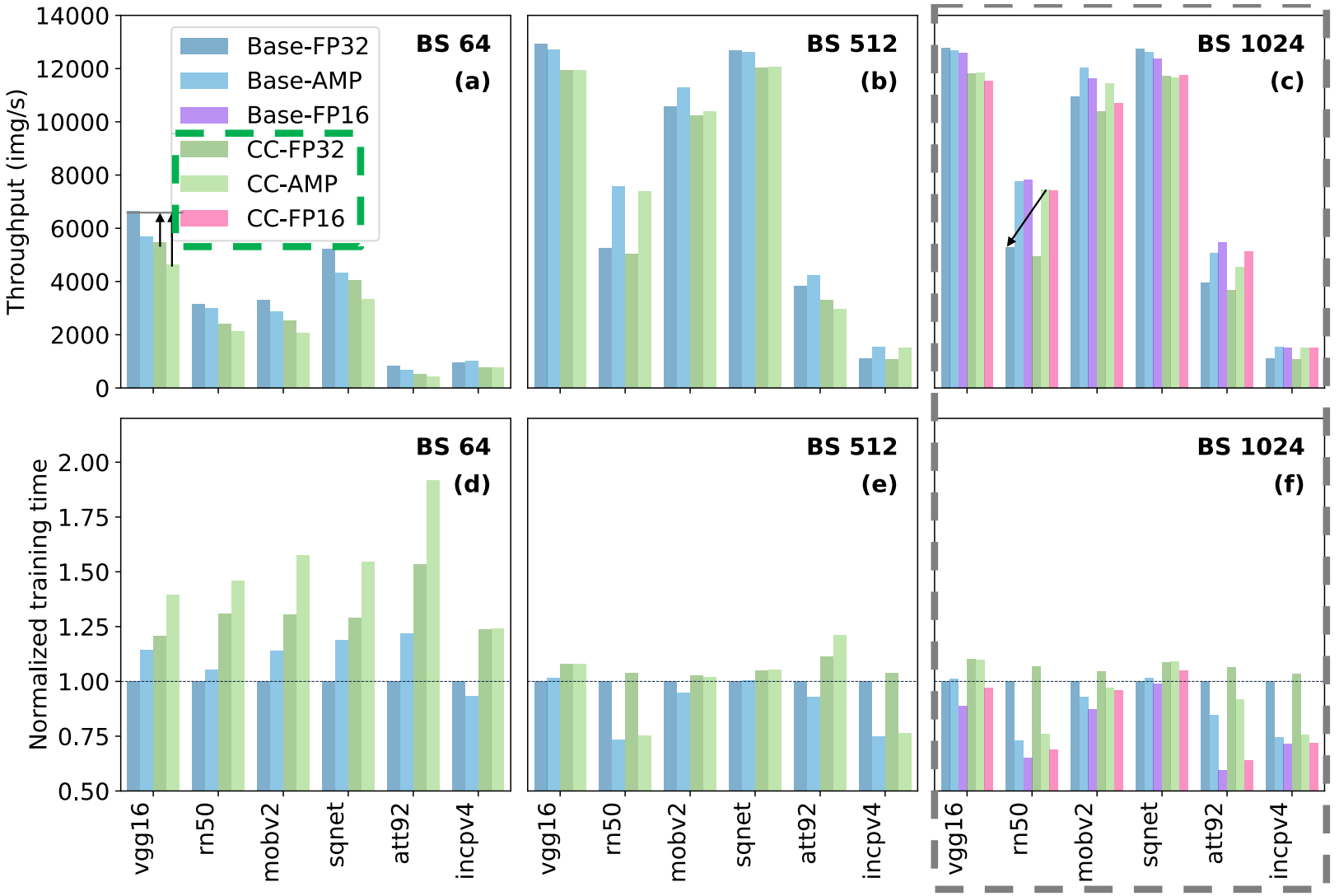

CNN training throughput and training time for different batch sizes.

## Observation 6.

- ✓ With a batch size of 64 and CC on, throughput drops average 24%, and training time increases average 31%.

# CNNs





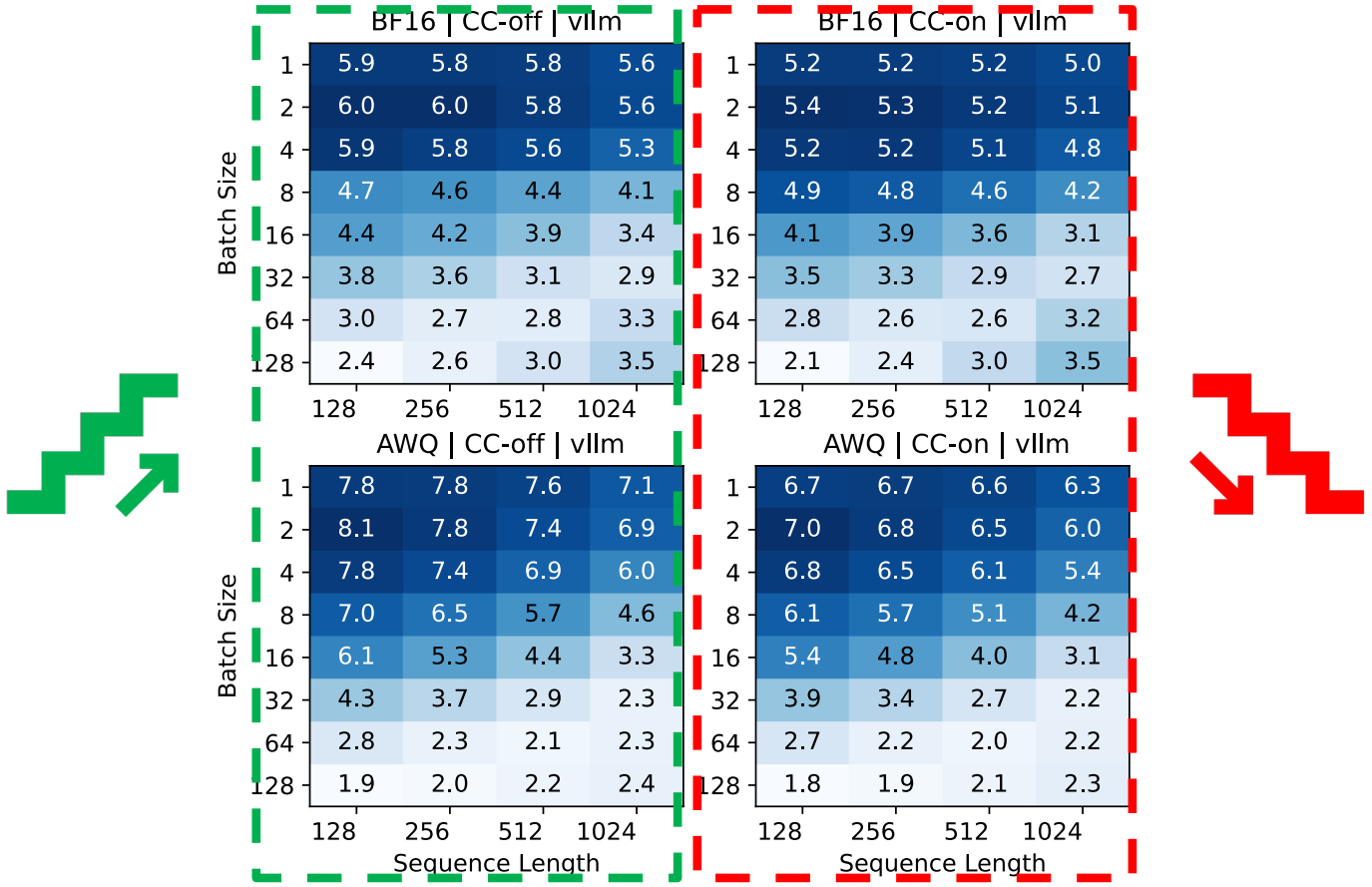

CNN training throughput and training time for different batch sizes.

**Observation 7.**

✓ Increasing the batch size to 1024 significantly reduces overhead, with an average loss in throughput of 7.3% and an increase in training time by 6.7%.

All values are compared to the HuggingFace non-quantized CC-off baseline.

# LLMs

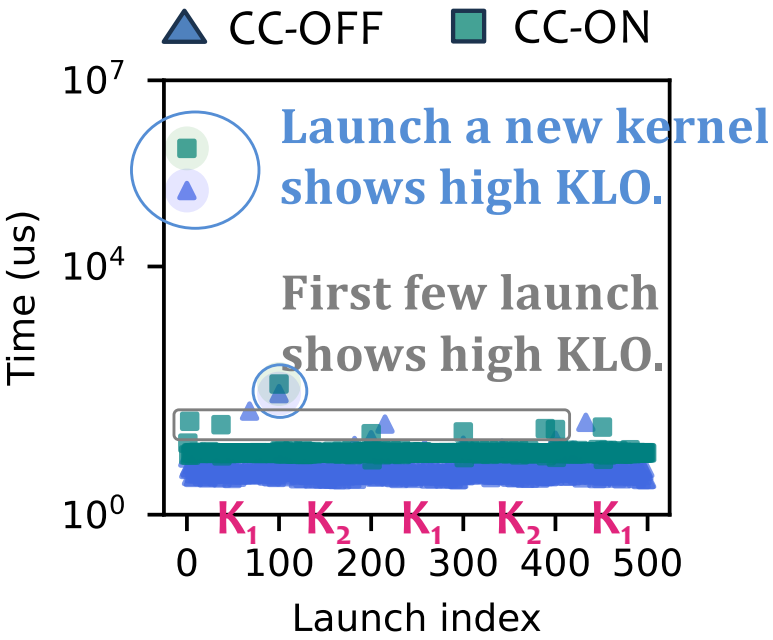


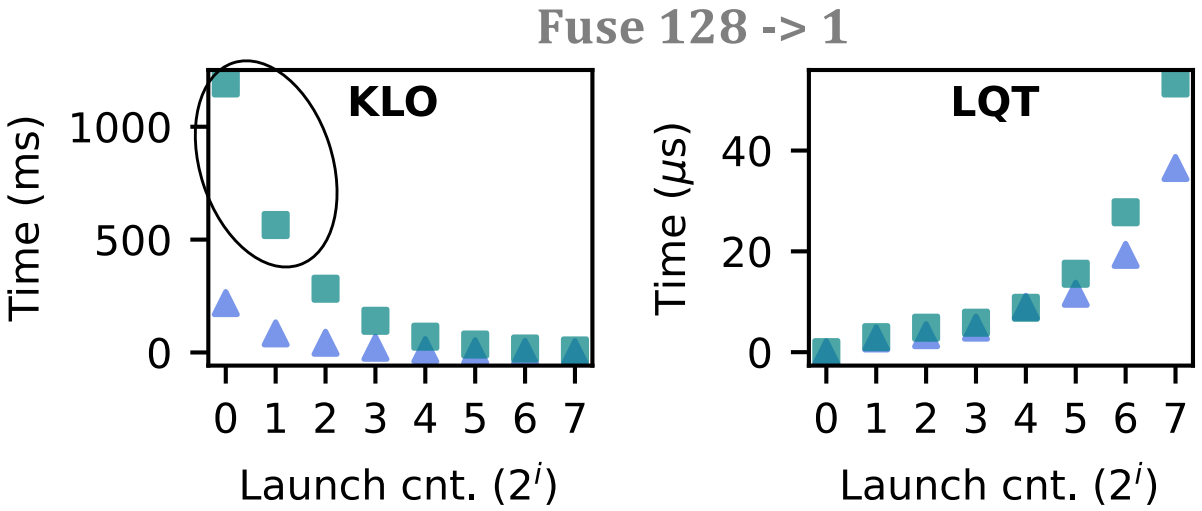

Throughput (Tokens/s) speedup of the vLLM serving framework for the Llama-3-8B model.

**Observation 8.**

✓ CC-on incurs throughput overhead for both BF16 and AWQ, however, quantization benefits remain positive.

# Optimization: Kernel Fusion




**A right kernel fusion parameter is needed for better performance.**

# Performance Model


$$P = T_{mem} + \Sigma(KLO+LQT) + \Sigma[(1-\beta_i)(KET+KQT)] + T_{other}$$

**Focus of this model**

- Data movement (H2D, D2H, page migration)
- Encryption
- Kernel Execution
- Kernel Launch
- Queuing

# Additional Results

- **Memory Management**
- **Kernel-to-Launch Ratio**
- **Overlapping**
- **CNNs**
- **LLMs**
- **Quantization**
- **…**

# More details are in the paper!

Preprint: https://adwaitjog.github.io/docs/pdf/dissectcc-ispass25.pdf
Code: https://github.com/insight-cal-uva/hcc-ispass25-artifact

# Conclusions and Future Work

**Our Goal:** To make GPU-based Confidential Computing more performance efficient and complement existing and new technological advances.

**Future Work:**

- Possible Optimizations: (1) Multi-core Encryption (2) Pro-active Encryption + Scheduling
- Performance Evaluation of CC for systems (when become available) with
  - TDISP solutions w/o bounce buffers
  - Multi-GPUs and CPU-GPUs Architectures
- Performance Models that will consider reliability issues (e.g., bit flips) and Post-Quantum Era (e.g., periodic key updates)

Happy to chat! Adwait Jog (ajog@virginia.edu)