10 Years Later: Cloud Computing is Closing the Performance Gap

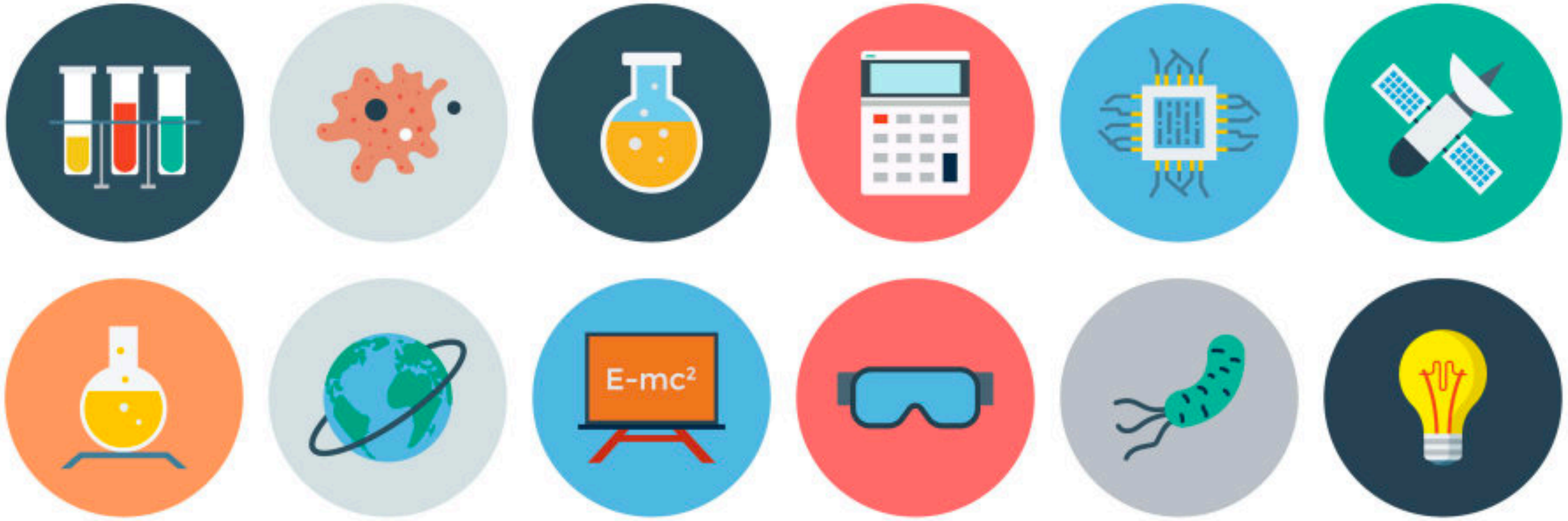
A Multi-Level Approach to Investigate the Performance Gap between HPC and AWS Cloud

Giulia Guidi, Marquita Ellis, Aydın Buluç, Katherine Yelick, David Culler

Tuesday May, 11th

A New Dawn for High-Performance Computing

The benefit of **high-performance computing** for science **has grown rapidly in the recent years** due to the increasing need for computational resources in **data analysis** and **machine learning** for science in addition to simulations



A Brief History of HPC in the Cloud

In the **literature of the last 10 years**, there have been several efforts to measure the performance of scientific applications in the cloud:

He et al. concluded that the **high latency network** is the main bottleneck and that **virtualization has no significant overhead**

The Magellan report concluded that **high latency network, virtualization overhead, performance variability** and **lack of batch scheduling** are the main bottleneck

Gupta et al. also concluded that the **high latency network, virtualization overhead** and **performance variability** are the main bottleneck

Netto et al. surveyed the contemporary literature and concluded that the **lack of low-latency network is the main challenge for the cloud**

A **key take away** is that the <u>lack of a low-latency network</u> has prevented the cloud from achieving competitive performance on a broad scale and **this has not changed in 8 years between 2010 and 2018**

But How's the Situation Today?

In the **literature of the last 10 years**, there have been several efforts to measure the performance of scientific applications in the cloud:

The Magellan report concluded that **high latency network, virtualization overhead, performance variability** and **lack of batch scheduling** are the main bottleneck

Netto et al. surveyed the contemporary literature and concluded that the **lack of low-latency network is the main challenge for the cloud**

He et al. concluded that the **high latency network** is the main bottleneck and that **virtualization has no significant overhead**

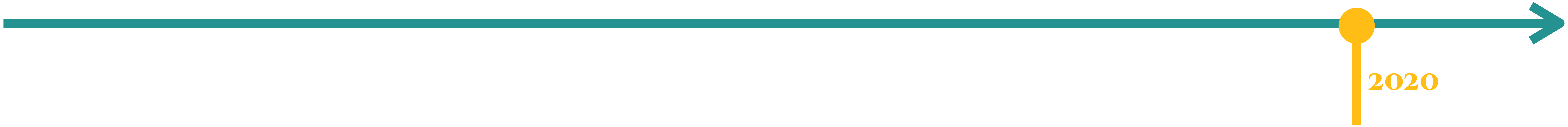
Gupta et al. also concluded that the **high latency network, virtualization overhead** and **performance variability** are the main bottleneck

?

?

A **key take away** is that the lack of a low-latency network has prevented the cloud from achieving competitive performance on a broad scale and **this has not changed in 8 years between 2010 and 2018**

But How's the Situation Today?



That's what **this presentation** is about!

Our **results in a nutshell**:

- The compute and memory subsystem performance of cloud instances is competitive with HPC systems, **in line with previous literature**
- The **cloud platform optimized for memory-intensive workload** significantly outperformed all other machines **overturning historical results**
  - In particular, **significant advances in communication performance**

HPC and cloud computing have been compared for a long time —**why have the results changed now?**

A Step Back: The Context is Important

Cloud computing and traditional HPC have **different purposes**, **economic objectives** and **access policies:**

HPC

- Designed to for **dedicated scientific computing**
- Operated by a **non-profit organization**, funded by a **government agency**
- Devoted to a **particular research community**
- High utilization (> 90%) and possibly long wait times
- Large-scale, **homogeneous hardware**

Cloud

- Designed for **general use**
- Built **for profit**
- Configured to meet **market demand**
- Lower utilization rates and little (or no) wait times
- Frequently-updated, **heterogenous hardware**

Growing commercial interest in large-scale machine learning training has led to an increasing popularity of HPC in the cloud, triggering configuration changes and **resurfacing questions about use of the cloud for scientific computing**

A Step Back: The Context is Important

Cloud computing and traditional HPC have **different purposes**, **economic objectives** and **access policies:**

HPC

- Designed to for **dedicated scientific computing**
- Operated by a **non-profit organization**, funded by a **government agency**
- Devoted to a **particular research community**
- High utilization (> 90%) and possibly long wait times
- Large-scale, **homogeneous hardware**

Cloud

- Designed for **general use**
- Built **for profit**
- Configured to meet **market demand**
- Lower utilization rates and little (or no) wait times
- Frequently-updated, **heterogenous hardware**

Cloud heterogeneity can rapidly provision new hardware for applications that require the latest technology — However, **it could limit the ability to reserve a large number of HPC-like instances** for large-scale scientific computing

A Step Back: The Context is Important

Cloud computing and traditional HPC have **different purposes**, **economic objectives** and **access policies:**

HPC	**Cloud**
• Designed to for **dedicated scientific computing**	• Designed for **general use**
• Operated by a **non-profit organization**, funded by a **government agency**	• Built **for profit**
• Devoted to a **particular research community**	• Configured to meet **market demand**
• High utilization (> 90%) and possibly long wait times	• Lower utilization rates and little (or no) wait times
• Large-scale, **homogeneous hardware**	• Frequently-updated, **heterogenous hardware**

For the cloud, we don't know the bisection/global bandwidth at large scales, which might limit us when running large-scale applications

Our Approach

Here, we **isolate the contribution of the different variables** to the performance gap:

Hardware and System

- **Processor**: LINPACK benchmark
- **Memory Bandwidth**: STREAM benchmark
- **Memory Hierarchy**: CacheBench benchmark
- **Inter-Node Communication**: OSU microbenchmark

User Application

- **Compute-intensive**: N-body simulation
- **Communication-intensive**: FFT

To characterize application performance we use a subset of **hardware events** and the **communication-to-computation ratio**

Experimental Setting

To carry out our study we used **four machines**:

	Platform	Age	Core/P	Fr (GHz)	Processor	Memory (GiB)	Network (Gbps)	L1	L2	L3
HPC	Cori Haswell	4	32	2.3	Xeon E5-2698V3	120	82	64 KB	256 KB	40 MB
	Cori KNL	4	68	1.4	Xeon Phi 7250	90	82	64 KB	1 MB	-
Cloud	AWS r5dn.16xlarge (R5)	1	32	2.5	Xeon Platinum 8259CL	512	75	64 KB	1 MB	36MB
	AWS c5.18xlarge (C5)	1	36	3.0	Xeon Platinum 8124M[1]	144	25	64 KB	1 MB	25MB

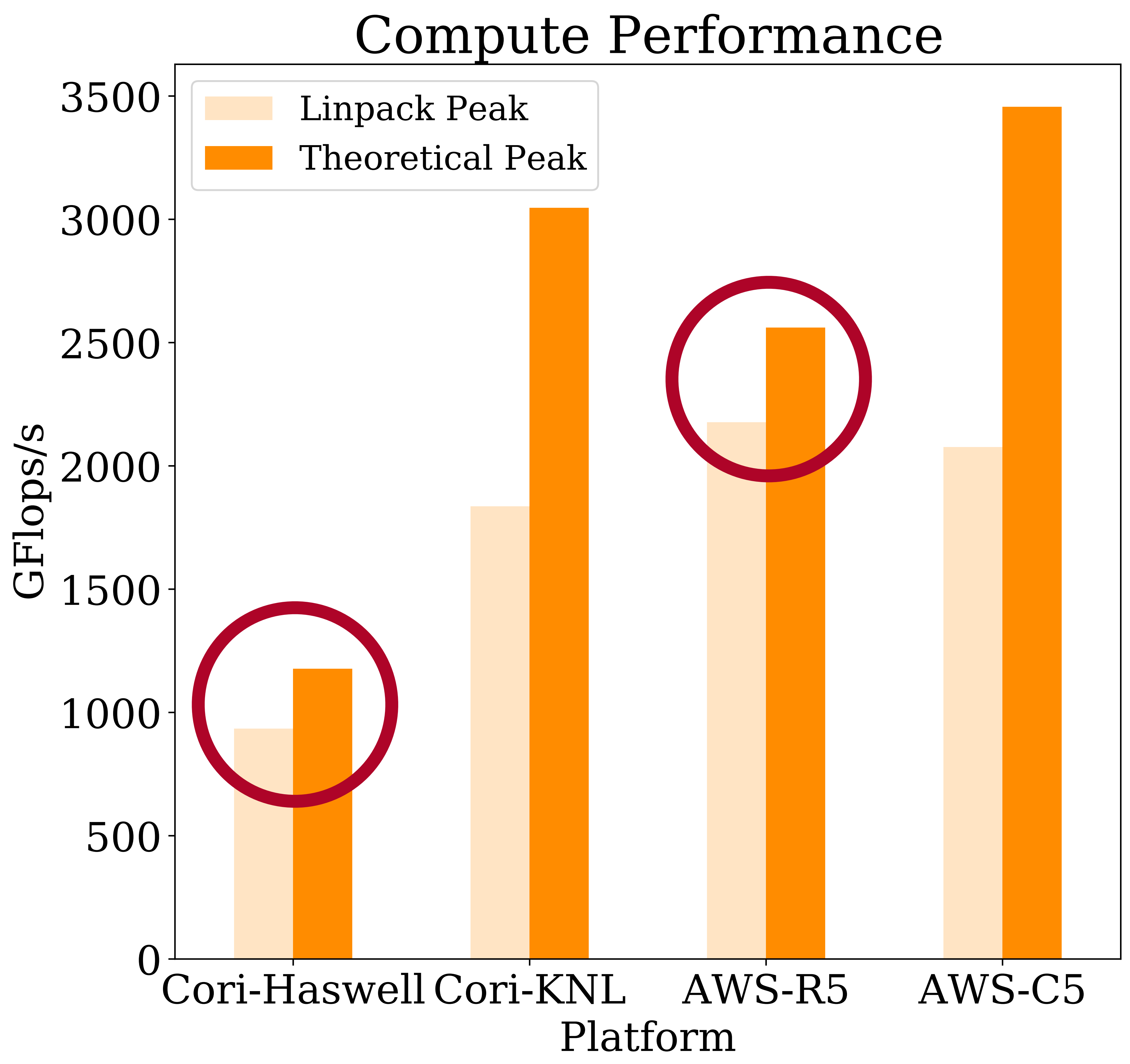
- **AWS ParallelCluster** to set up the cluster
- AWS instances run as **dedicated instances**
- AWS instances placed in the **same placement group**
- High-end instances, i.e. expensive instances
- Cori has the Cray Aries "Dragonfly" topology for its interconnect
- On Cori, we tested **both** Cray-MPICH and OpenMPI —Performance were comparable so we used **OpenMPI**
- Cori KNL was used in the default quad-cache mode

1 Custom model for Amazon AWS

A Hardware and System View

Processor: LINPACK benchmark

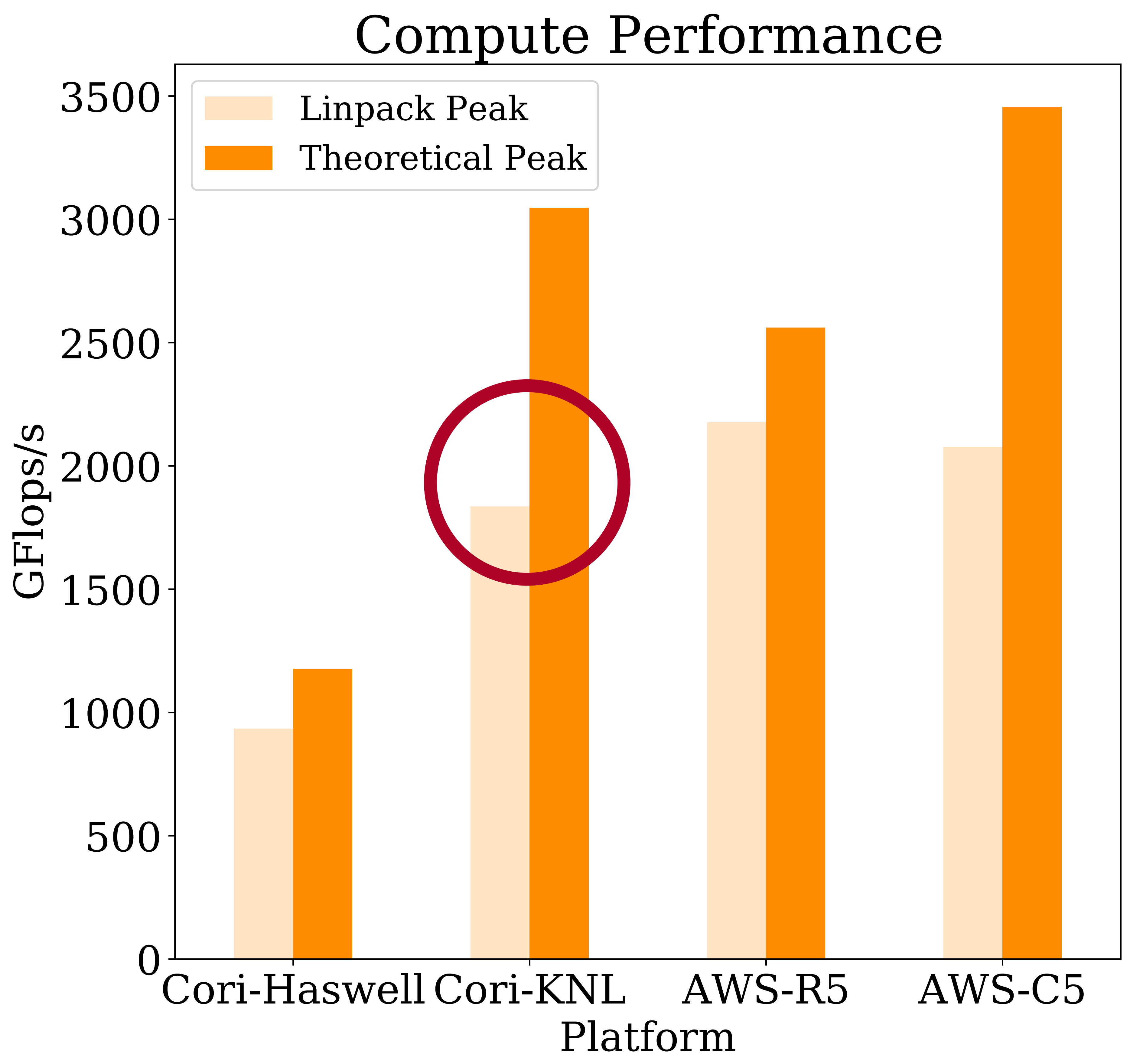
- Cori Haswell and R5 peak performance **much closer** to their theoretical peak than the other two machines



A Hardware and System View

Processor: LINPACK benchmark

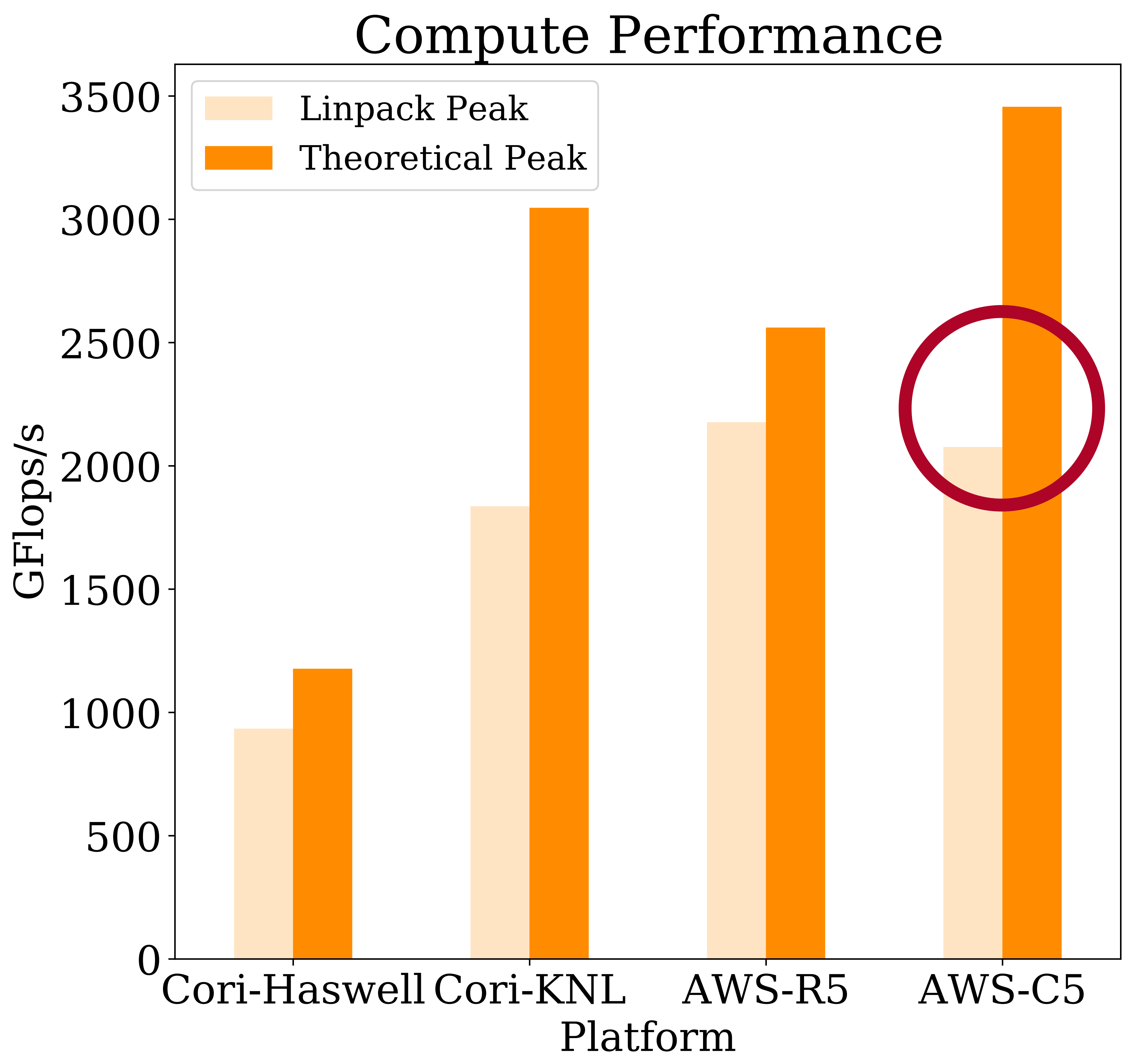
- Cori Haswell and R5 peak performance **much closer** to their theoretical peak than the other two machines
- Closing the gap between theoretical peak and LINPACK peak on Cori KNL is **notoriously difficult**



A Hardware and System View

Processor: LINPACK benchmark

- Cori Haswell and R5 peak performance **much closer** to their theoretical peak than the other two machines
- Closing the gap between theoretical peak and LINPACK peak on Cori KNL is **notoriously difficult**
- C5's profiling revealed relatively **low core utilization** which could explain the gap

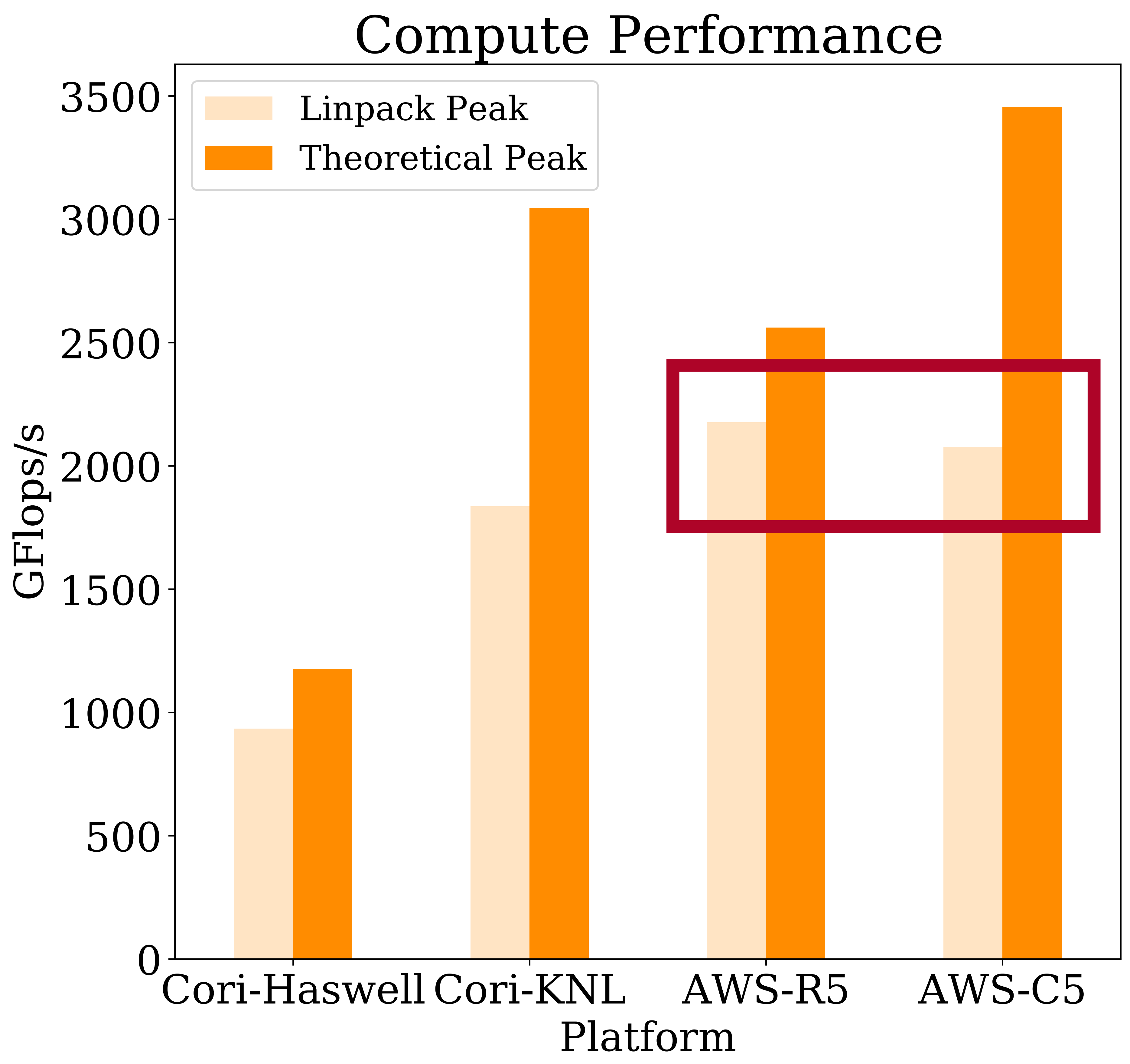


A Hardware and System View

Processor: LINPACK benchmark

- Cori Haswell and R5 peak performance **much closer** to their theoretical peak than the other two machines
- Closing the gap between theoretical peak and LINPACK peak on Cori KNL is **notoriously difficult**
- C5's profiling revealed relatively **low core utilization** which could explain the gap

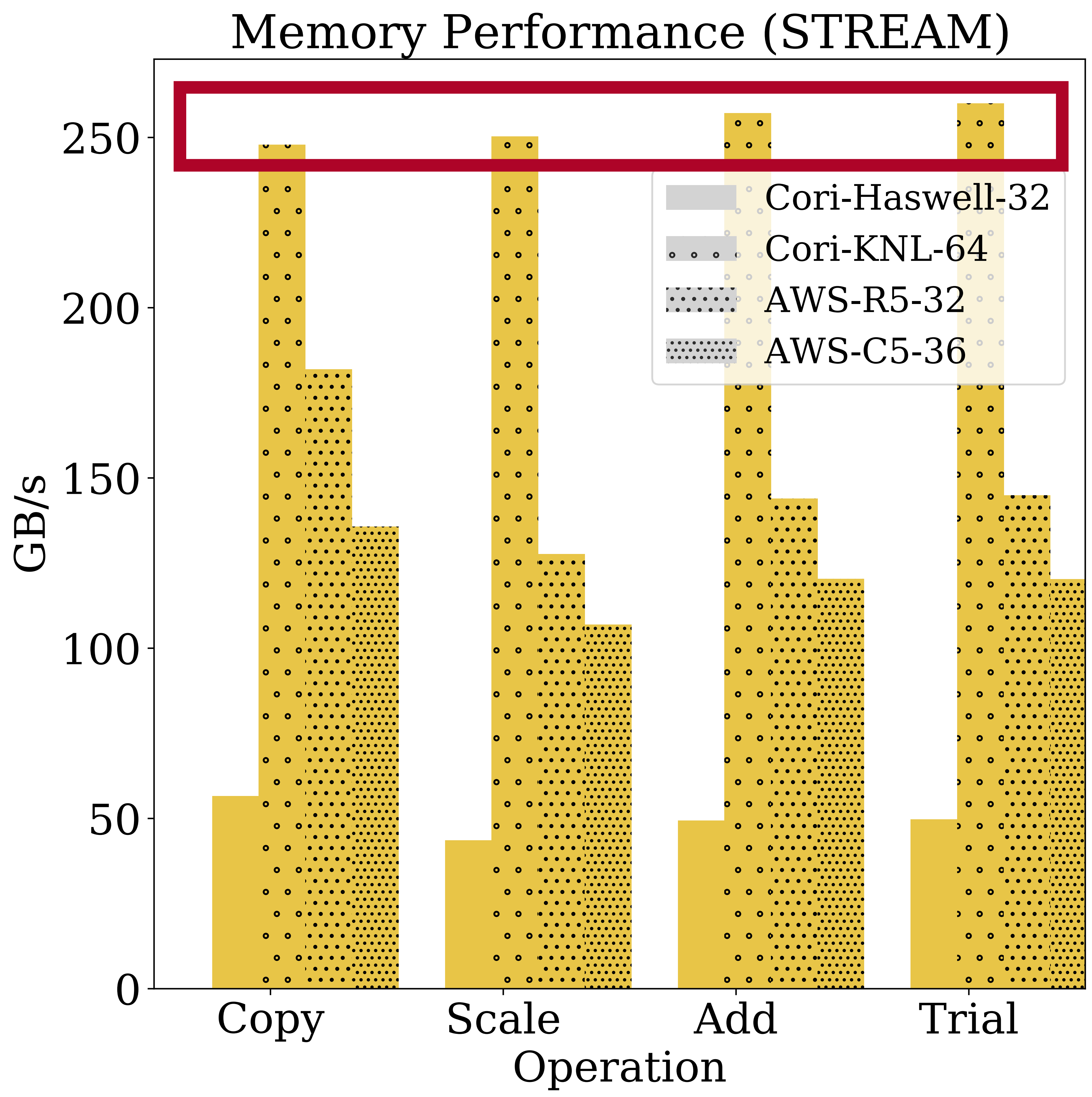
Take away: Cloud's faster procurement cycles —thus the newer hardware —may explain the greater processing power



A Hardware and System View

Memory Bandwidth: STREAM benchmark

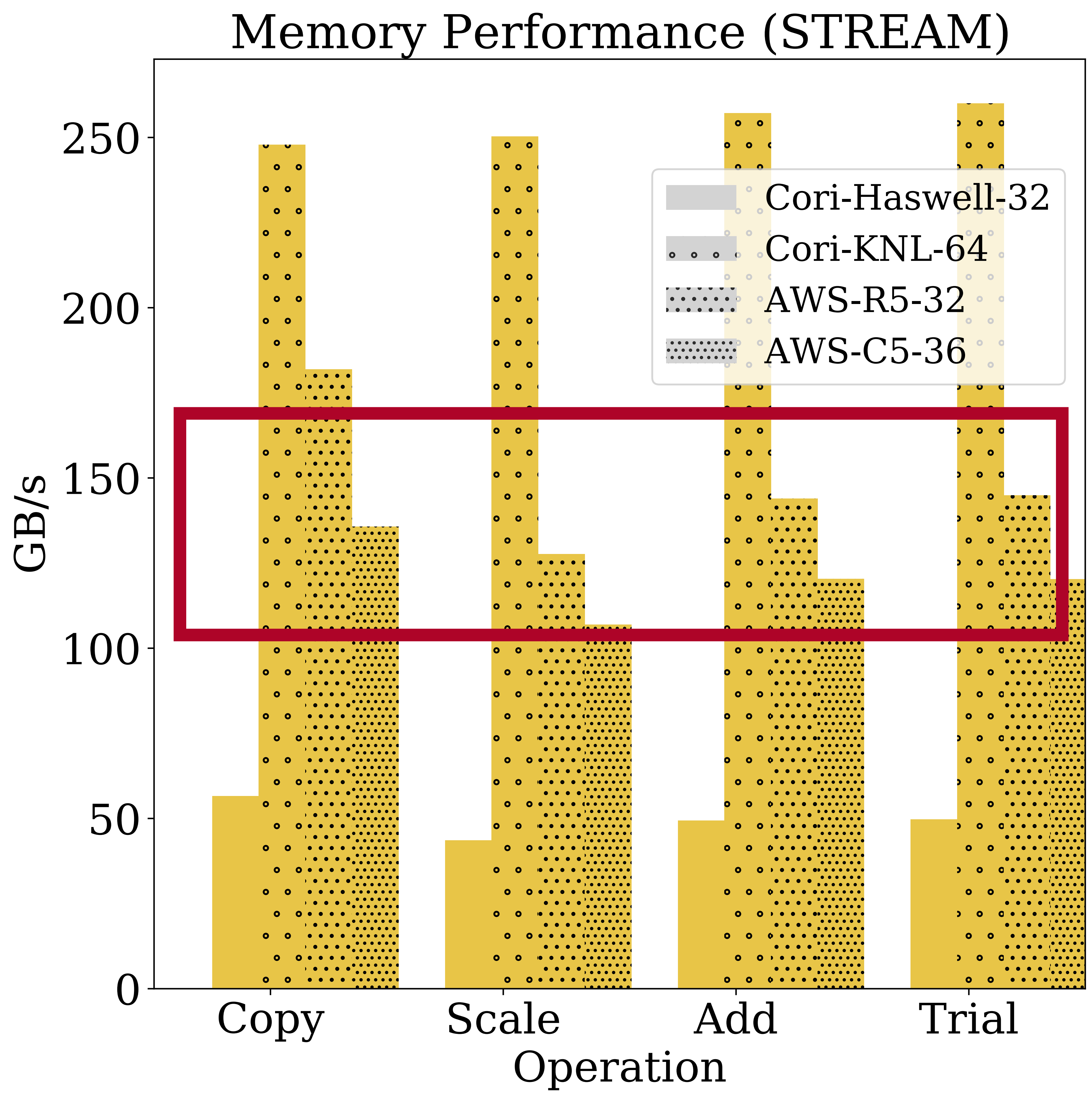
- Cori KNL has the higher memory bandwidth thanks to the on-chip multi-channel DRAM chip of 16GB



A Hardware and System View

Memory Bandwidth: STREAM benchmark

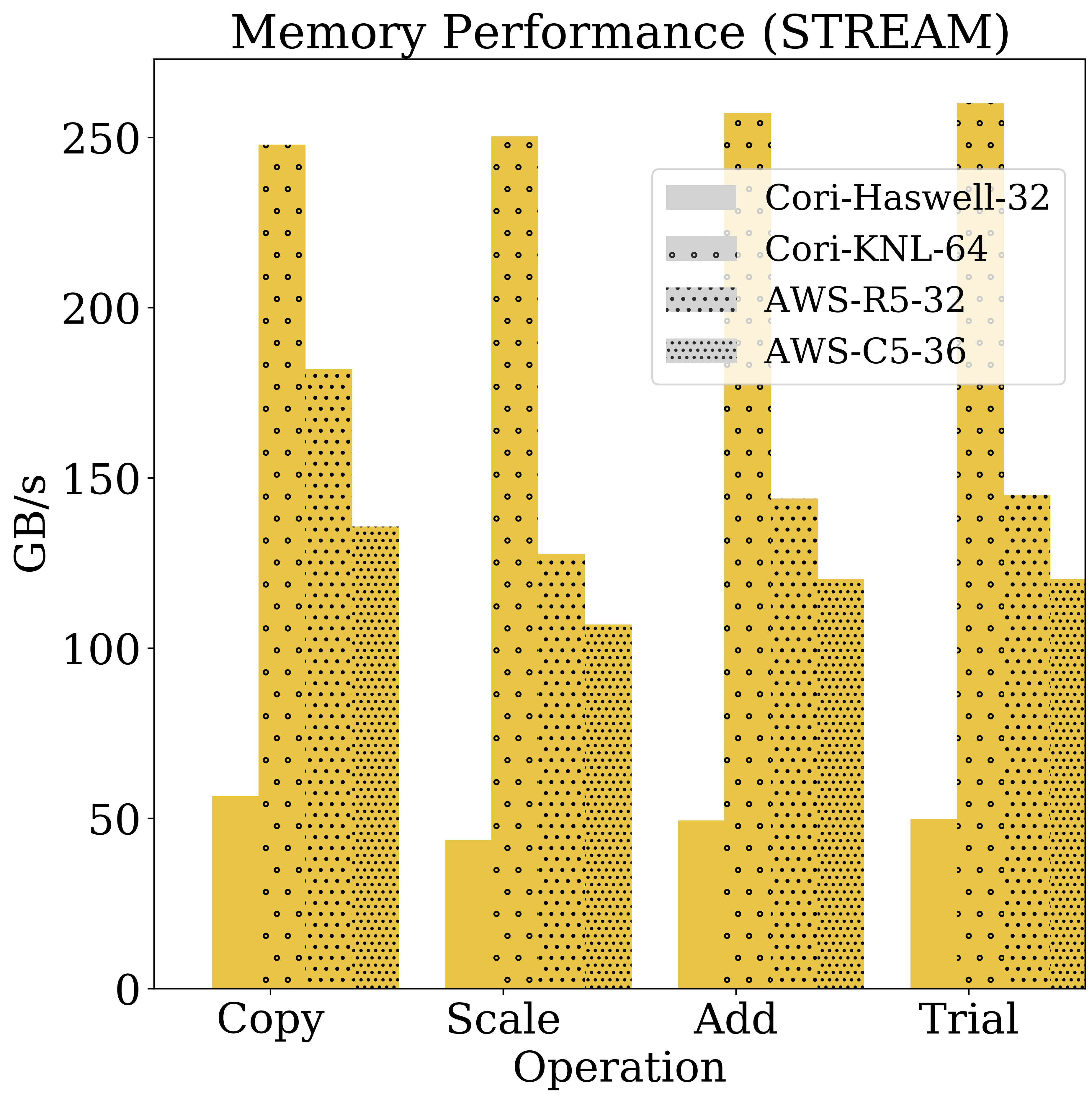
- Cori KNL has the higher memory bandwidth thanks to the on-chip multi-channel DRAM chip of 16GB
- If no on-chip memory, cloud instances show higher memory bandwidth than Cori Haswell



A Hardware and System View

Memory Bandwidth: STREAM benchmark

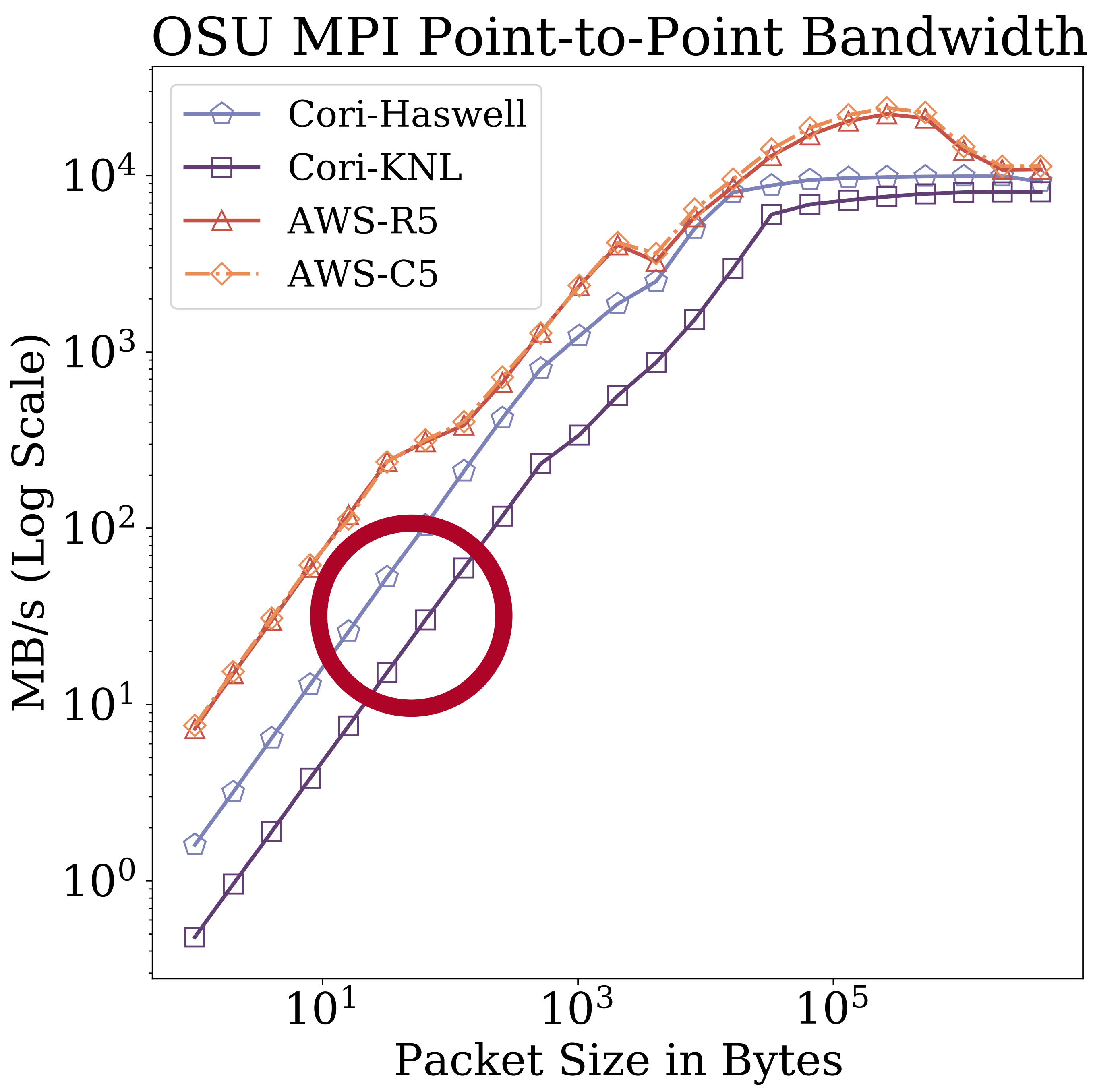
- Cori KNL has the higher memory bandwidth thanks to the on-chip multi-channel DRAM chip of 16GB
- If no on-chip memory, cloud instances show higher memory bandwidth than Cori Haswell

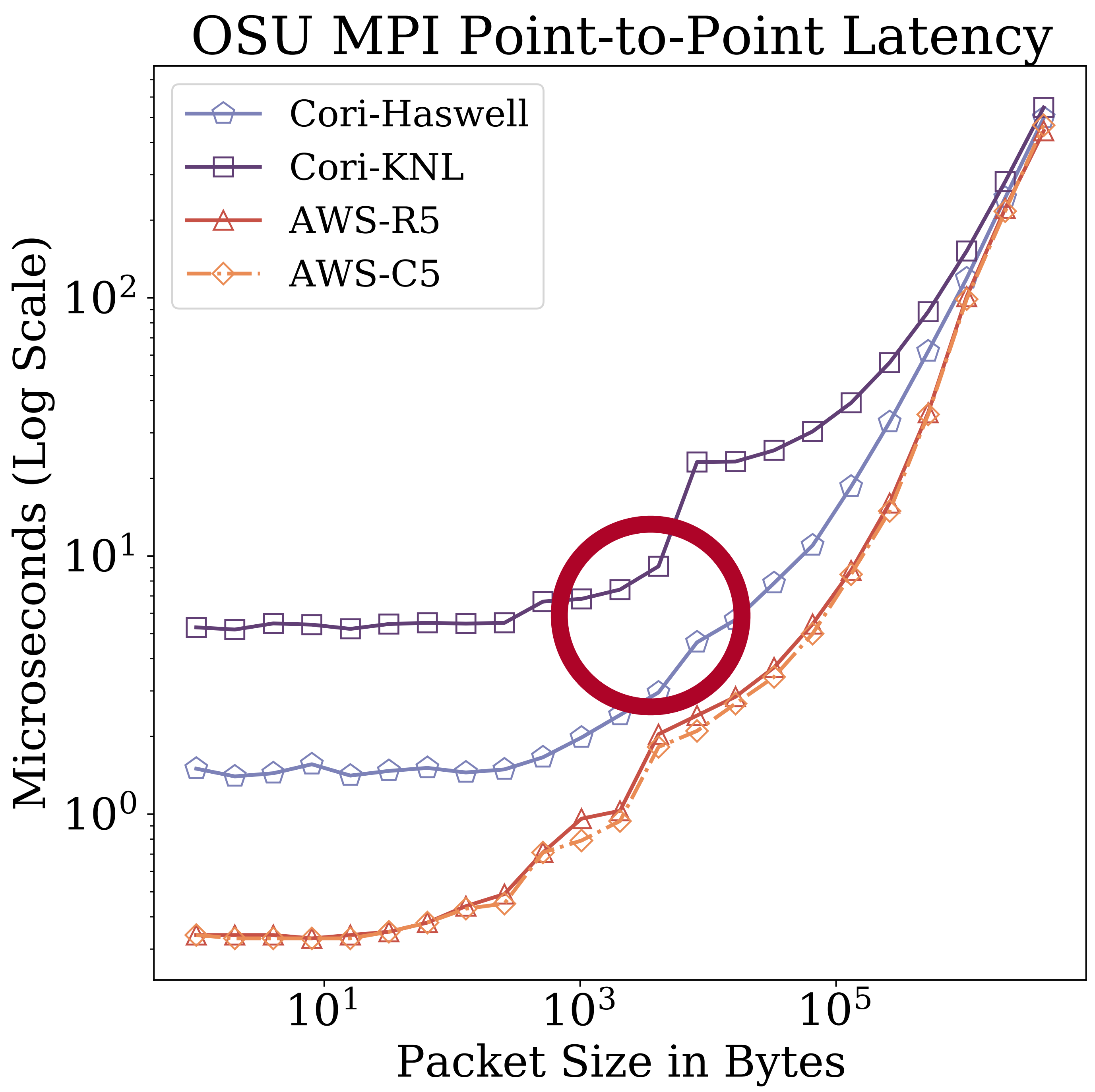


Take away: A faster hardware turnaround could benefit both compute-intensive workload and data-intensive ones

A Hardware and System View

Inter-Node Communication: OSU microbenchmark

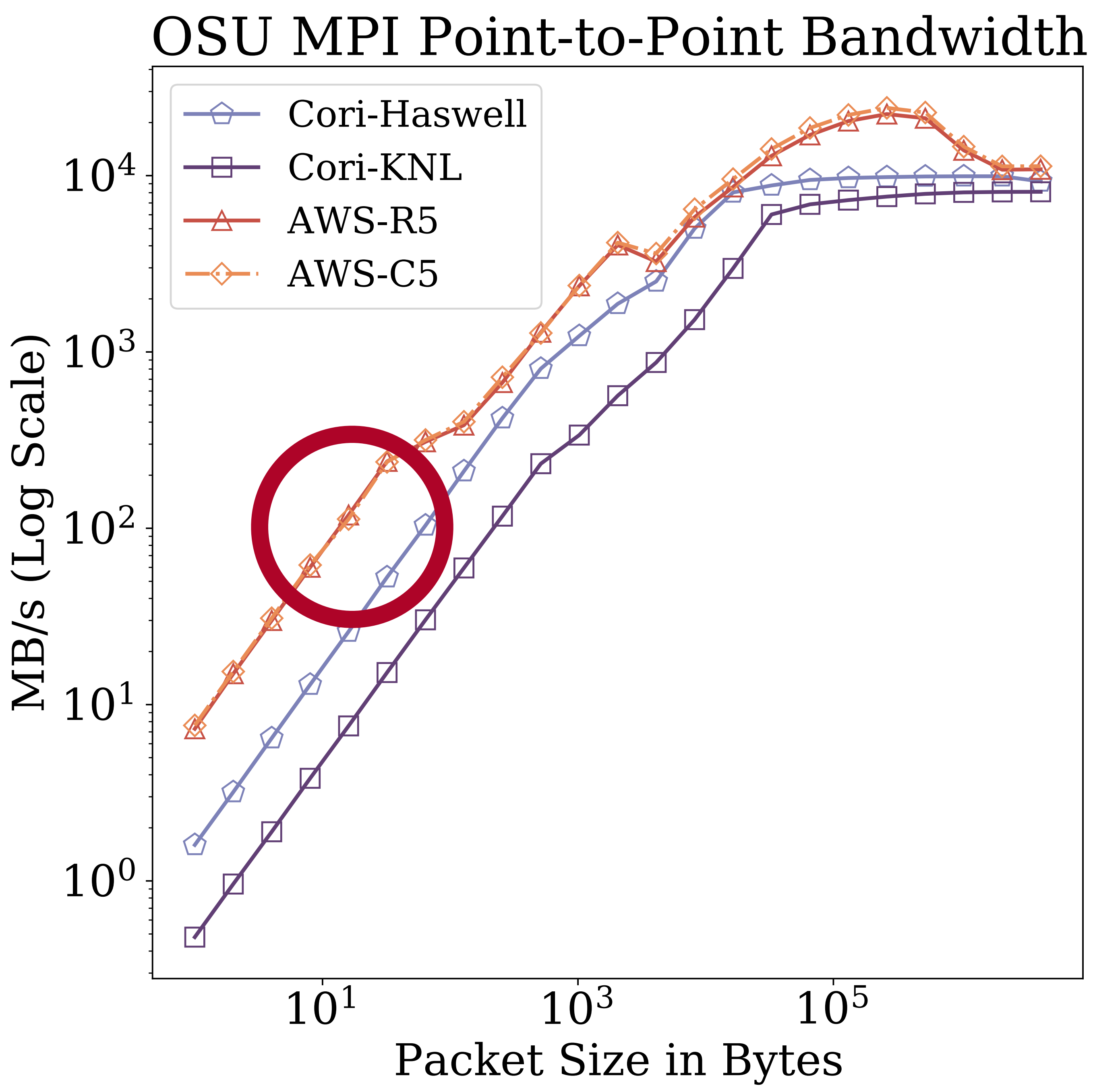


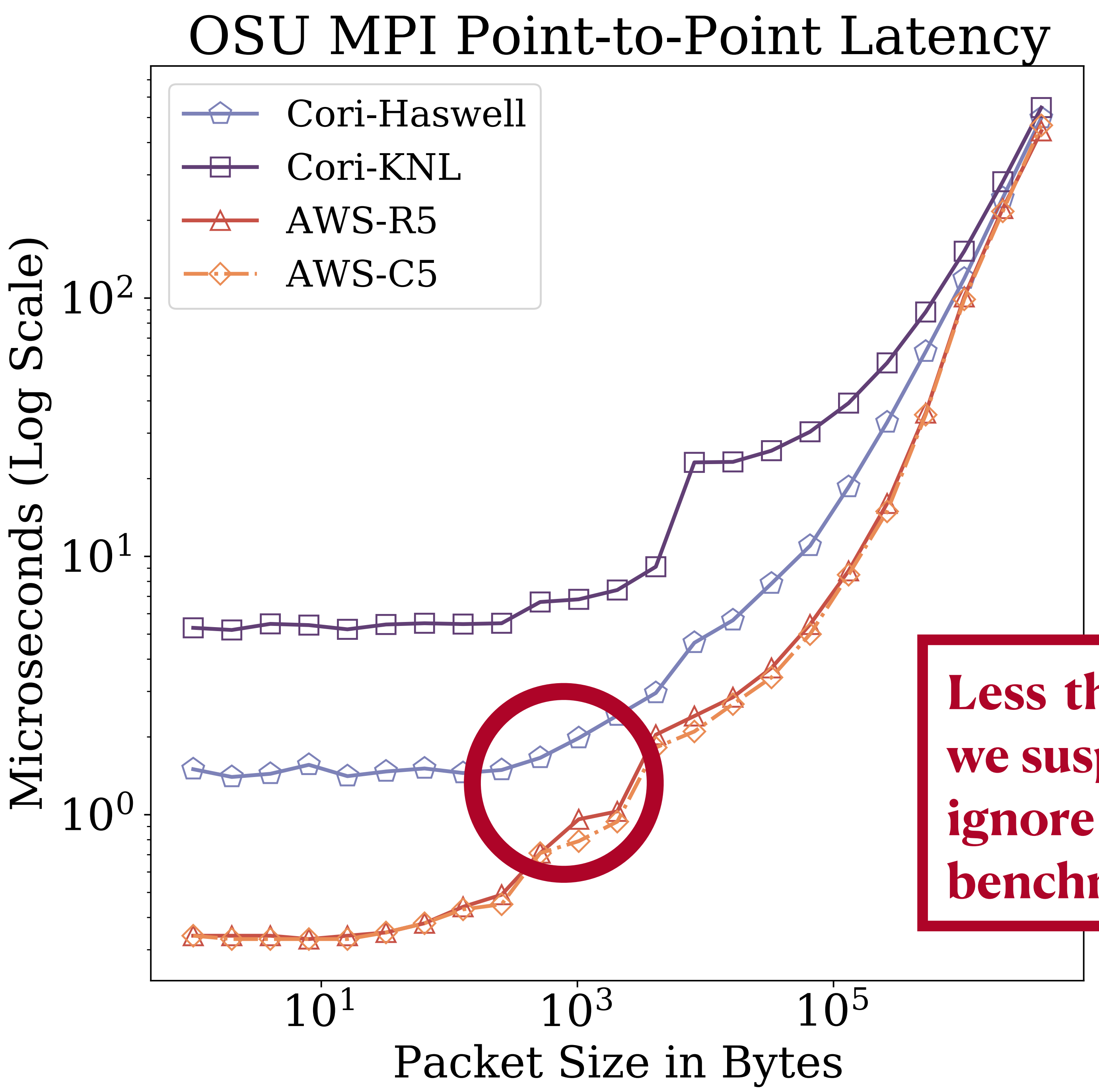


- Cori systems share the same network, however, the overhead of MPI penalizes lower frequency Cori KNL cores

A Hardware and System View

Inter-Node Communication: OSU microbenchmark



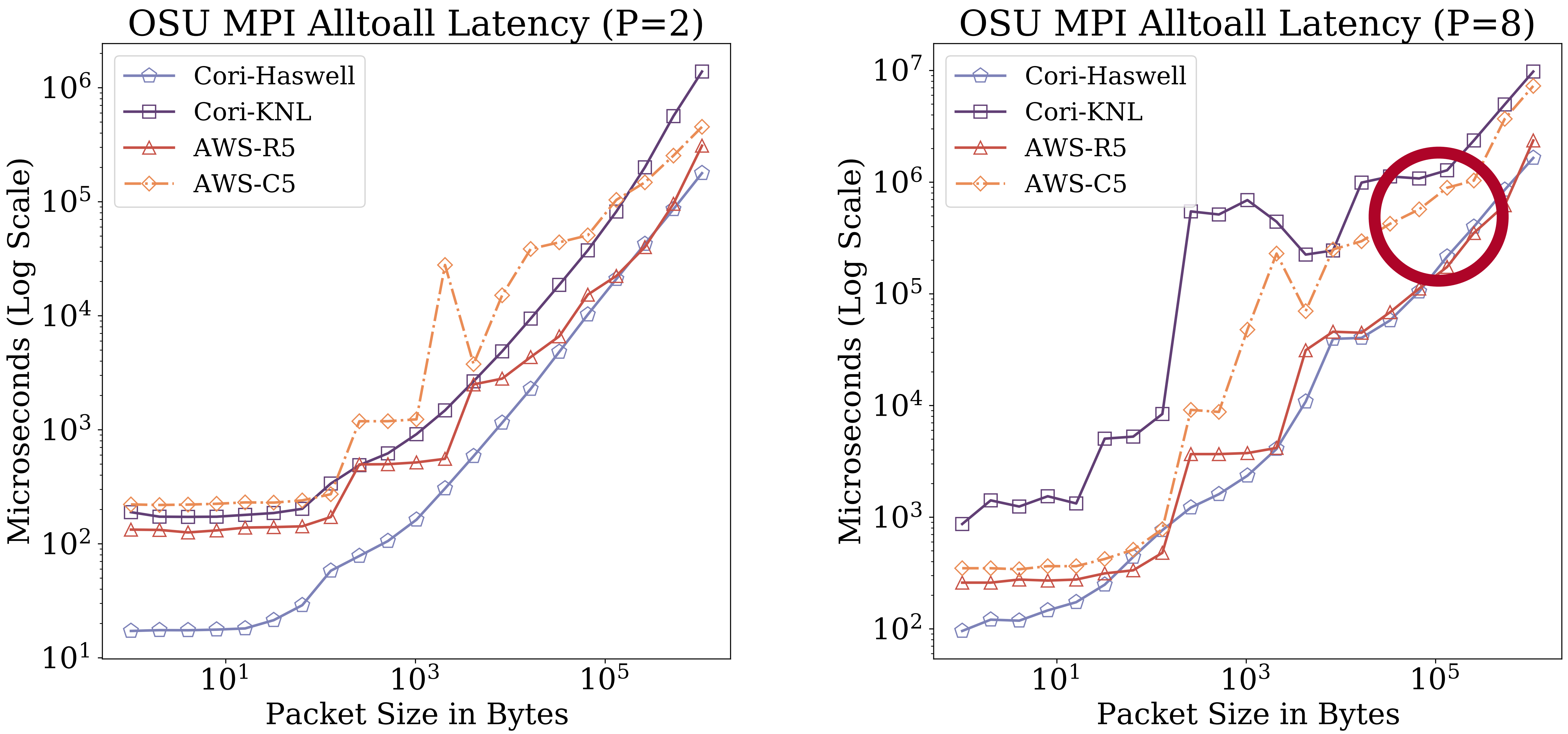


Less than 1us is a very low latency, we suspect that the scheduler could ignore our command and run the benchmark within a node

- Cori systems share the same network, however, the overhead of MPI penalizes lower frequency Cori KNL cores
- Cloud instances outperform HPC systems in both bandwidth and latency

A Hardware and System View

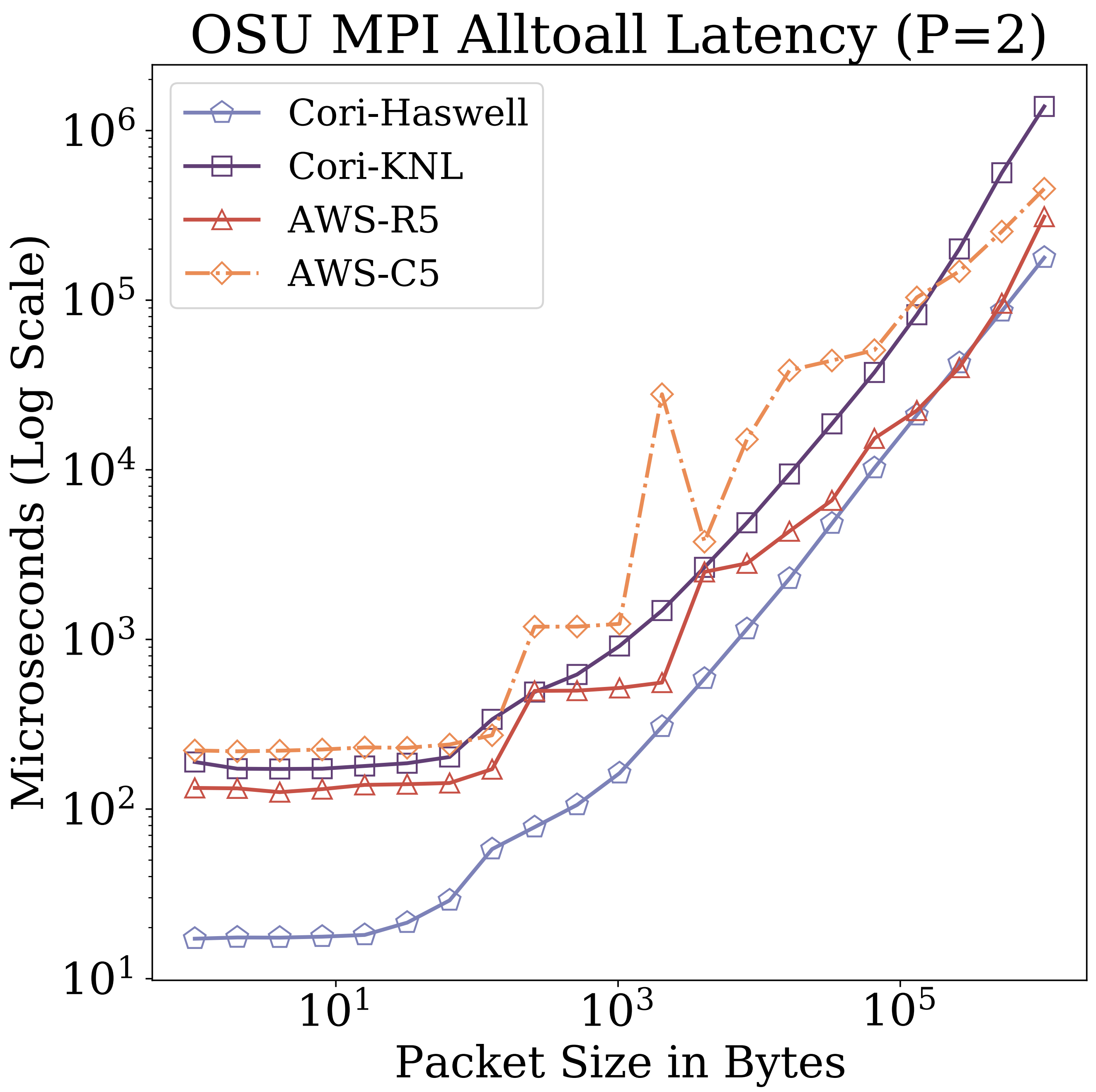
Inter-Node Communication: OSU microbenchmark

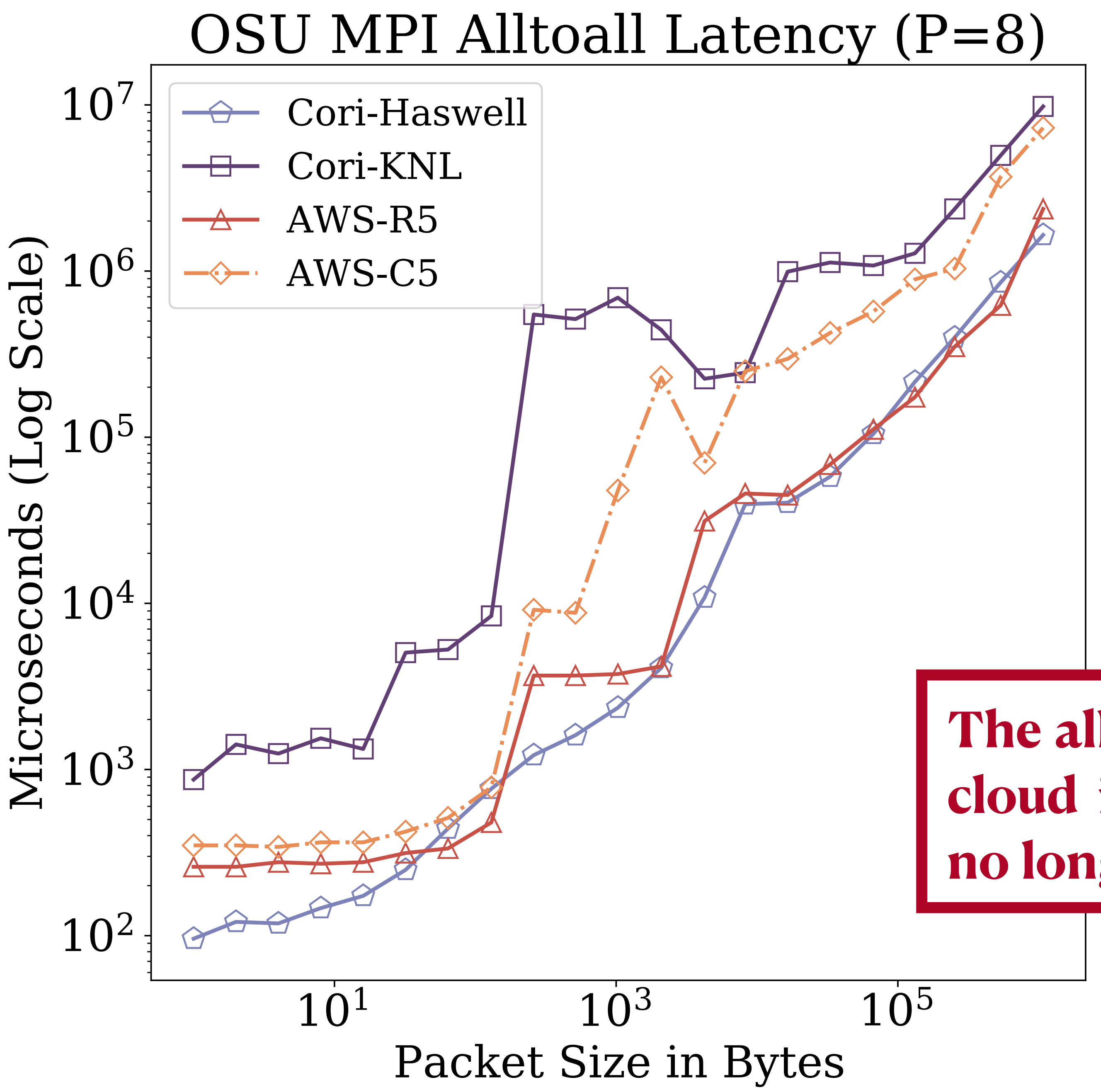


- Cori Haswell dominates on P = 2 and small message sizes; the **gap decreases** as the number of nodes is increased
- Cloud instances gain performance and the gap decreases as the number of nodes increases
- C5 could suffer of network contention being a compute-optimized instance with lower advertised bandwidth

A Hardware and System View

Inter-Node Communication: OSU microbenchmark



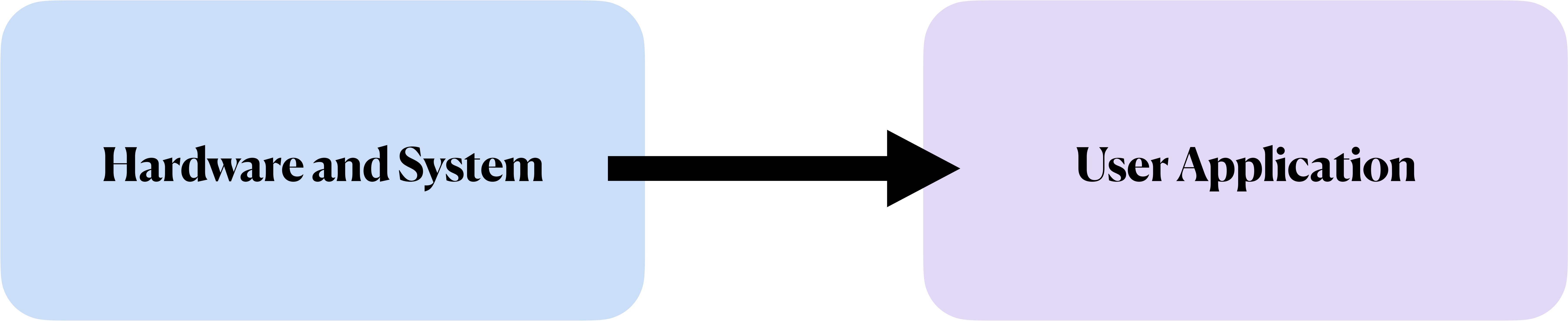


The all-to-all was a big issue for the cloud in previous studies, but that no longer seems to be the case

- Cori Haswell dominates on P = 2 and small message sizes; the **gap decreases** as the number of nodes is increased
- Cloud instances gain performance and the gap decreases as the number of nodes increases
- **Take away**: Communication-intensive applications have not benefited from cloud computing in the past because of their bandwidth requirement, **but that may now change**

Take Away So Far

Based on our microbenchmarks, **Cori Haswell and AWS R5 are comparable**, followed by AWS C5, with Cori KNL having overall the lowest performance



A User Application View

Application Overview

These applications have been chosen as extremes in scientific computation (used in previous literature as well):

N-Body Simulation

- **C**++
- It is **nearly O(*n*)** where *n* is the number of particles
- It uses **all-to-all communication** using MPI ISend/IRecv
- **Low communication-to-computation ratio**
- In-house implementation

Fast Fourier Transform (FFT)

- **C**
- It is **O(*n*log*n*)** where *n* is the size of the FFT
- It uses a **butterfly communication** using MPI Send/Recv
- **High communication-to-computation ratio**
- Frigo and Johnsons: Fast Fourier Transform in the West (FFTW)

FFTW ran multiple instances of FFT and **chose the best performing implementation** —The same on both Cori and AWS

A User Application View
Application Overview

These applications have been chosen as extremes in scientific computation (used in previous literature as well):

N-Body Simulation

- **C**++
- It is **nearly O(*n*)** where *n* is the number of particles
- It uses **all-to-all communication** using MPI ISend/IRecv
- **Low communication-to-computation ratio**
- In-house implementation

Fast Fourier Transform (FFT)

- **C**
- It is **O(*n*log*n*)** where *n* is the size of the FFT
- It uses a **butterfly communication** using MPI Send/Recv
- **High communication-to-computation ratio**
- Frigo and Johnsons: Fast Fourier Transform in the West (FFTW)

Future work: Extend the spectrum of applications to reflect today's diverse workloads

A User Application View

Serial Performance

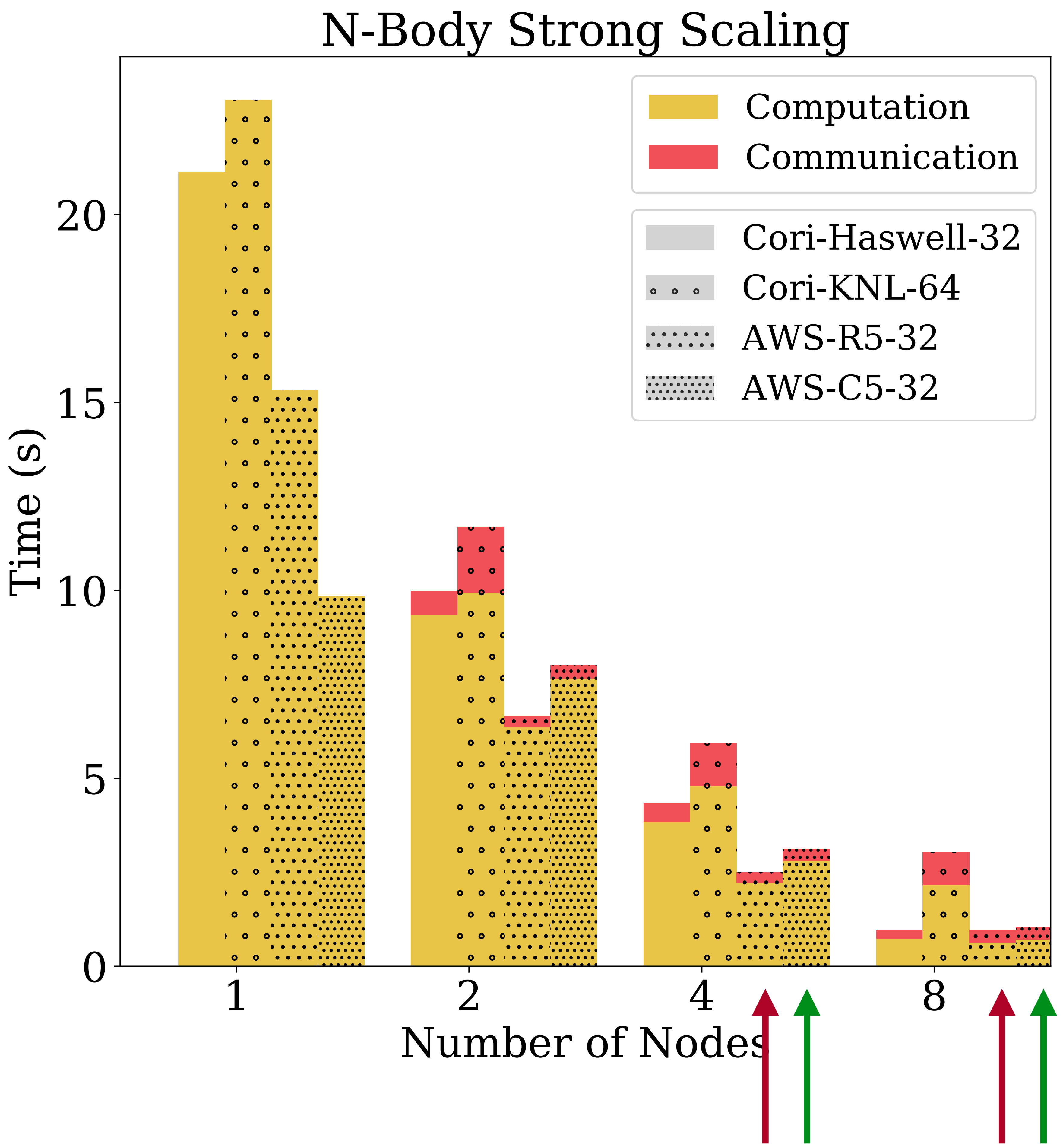
N-Body Simulation:

	Platform	Instruction (G)	Page Fault (K)	Cache Miss (M)	Time (s)
HPC	Cori Haswell	414.7	367.2	11,347.8	461.7
HPC	Cori KNL	415.4	367.4	11,220.1	1,736.5
Cloud	AWS r5dn.16xlarge (R5)	-	367.2	-	486.9
Cloud	AWS c5.18xlarge (C5)	427.2	367.2	21,457.4	480.6

Fast Fourier Transform (FFT):

	Platform	Instruction (G)	Page Fault (K)	Cache Miss (M)	Time (s)
HPC	Cori Haswell	782.1	9,766.8	871.5	312.4
HPC	Cori KNL	784.9	9,766.8	20,915.0	2,348.1
Cloud	AWS r5dn.16xlarge (R5)	-	9,767.5	-	303.3
Cloud	AWS c5.18xlarge (C5)	1,097.9	9,766.6	2,953.6	335.8

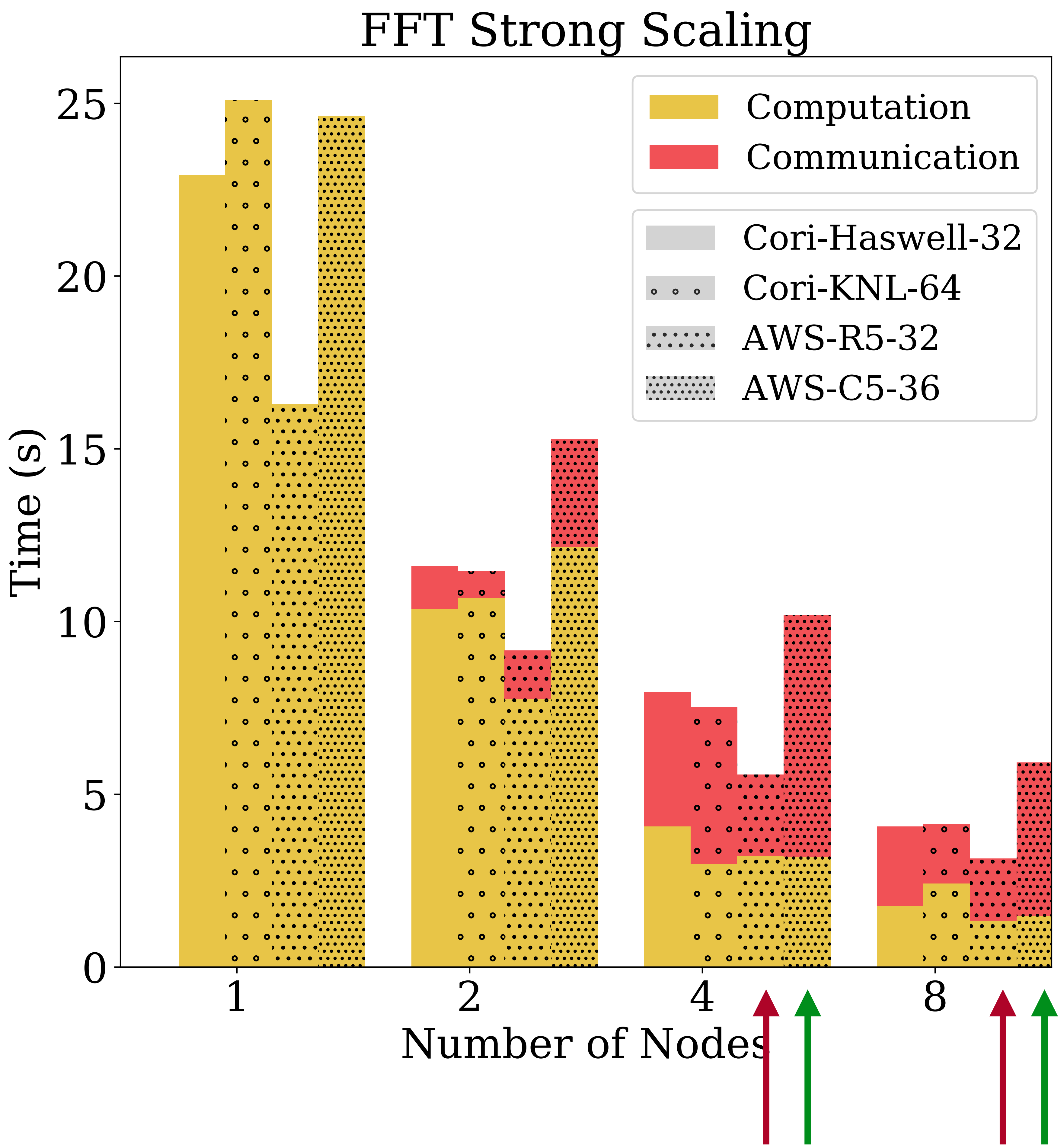
A User Application View
N-Body Parallel Performance



Cloud confirmed itself
suitable for compute-intensive applications

A User Application View

FFT Parallel Performance

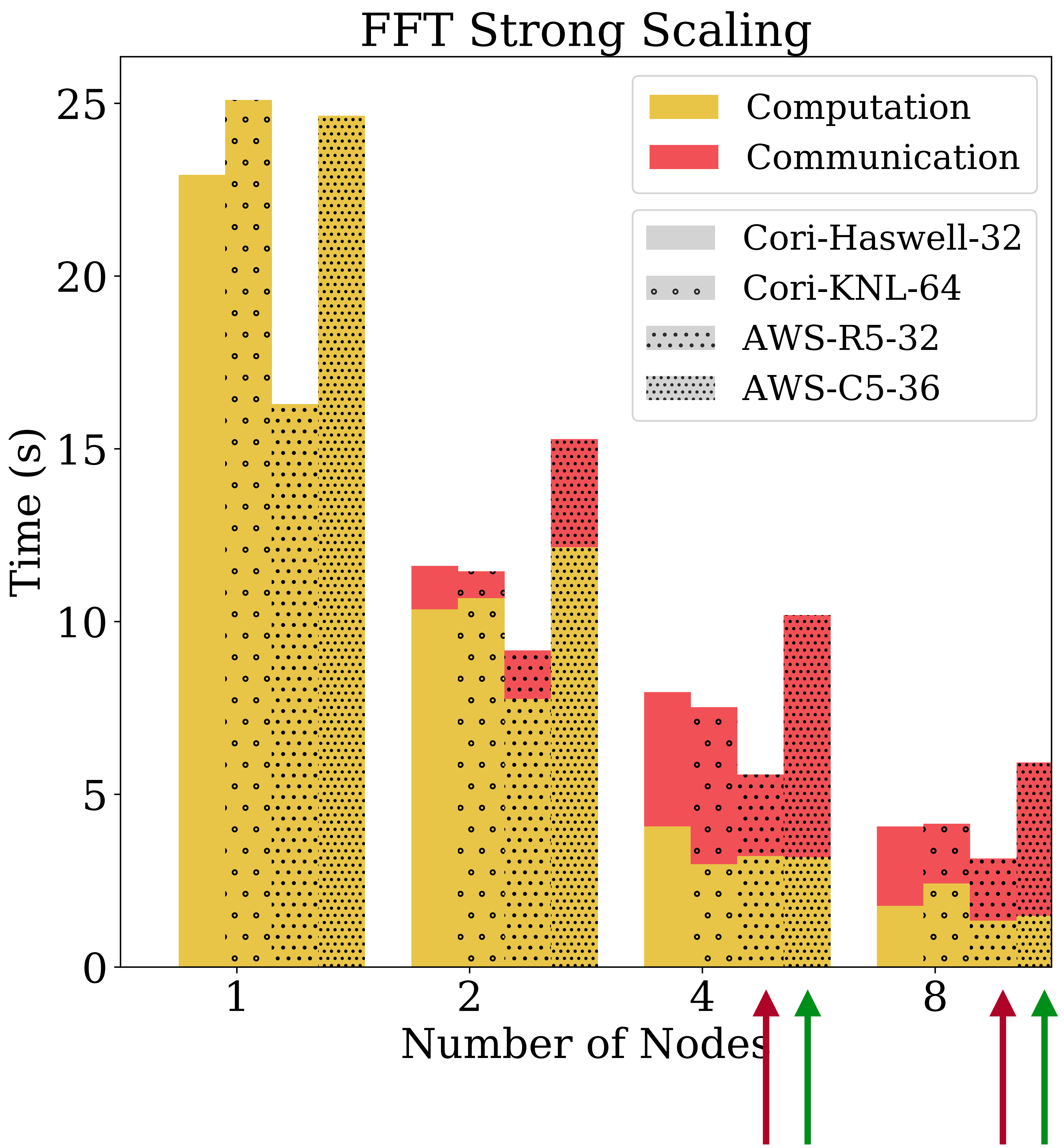


Cloud proved itself

competitive for communication-intensive applications

A User Application View

FFT Parallel Performance

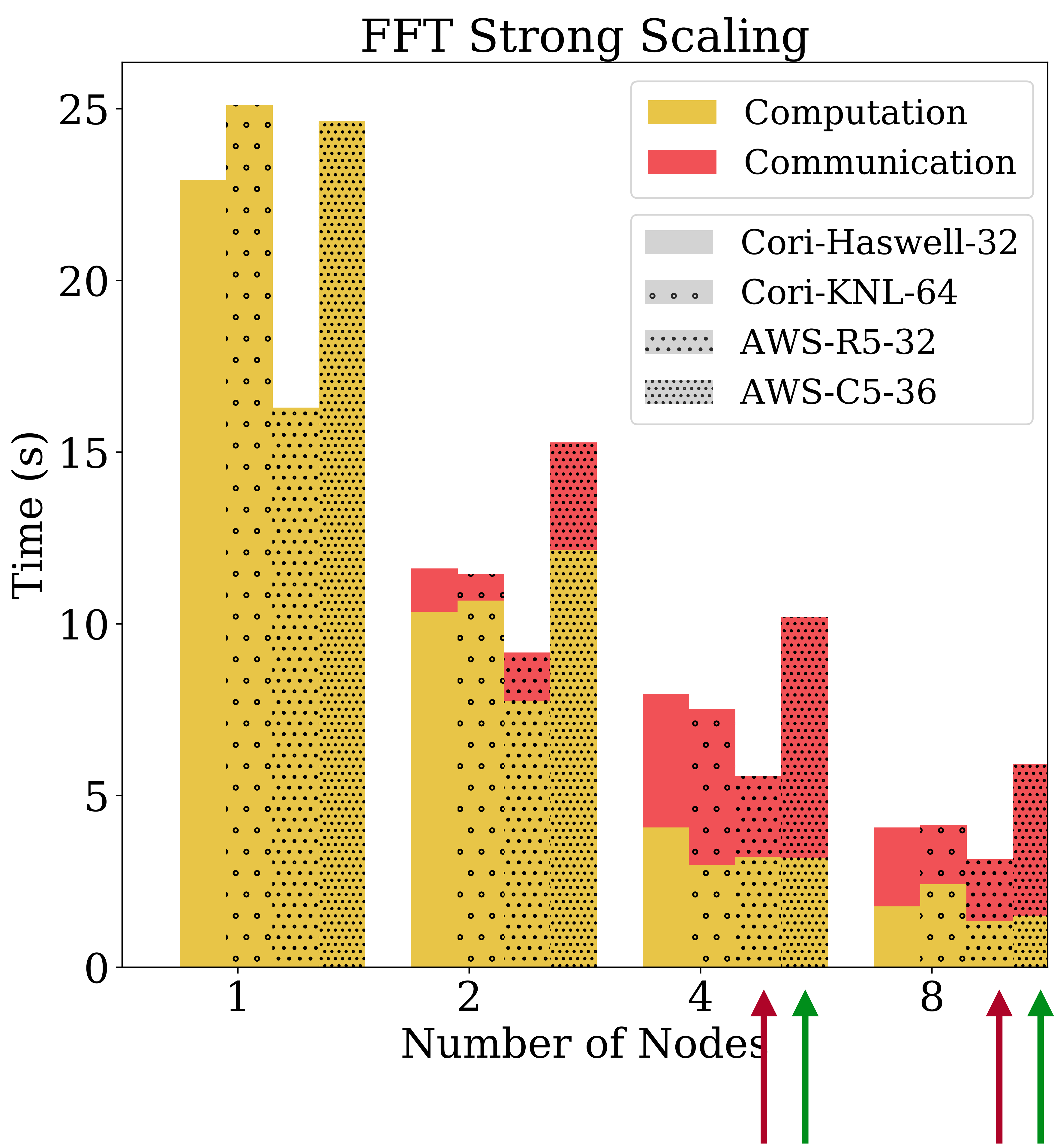


These results overturn historical ones!

In 2011, the Magellan report showed the **FFT was 52x slower on EC2** (AWS) than the used supercomputer on 8 nodes

A User Application View

FFT Parallel Performance

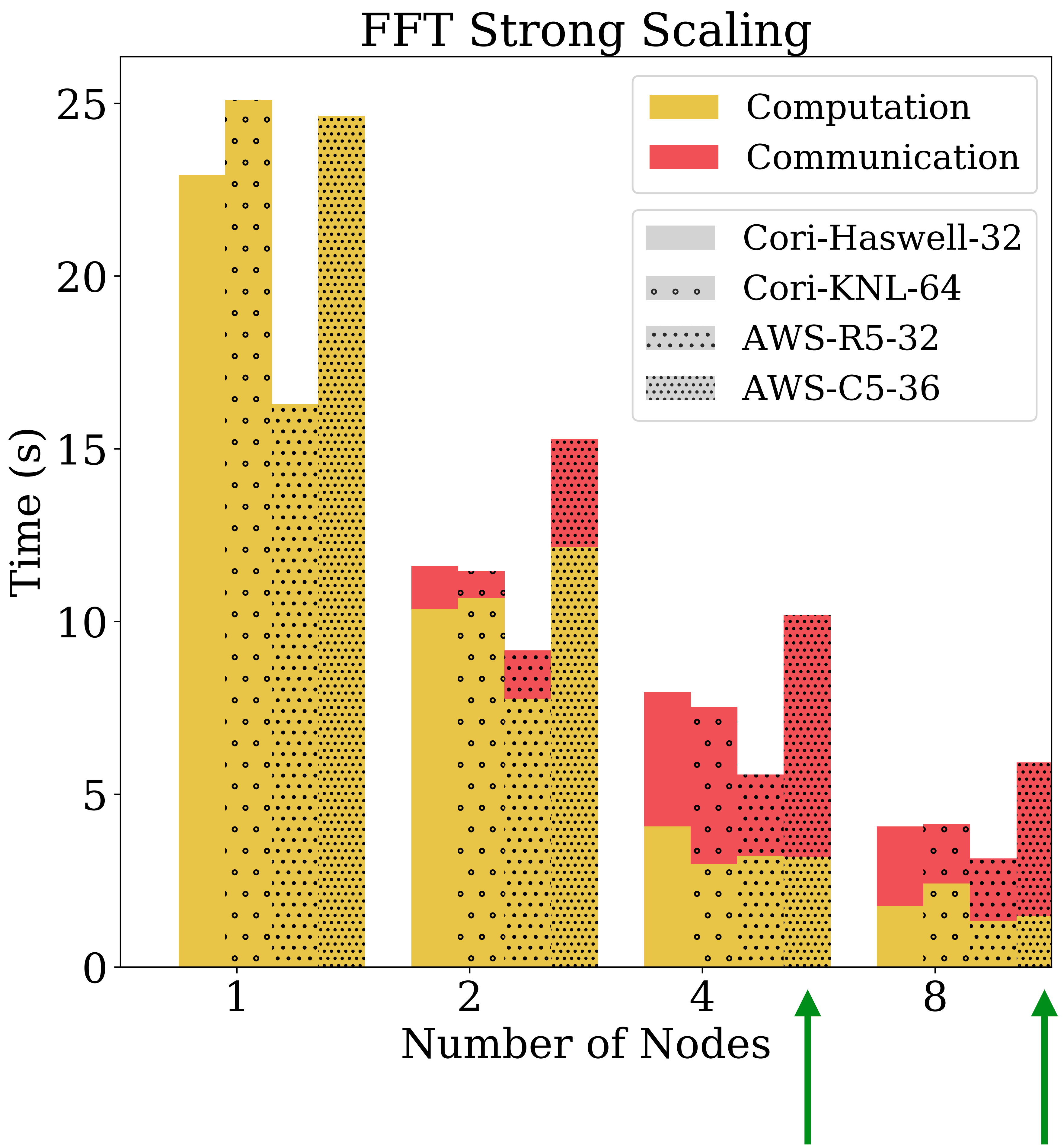


These cloud performance are <u>very good</u> in relation to the used supercomputer and to the historical results...

...but they could be ***better***

A User Application View

FFT Parallel Performance

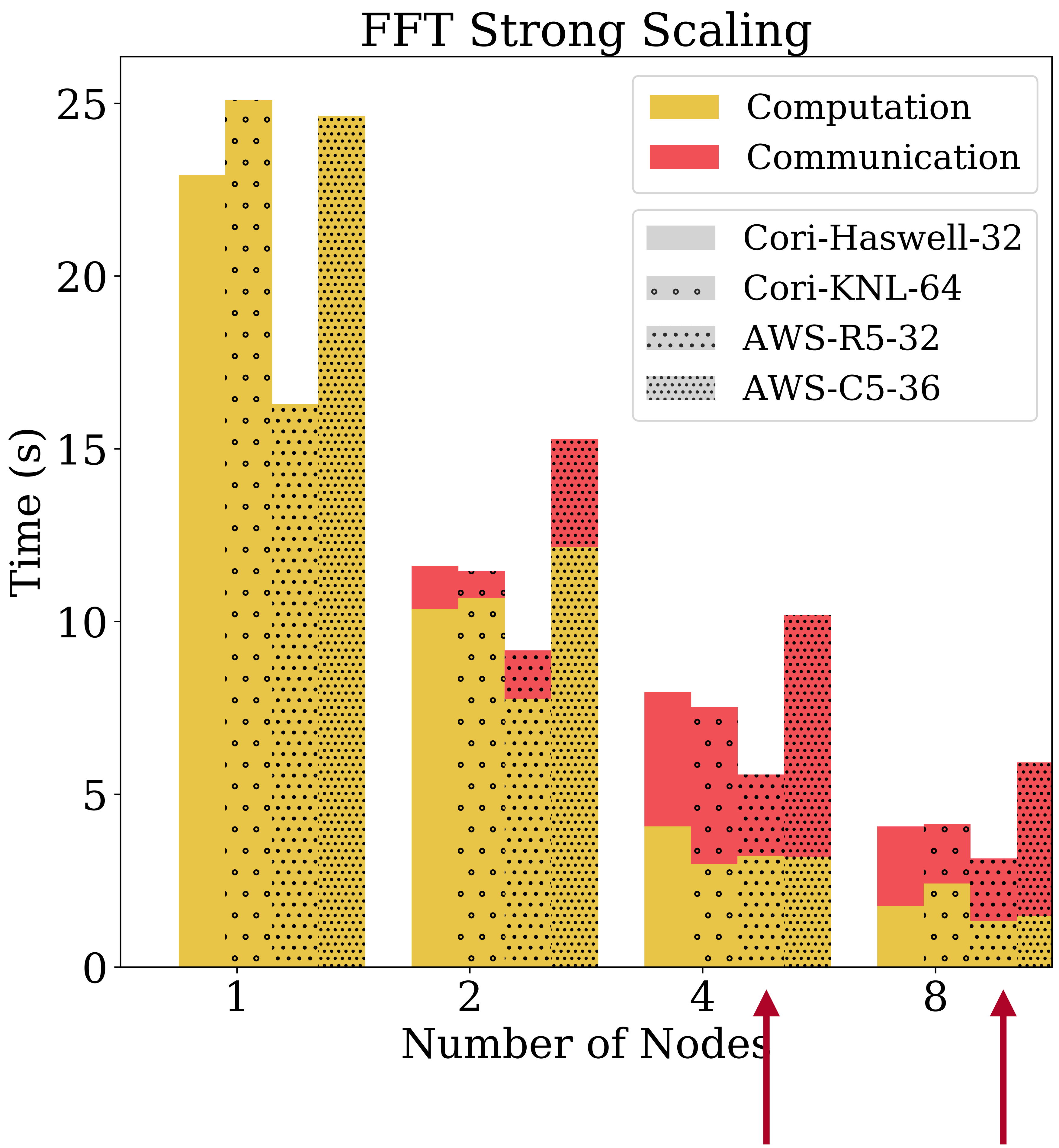


C5n.18xlarge uses **Amazon in-house Elastic Fabric Adapter (EFA)** as network interface offering **increased** performance

C5.18xlarge (in this study) does not use **EFA**

A User Application View

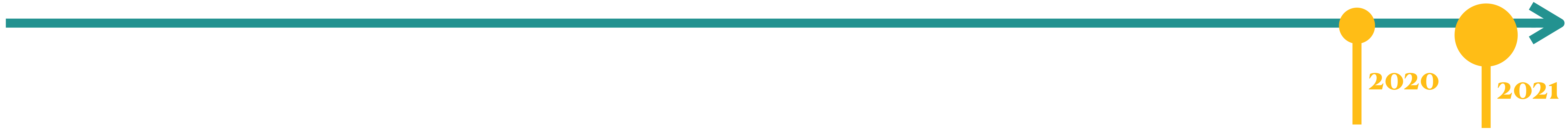
FFT Parallel Performance



R5dn.16xlarge is not the largest instance of its kind, r5dn.24xlarge is the largest (48 physical cores instead of 32)

The r5dn type of instance could achieve **ever greater** network performance if used in its entirety (100 Gbps instead of 75)

Take Away



That's what **this presentation** is about!

Our results show that:

- Cloud computing — AWS in this case — made **significant improvements in network performance**
- There's **no evidence** of significant virtualization overhead
- Cloud confirmed itself suitable for compute-intensive applications
- Cloud proved itself competitive for communication-intensive applications — **overturning historical results**

Limitations and future work:

- Run **larger scale experiments** up to hundreds of nodes
- Add workload and cloud provider **variety** (Google and IBM are interested in follow-up work on their cloud)
- Closer look at **performance variability**
- **How can we build better HPC systems** in the cloud (and not) starting from these results?

Acknowledgment

This work is supported by the Advanced Scientific Computing Research (ASCR) program within the Office of Science of the DOE under contract number DE-AC02-05CH11231. We used resources of the NERSC supported by the Office of Science of the DOE under Contract No. DEAC02-05CH11231. This research was also supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration. AWS Cloud Credits provided through the AWS Cloud Credits for Research program. Thanks to Rohan Bavishi, Rohan Padhye, Neesha Zerin, and James Demmel for useful suggestions and valuable discussions.