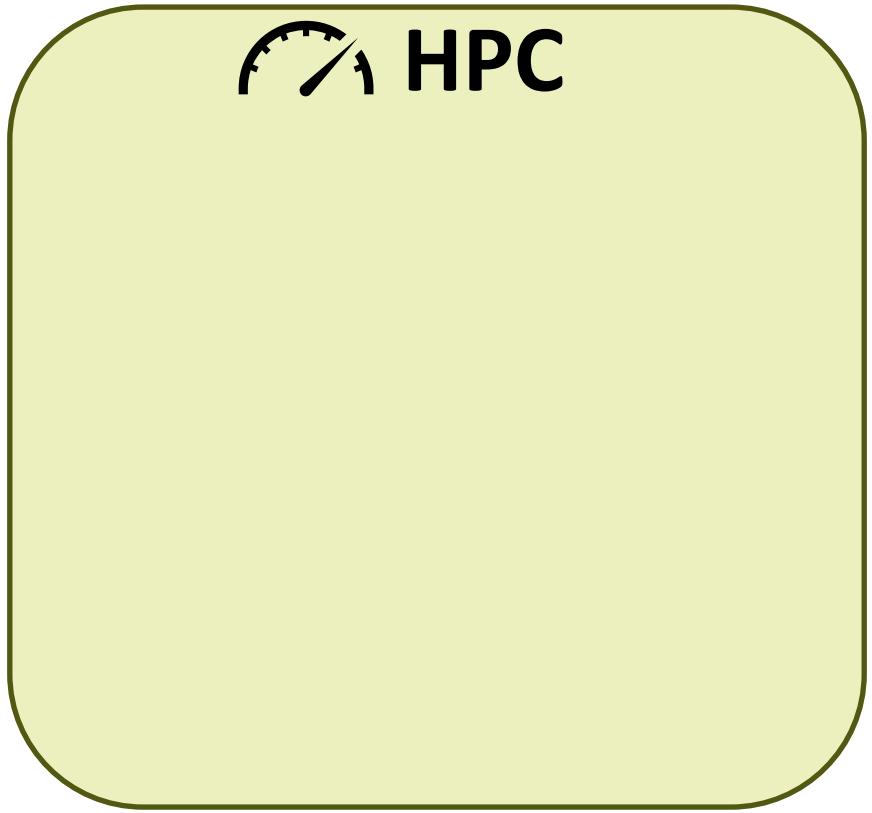
MARCIN COPIK

High Performance Serverless for HPC and Clouds

High-Performance Computing Systems

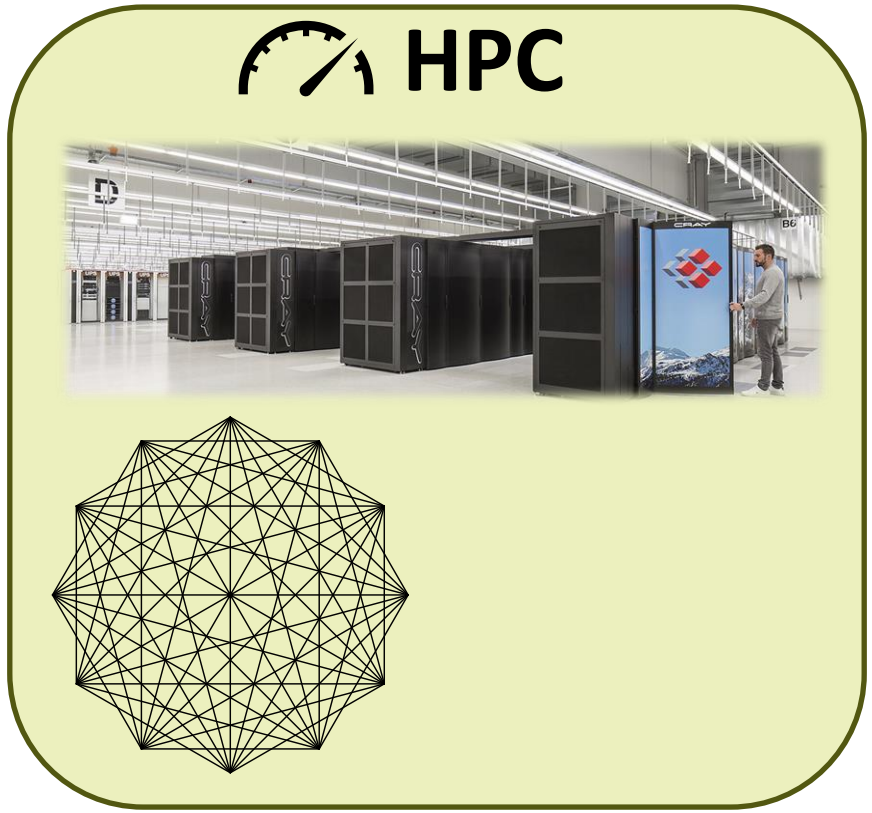
High-Performance Computing Systems



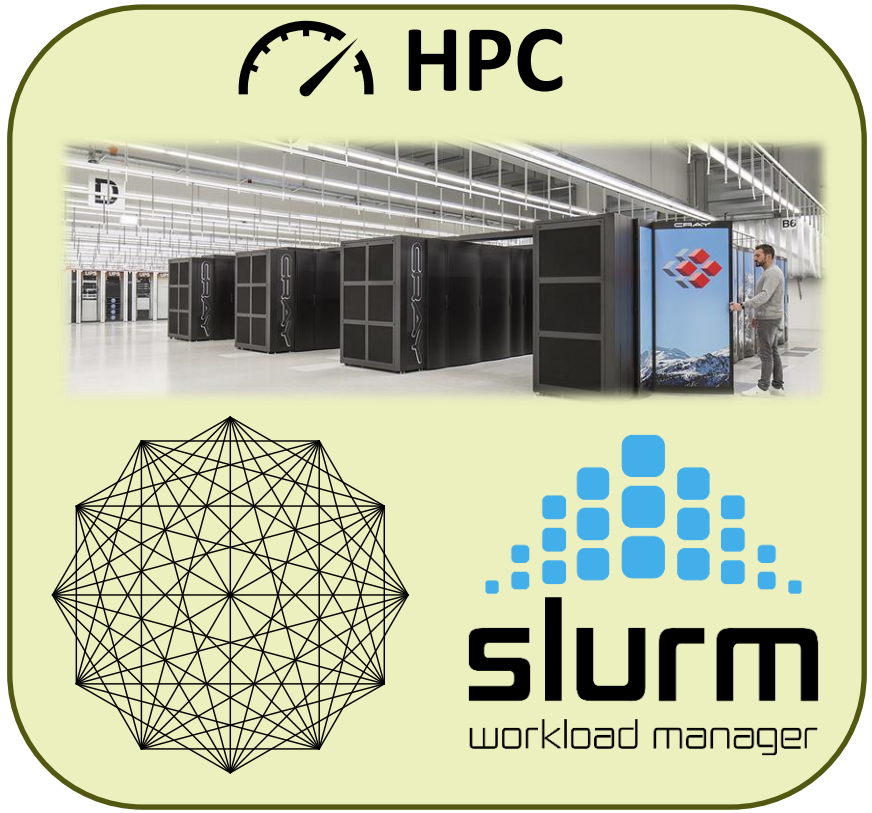
High-Performance Computing Systems

HPC

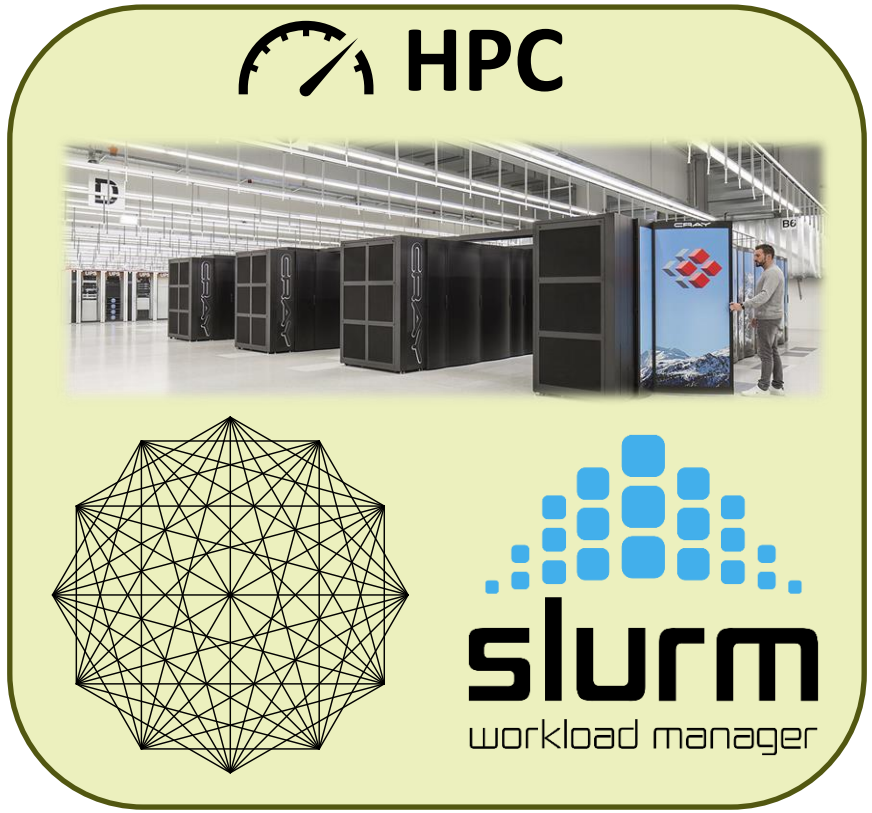
High-Performance Computing Systems



High-Performance Computing Systems



High-Performance Computing Systems

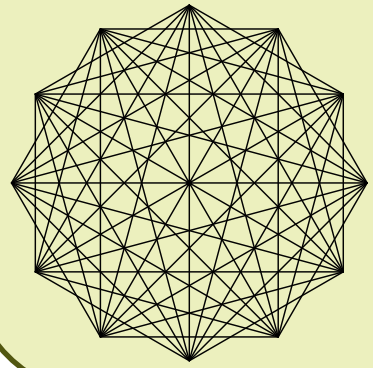


High-Performance Computing Systems

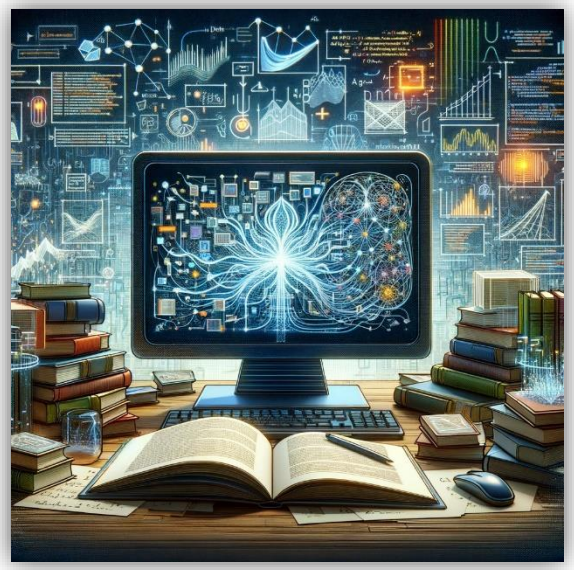
HPC

High-Performance Computing Systems

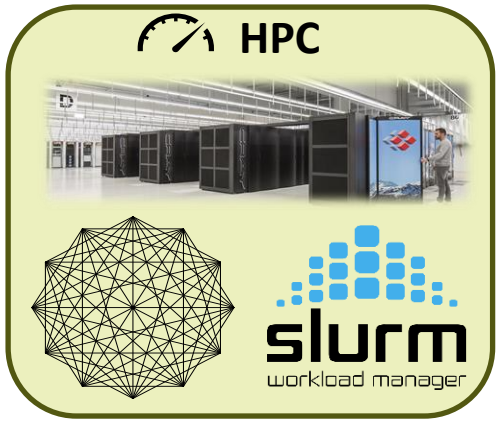
HPC



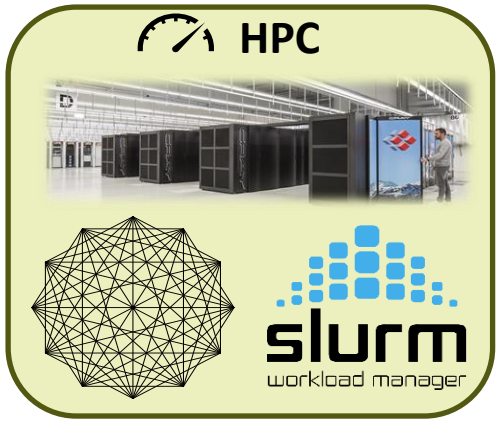
slurm workload manager

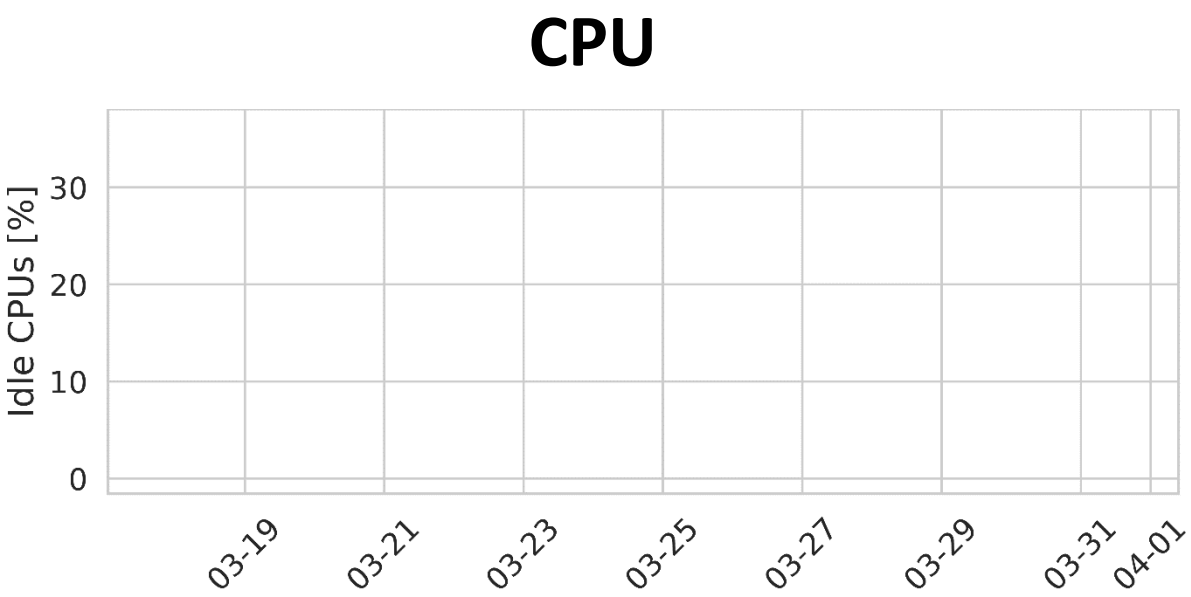


Tracking Wasted Resources in HPC

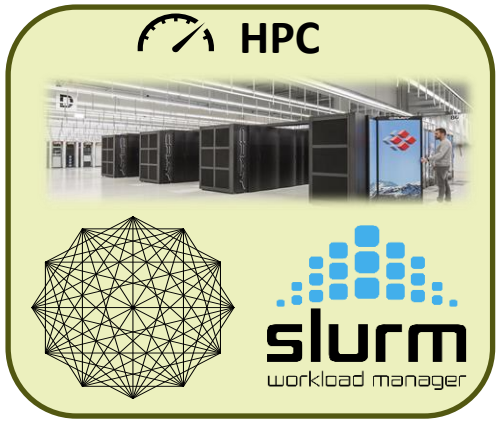


Tracking Wasted Resources in HPC

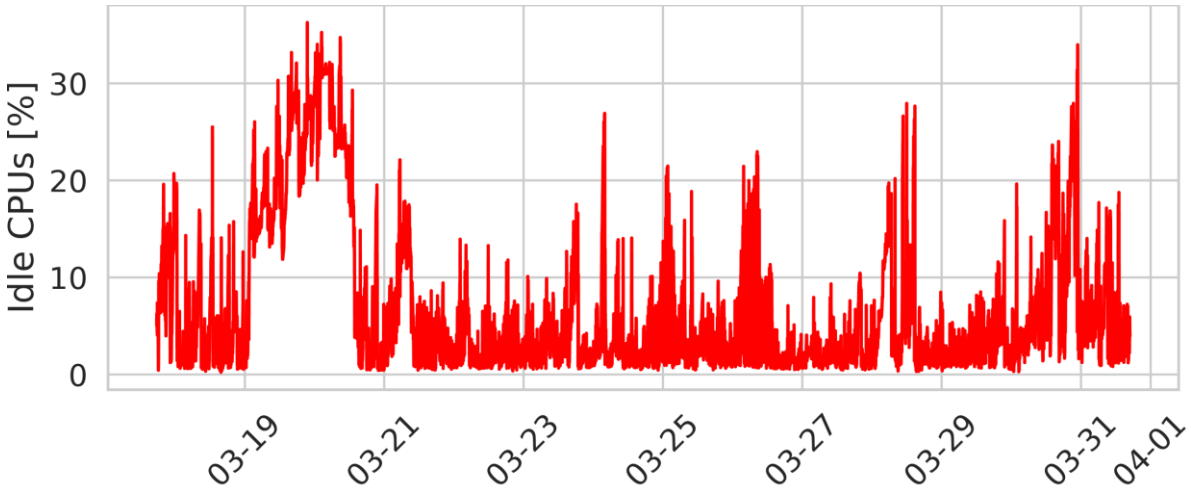




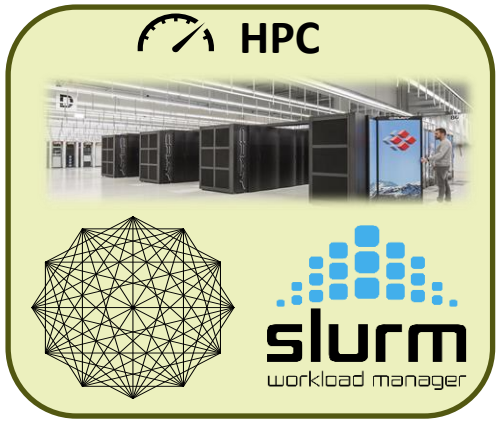
Tracking Wasted Resources in HPC



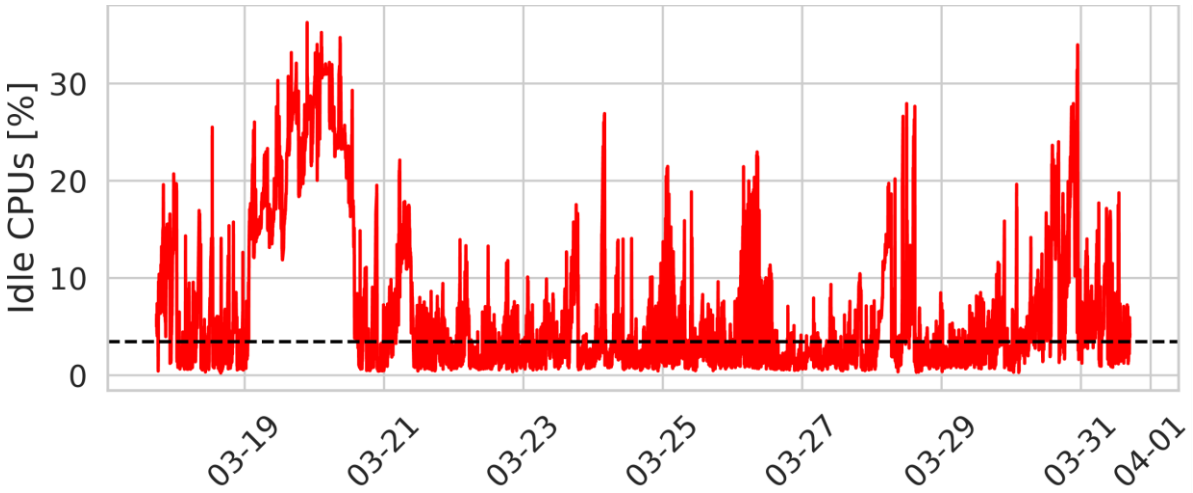
CPU



Tracking Wasted Resources in HPC

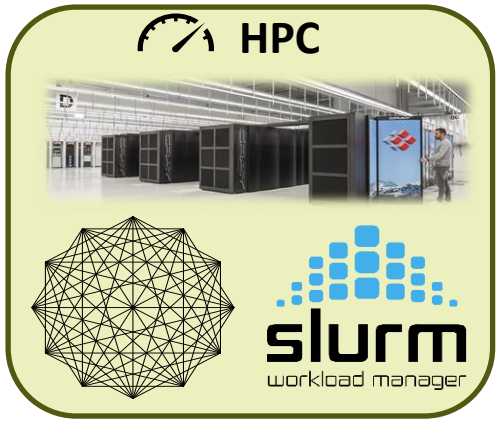


CPU

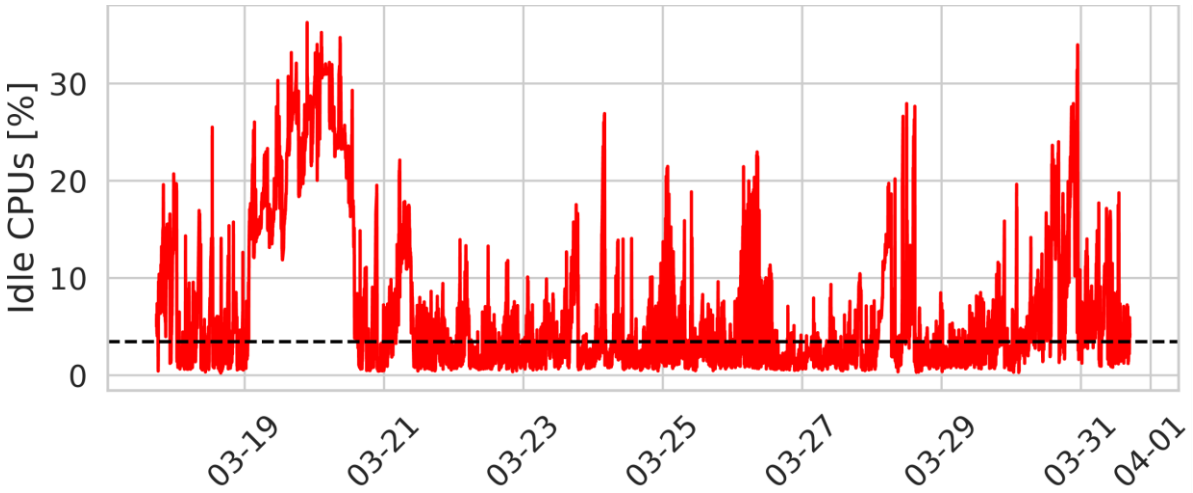


Mean idle CPUs: 6.6%

Tracking Wasted Resources in HPC

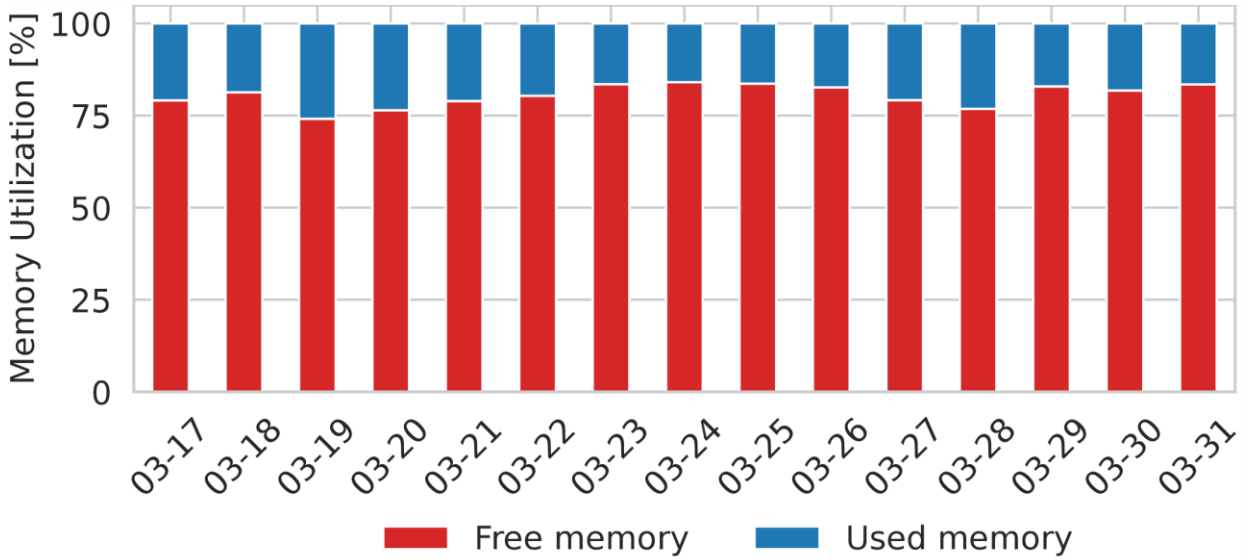


CPU



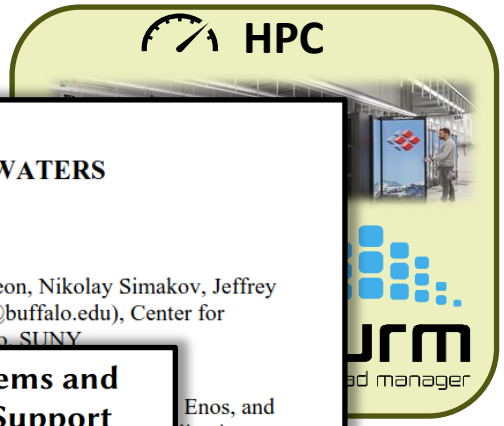
Mean idle CPUs: 6.6%

Memory



Mean free memory: 80.5%

Tracking Wasted Resources in HPC



Learning from Five-year Resource-Utilization Data of Titan System

Feiyi Wang*, Sarp Oral†, Satyabrata Sen ‡ and Neena Imam§

Oak Ridge National Laboratory

CLUSTER, 2019

FINAL REPORT

WORKLOAD ANALYSIS OF BLUE WATERS

(ACI 1650758)

Matthew D. Jones, Joseph P. White, Martins Innus, Robert L. DeLeon, Nikolay Simakov, Jeffrey T. Palmer, Steven M. Gallo, and Thomas R. Furlani (furlani@buffalo.edu), Center for Computational Research, University at Buffalo, SUNY

Enos, and lications

Xiv, 2017

Quantifying Memory Underutilization in HPC Systems and Using it to Improve Performance via Architecture Support

Gagandeep Panwar*
Virginia Tech
Blacksburg, USA
gpanwar@vt.edu

Da Zhang*
Virginia Tech
Blacksburg, USA
daz3@vt.edu

Yihan Pang*
Virginia Tech
Blacksburg, USA
pyihan1@vt.edu

Mai Dahshan
Virginia Tech
Blacksburg, USA
mdahshan@vt.edu

Nathan DeBardeleben
Los Alamos National Laboratory
Los Alamos, USA
ndebard@lanl.gov

Binoy Ravindran
Virginia Tech
Blacksburg, USA
binoy@vt.edu

Xun Jian
Virginia Tech
Blacksburg, USA
xunj@vt.edu

MICRO, 2019

A Case For Intra-rack Resource Disaggregation in HPC

GEORGE MICHELOGIANNAKIS, Lawrence Berkeley National Laboratory, USA
BENJAMIN KLENK, NVIDIA, USA
BRANDON COOK, Lawrence Berkeley National Laboratory, USA
MIN YE
LARRY D
KEREN
JOHN SH

Idle CPUs [%]

30
20
10
0

A Holistic View of Memory Utilization on HPC Systems: Current and Future Trends

Ivy B. Peng*
peng8@llnl.gov
Lawrence Livermore National Laboratory
USA

Ian Karlin
karlin1@llnl.gov
Lawrence Livermore National Laboratory
USA

Maya B. Gokhale
gokhale2@llnl.gov
Lawrence Livermore National Laboratory
USA

Kathleen Shoga
Shoga1@llnl.gov
Lawrence Livermore National Laboratory
USA

Matthew Legendre
legendre1@llnl.gov
Lawrence Livermore National Laboratory
USA

Todd Gamblin
gamblin2@llnl.gov
Lawrence Livermore National Laboratory
USA

MEMSYS, 2021

Comprehensive Workload Analysis and Modeling of a Petascale Supercomputer

Haihang You[1] and Hao Zhang[2]

[1] National Institute for Computational Sciences, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA
[2] Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, TN 37996, USA
{hyou,haozhang}@utk.edu

JSSPP, 2012

Tracking Wasted Resources in HPC

Tracking Wasted Resources in HPC

Static Jobs

slurm workload manager

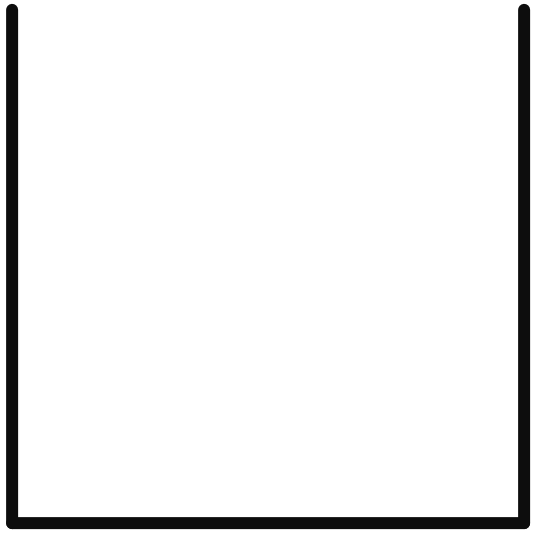
Rigid Scheduler

Tracking Wasted Resources in HPC

Static Jobs

slurm workload manager

Rigid Scheduler

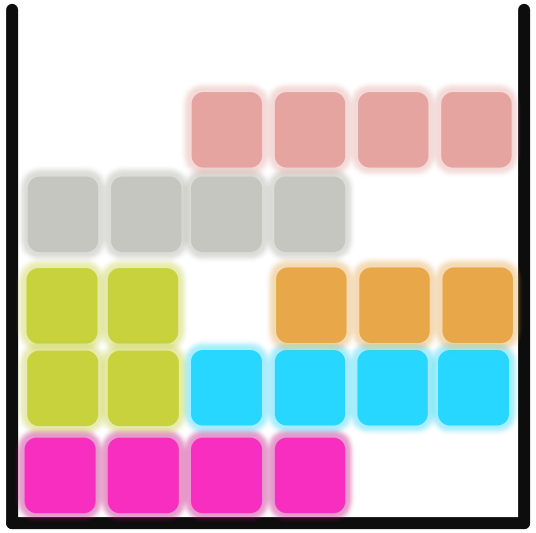


Tracking Wasted Resources in HPC

Static Jobs

slurm workload manager

Rigid Scheduler

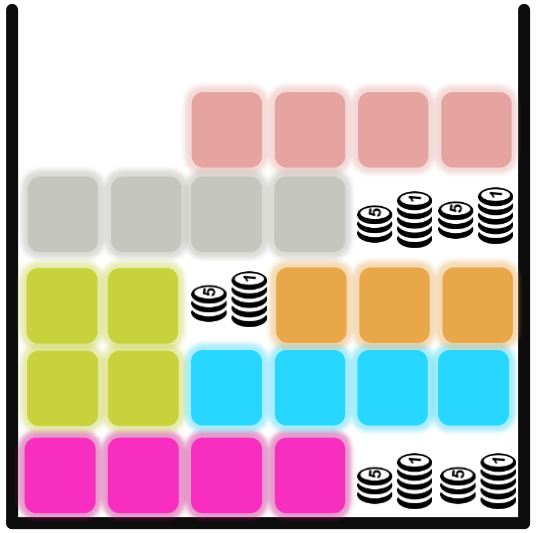


Tracking Wasted Resources in HPC

Static Jobs

slurm workload manager

Rigid Scheduler

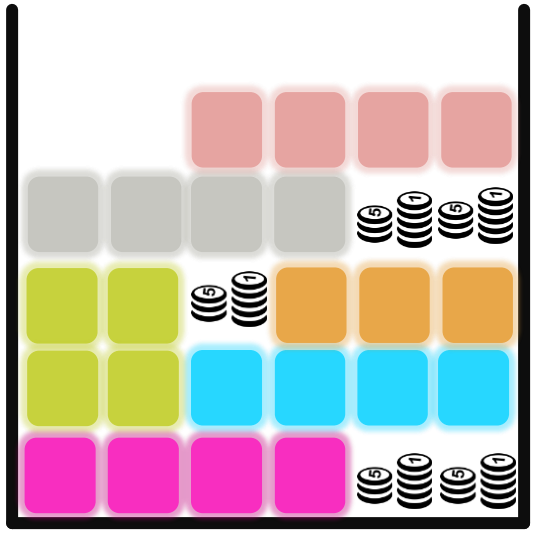


Tracking Wasted Resources in HPC

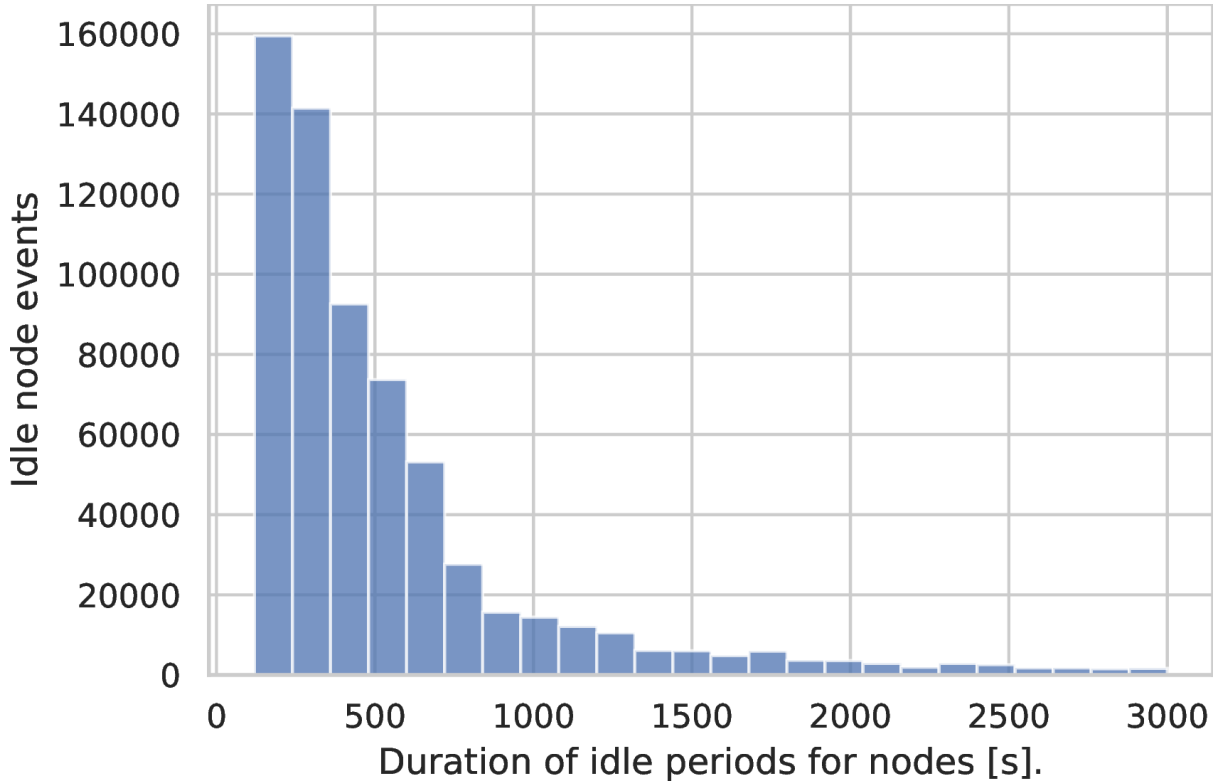
Static Jobs

slurm workload manager

Rigid Scheduler



Duration of Node Idleness

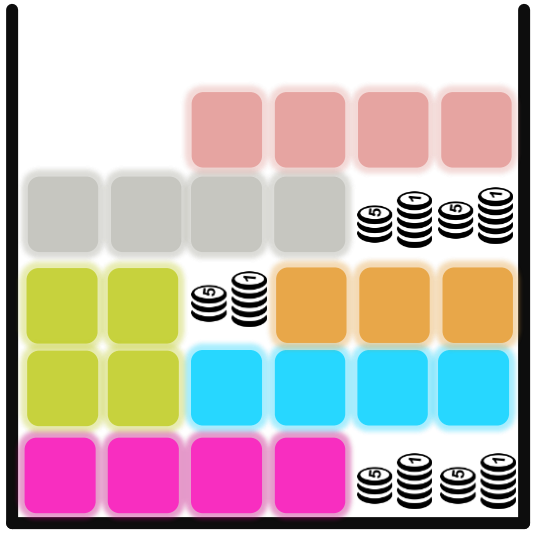


Tracking Wasted Resources in HPC

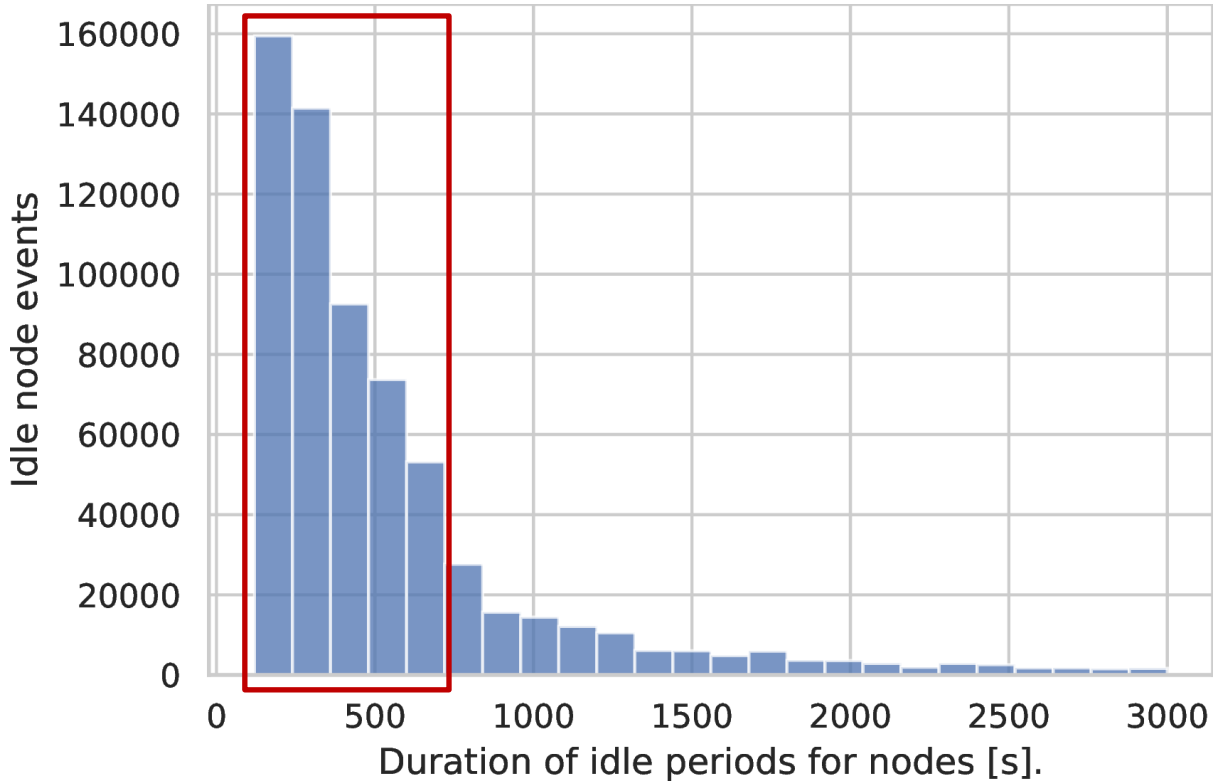
Static Jobs

slurm workload manager

Rigid Scheduler



Duration of Node Idleness



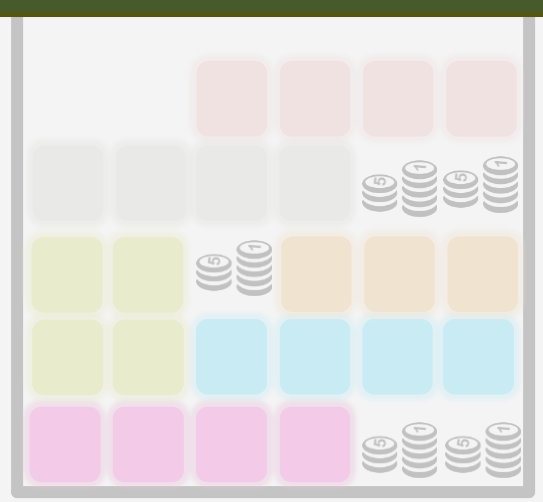
70% of idle node events last less than 10 minutes.

Tracking Wasted Resources in HPC

Static Jobs

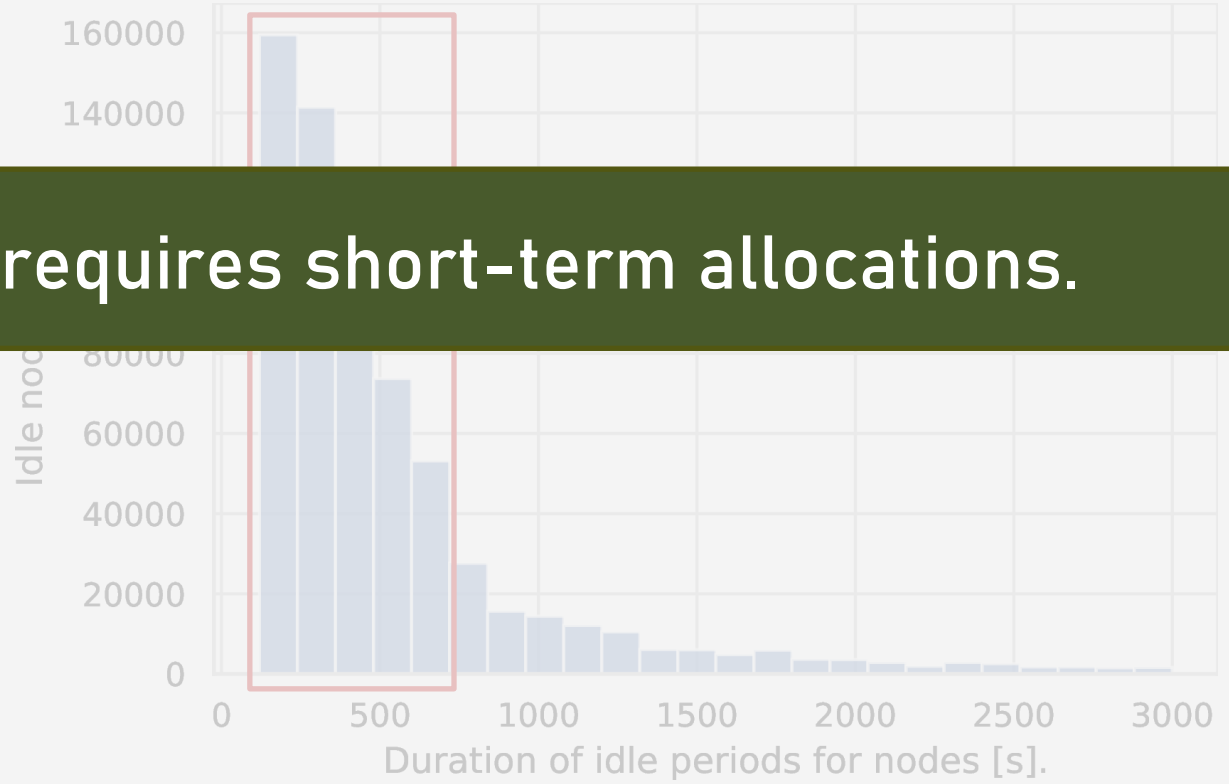
slurm workload manager

Rigid Scheduler



Short-term resource availability requires short-term allocations.

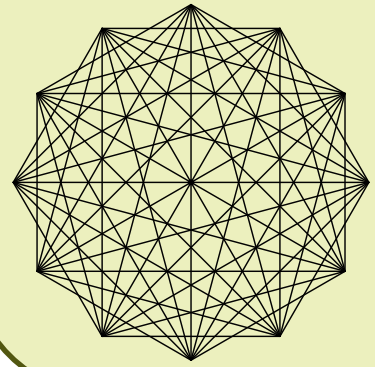
Duration of Node Idleness



70% of idle node events last less than 10 minutes.

Convergence of HPC and Cloud

HPC

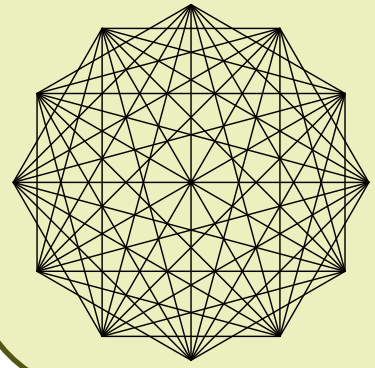


slurm workload manager

Cloud

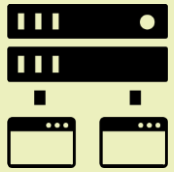
Convergence of HPC and Cloud

HPC



slurm workload manager

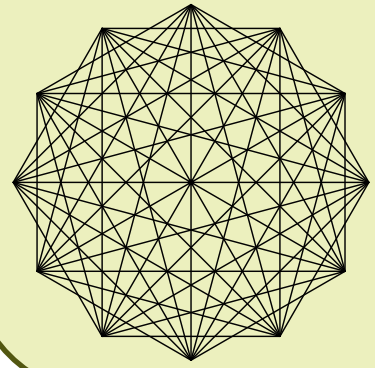
Cloud



Virtualization

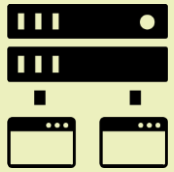
Convergence of HPC and Cloud

HPC



slurm
workload manager

Cloud

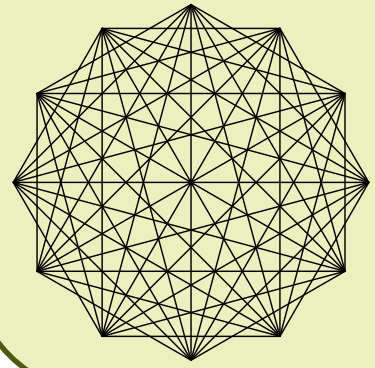


Virtualization

Containers

Convergence of HPC and Cloud

HPC



slurm workload manager

Cloud

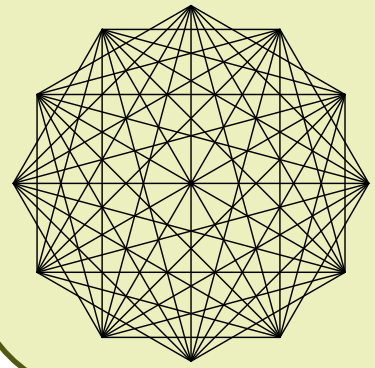
Virtualization

Containers

Pay-as-you-go

Convergence of HPC and Cloud

HPC



slurm workload manager

Cloud

Virtualization

Containers

Pay-as-you-go

Multi-tenancy

Convergence of HPC and Cloud

HPC

slurm workload manager

Cloud

Virtualization

Containers

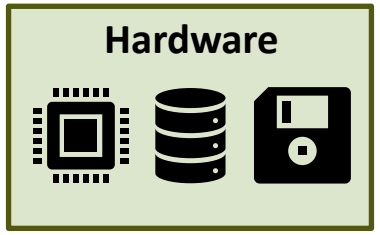
Elastic Runtimes

Pay-as-you-go

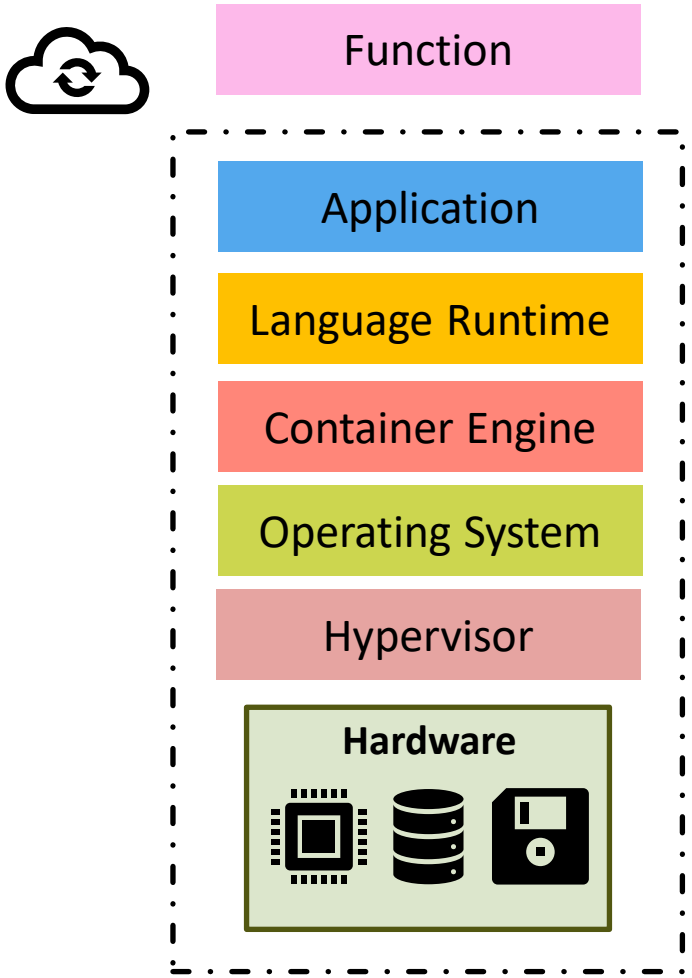
Multi-tenancy

Serverless as a Way Forward

Serverless as a Way Forward

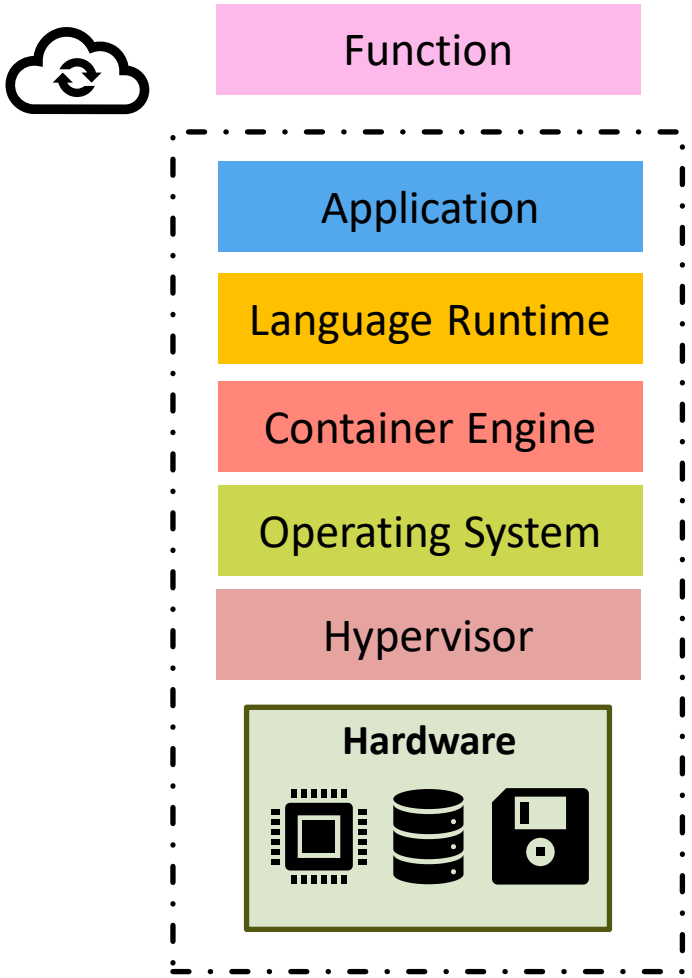


Serverless as a Way Forward



Function-as-a-Service

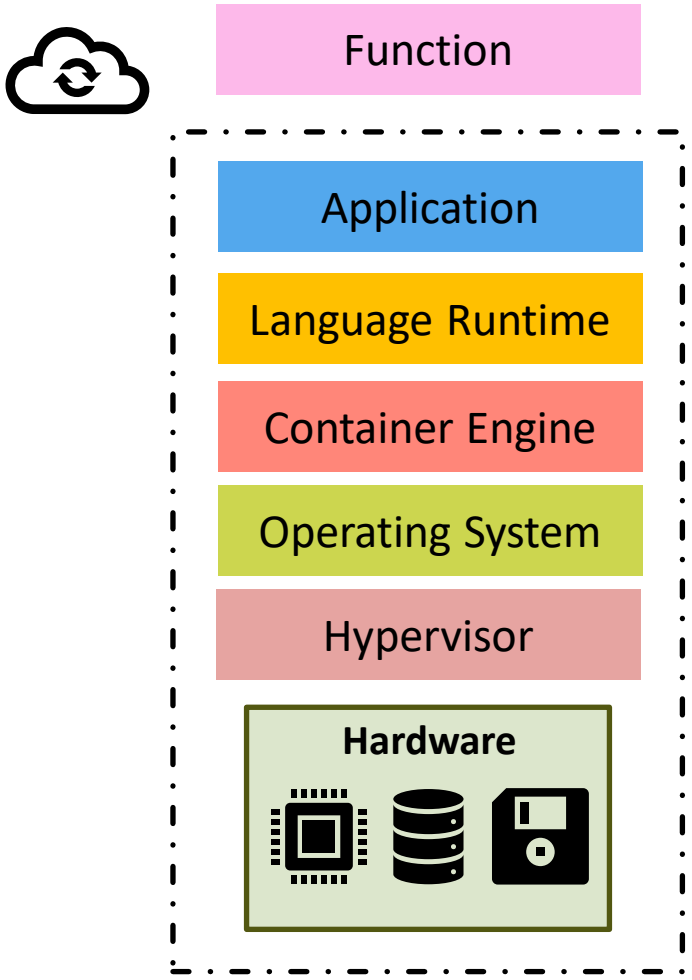
Serverless as a Way Forward



Function-as-a-Service

Fine-grained computing

Serverless as a Way Forward



Function-as-a-Service

Fine-grained computing

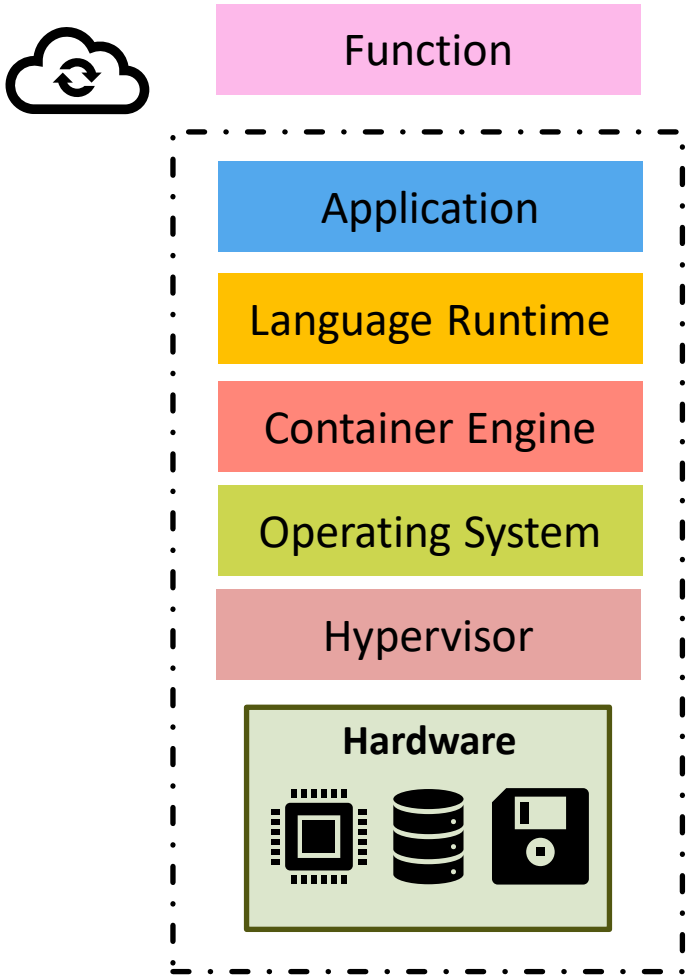
Cloud

Serverless in the Wild: Characterizing and Optimizing the Serverless Workload at a Large Cloud Provider

Mohammad Shahrad, Rodrigo Fonseca, Íñigo Goiri, Gohar Chaudhry, Paul Batum, Jason Cooke, Eduardo Laureano, Colby Tresness, Mark Russinovich, and Ricardo Bianchini, *Microsoft Azure and Microsoft Research*

"We observe that 50% of the functions execute for less than 1s on average, and 50% of the functions have maximum execution time shorter than ~3s; 90% of the functions take at most 60s, and ***96% of functions take less than 60s on average.****"*

Serverless as a Way Forward



Function-as-a-Service

Fine-grained computing

Cloud

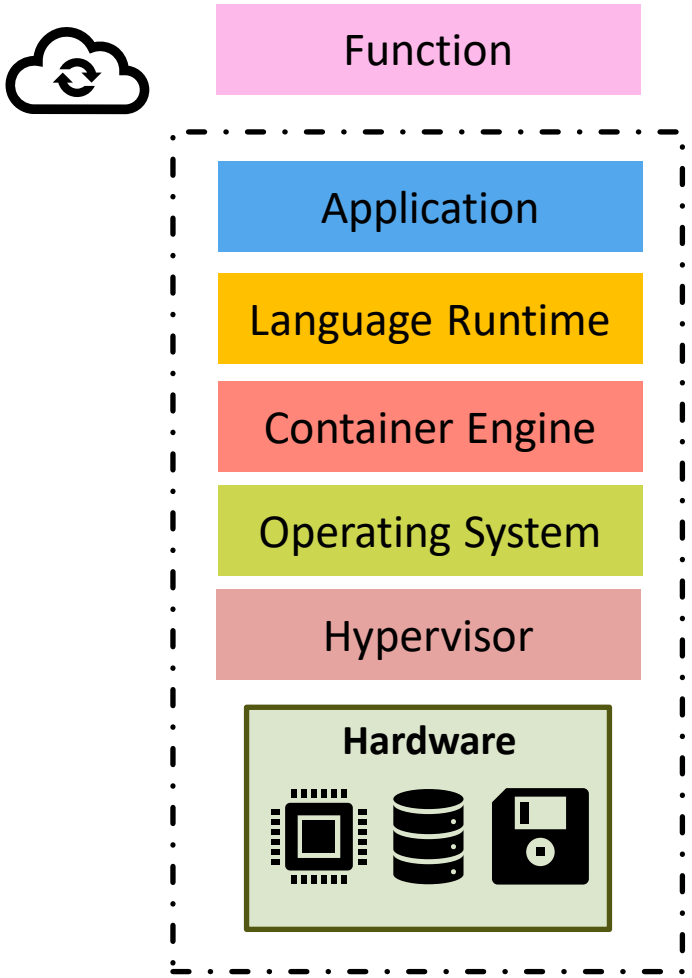
HPC

The globus compute dataset: An open function-as-a-service dataset from the edge to the cloud

André Bauer [a,b,*], Haochen Pan [a], Ryan Chard [b], Yadu Babuji [a], Josh Bryan [a], Devesh Tiwari [c], Ian Foster [b,a], Kyle Chard [a,b]

"The average execution time for a task was 49.04 s, with a median value of 0.03 s. Additionally, ***74% of submitted tasks had an execution time of less than 1 s****."*

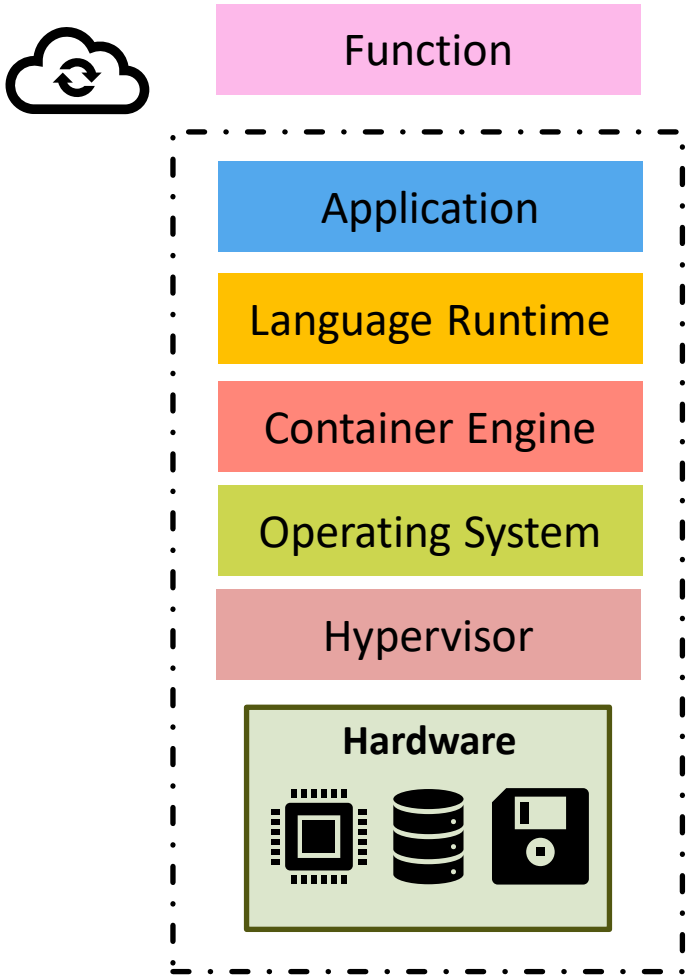
Serverless as a Way Forward



Function-as-a-Service

- **Fine-grained computing**
- **Abstracted resource management**

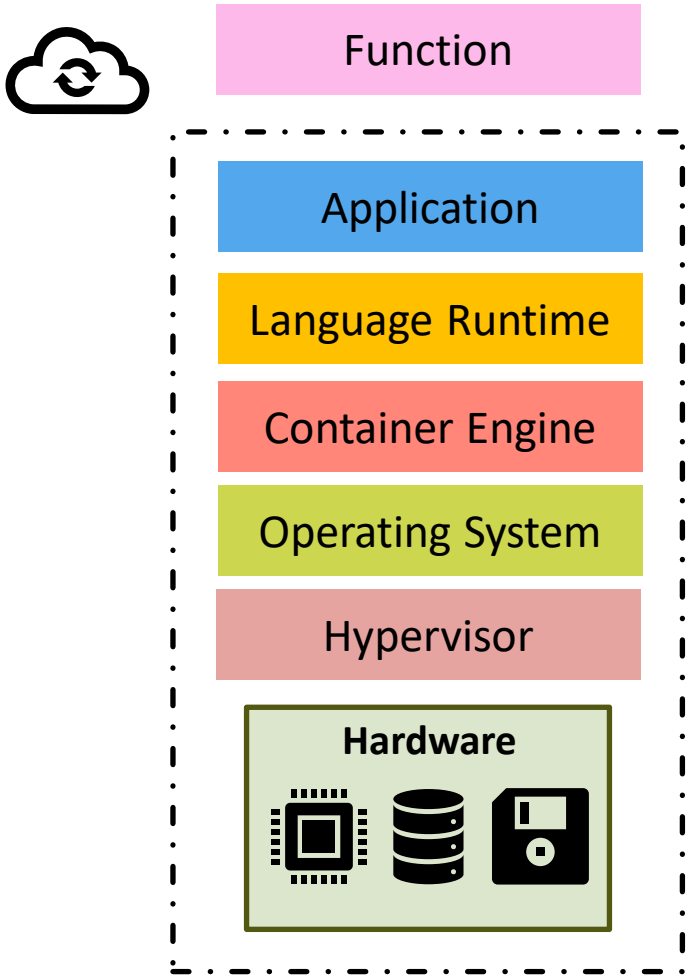
Serverless as a Way Forward



Function-as-a-Service

- **Fine-grained computing**
- **Abstracted resource management**
- **Elastic scheduling**

Serverless as a Way Forward



Function-as-a-Service

- **Fine-grained computing**
- **Abstracted resource management**
- **Elastic scheduling**

- **Performance?**

Convergence of HPC and Cloud

HPC

slurm workload manager

Cloud

Virtualization

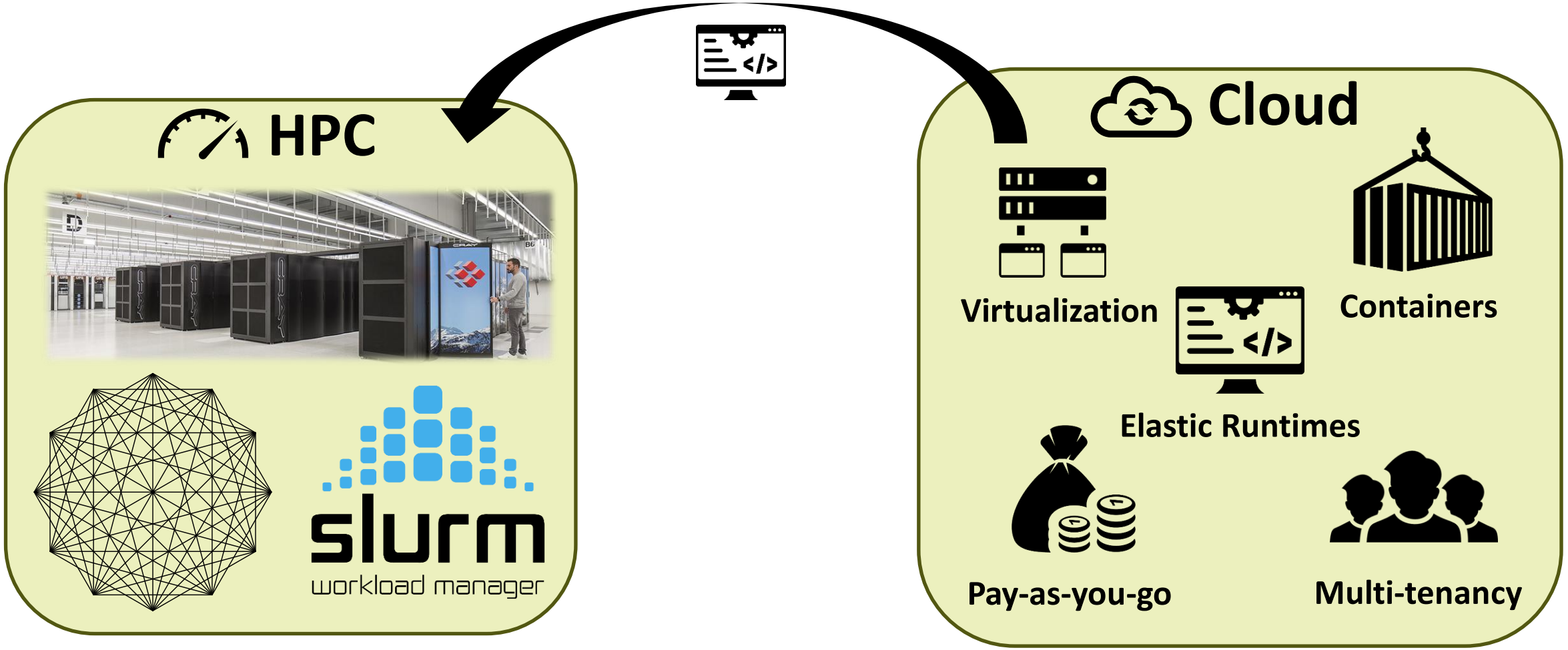
Containers

Elastic Runtimes

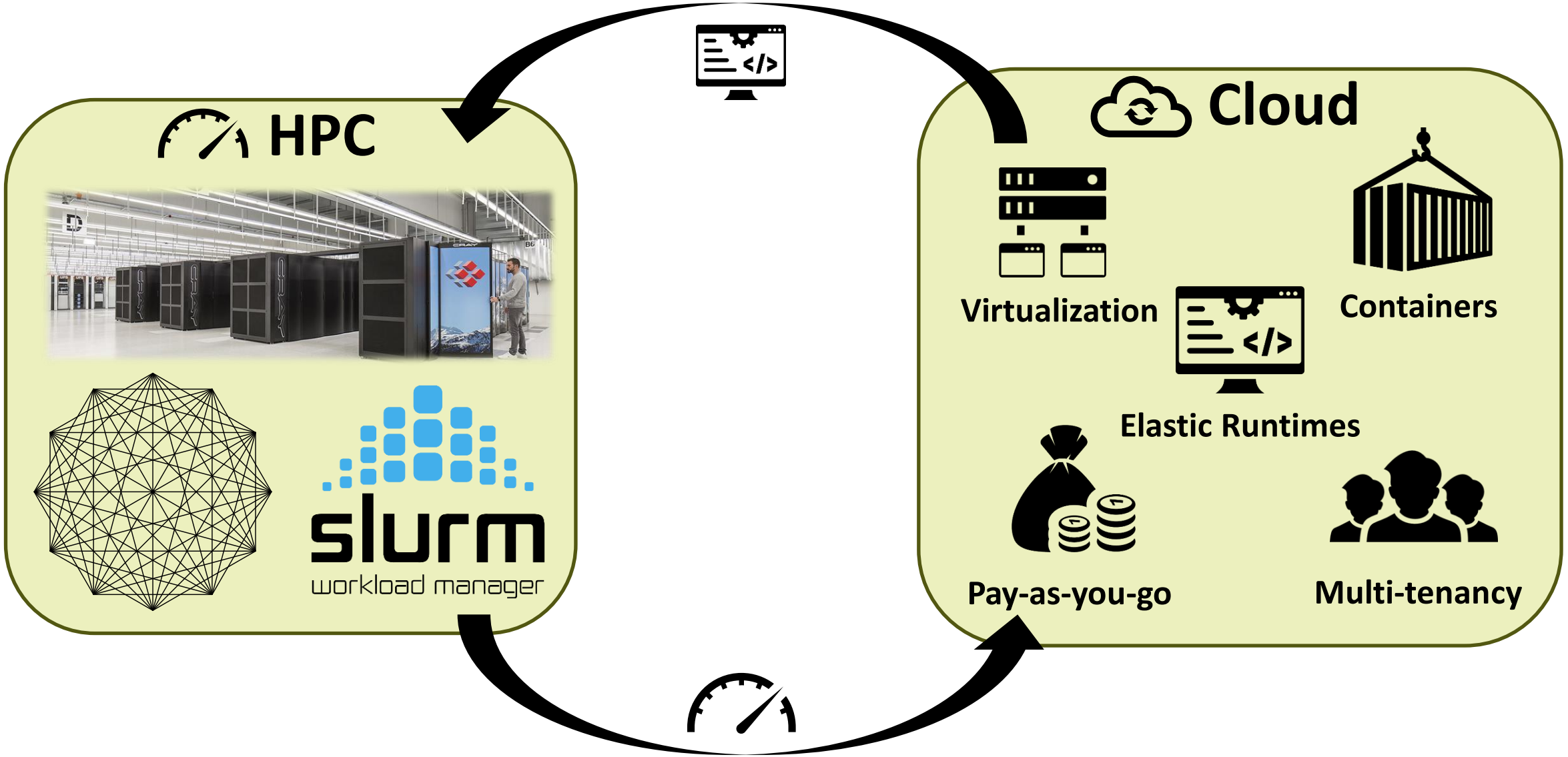
Pay-as-you-go

Multi-tenancy

Convergence of HPC and Cloud

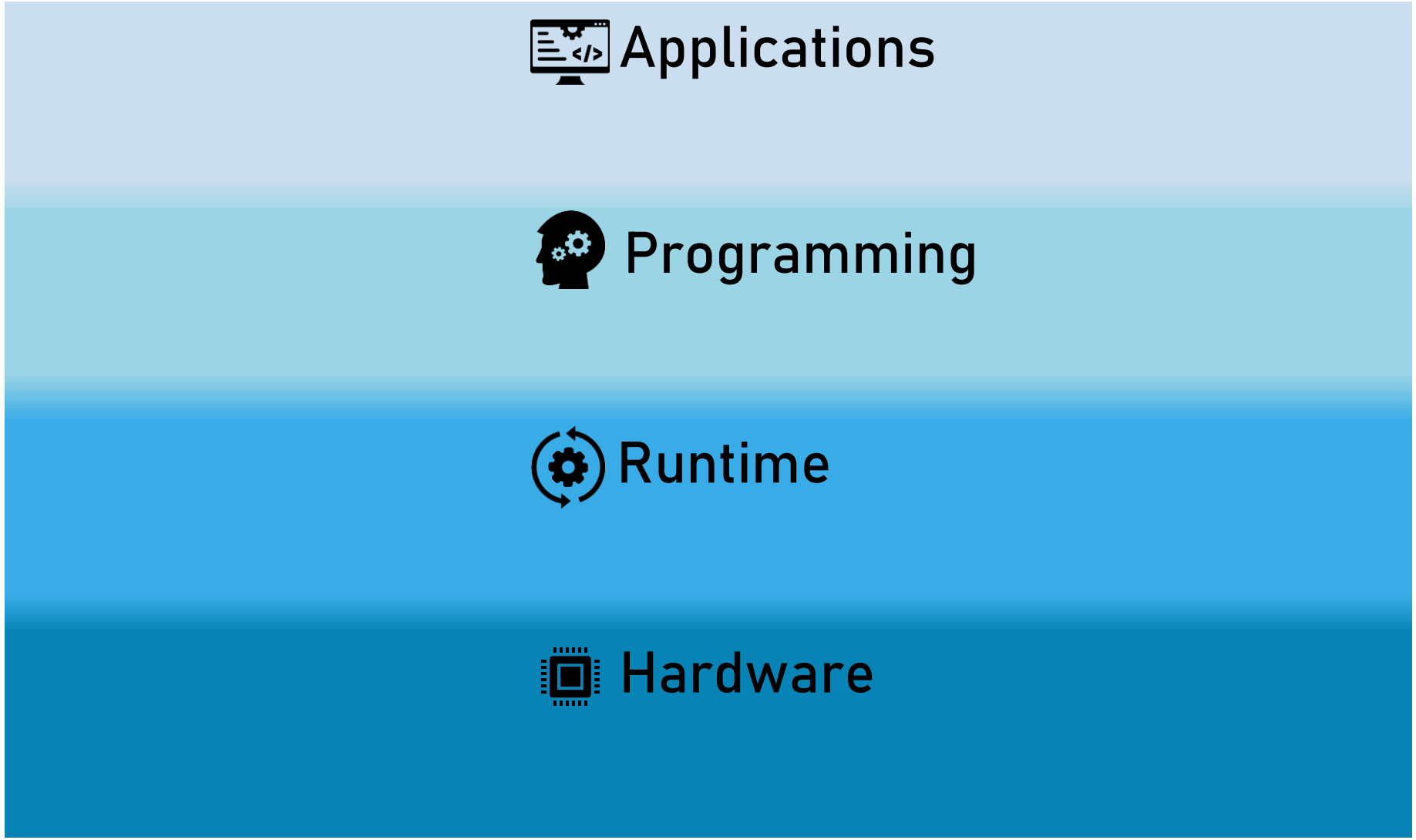


Convergence of HPC and Cloud

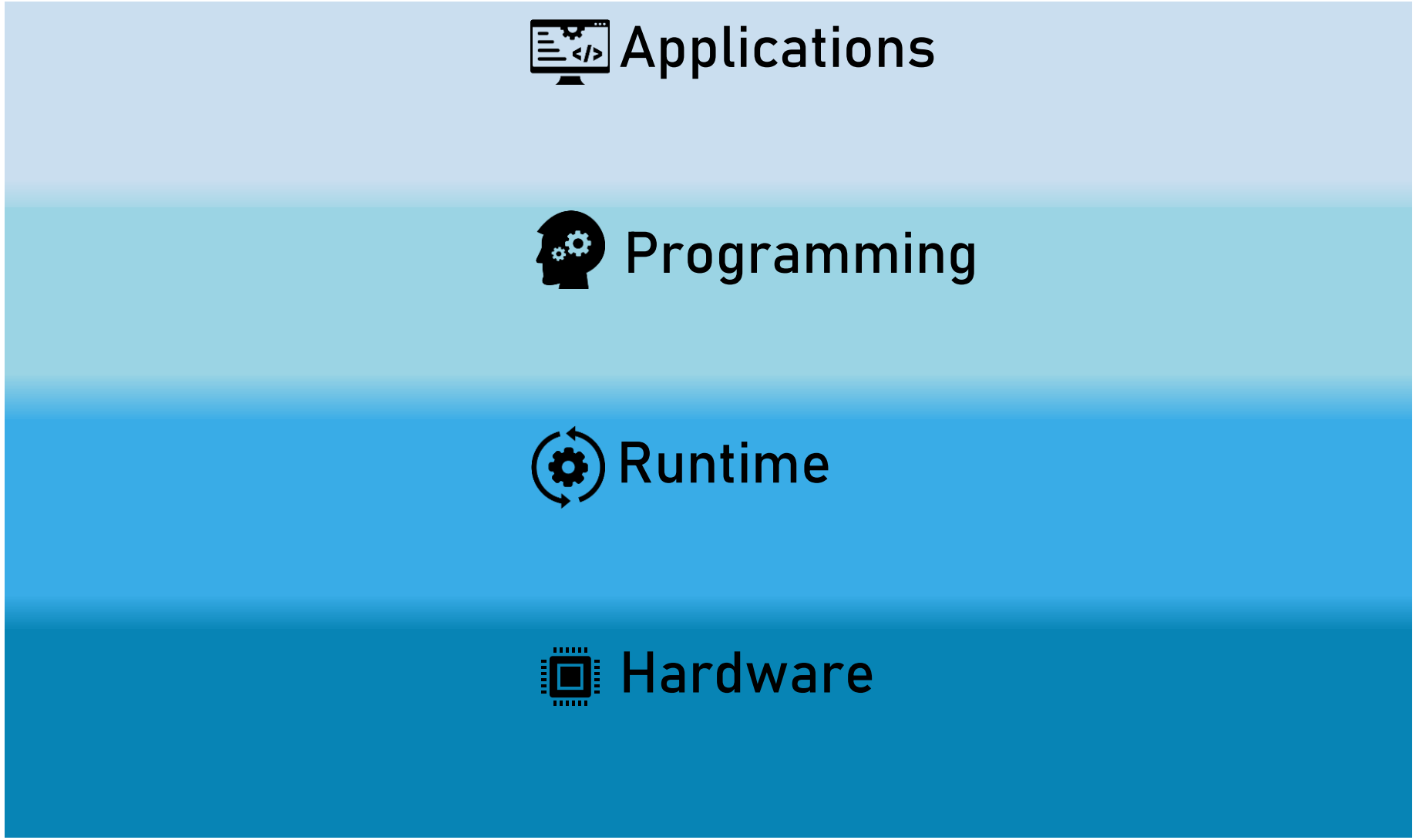


High-Performance Serverless Stack

High-Performance Serverless Stack

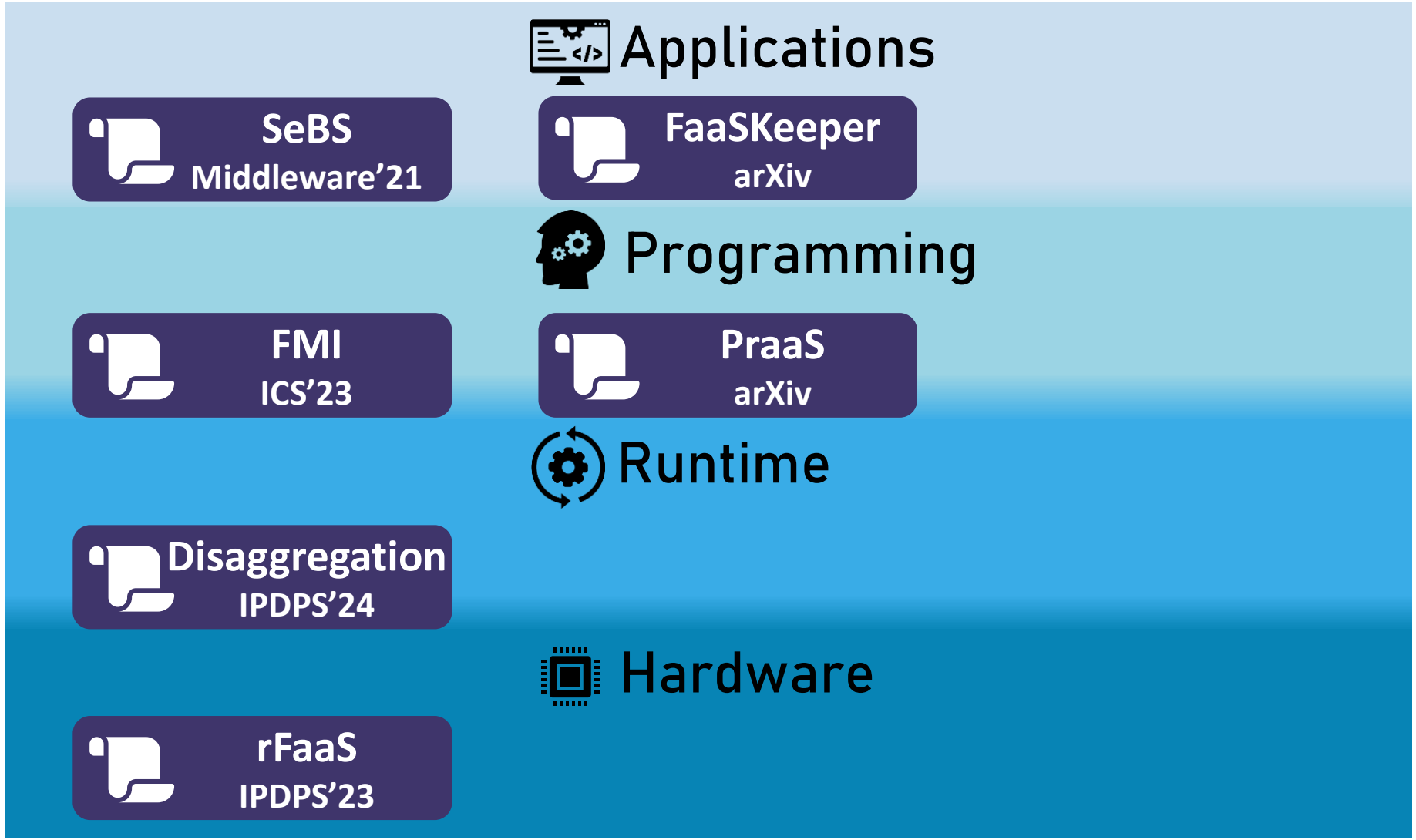


High-Performance Serverless Stack



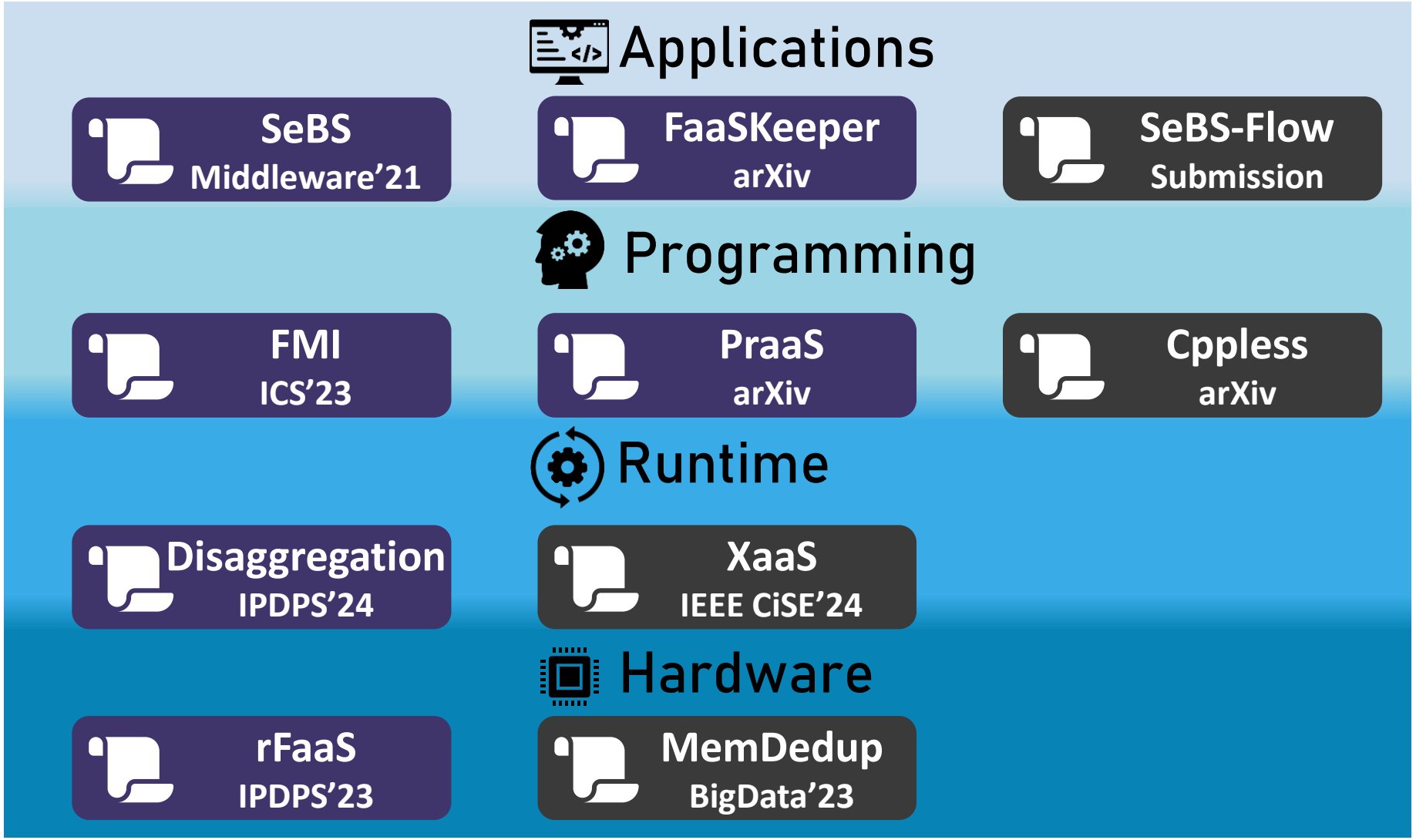
First-author
Collaborations
Submissions

High-Performance Serverless Stack



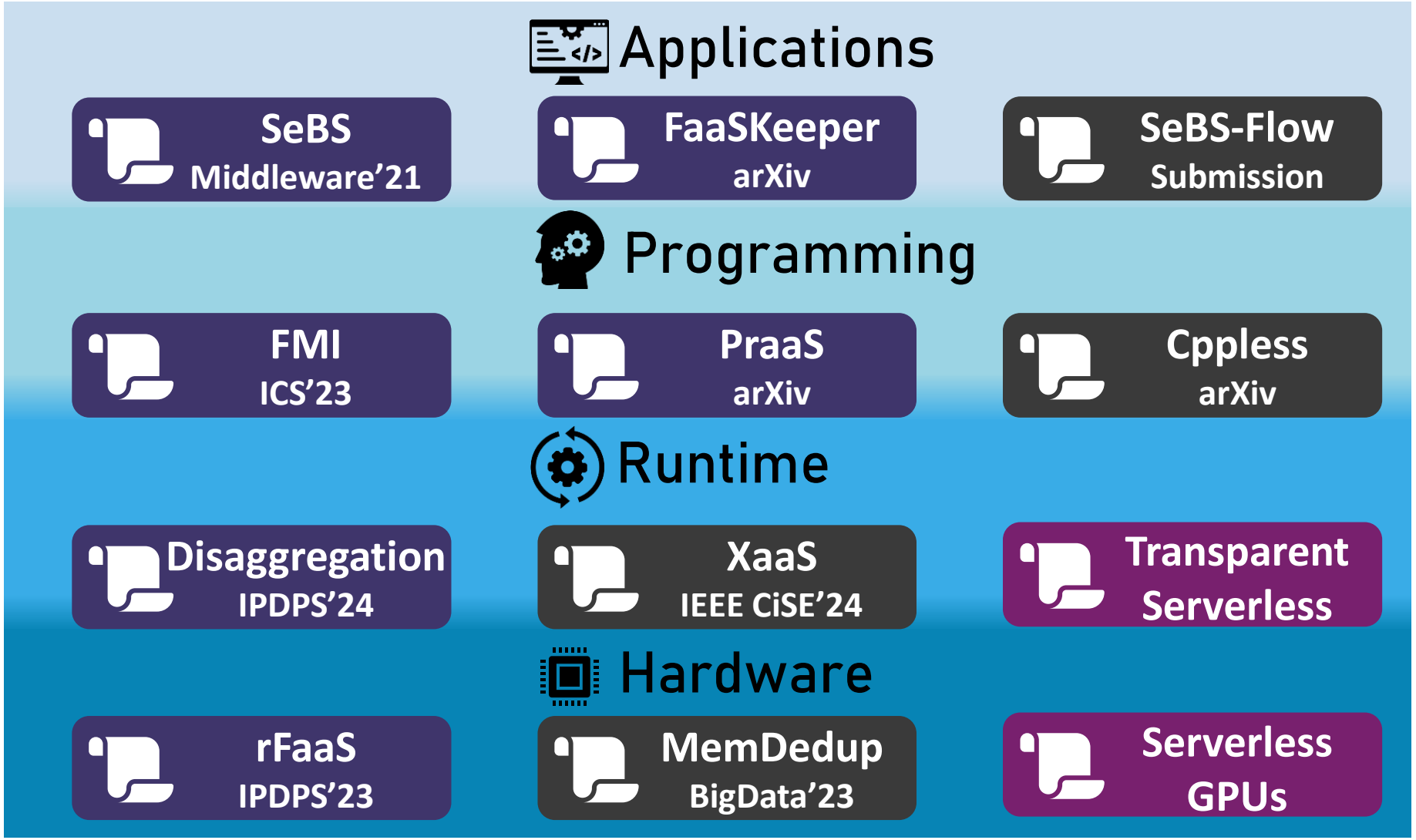
First-author
Collaborations
Submissions

High-Performance Serverless Stack



First-author
Collaborations
Submissions

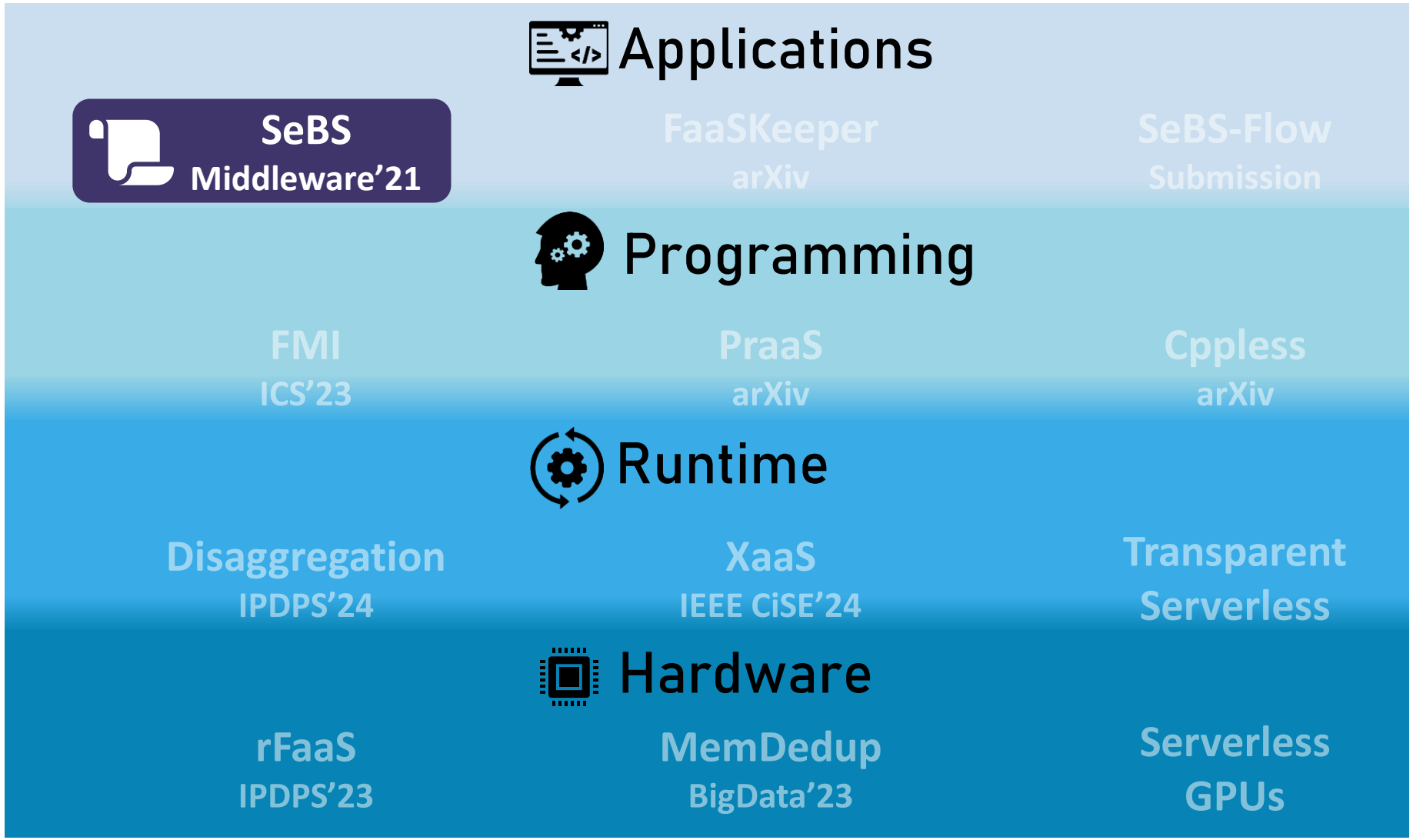
High-Performance Serverless Stack



First-author
Collaborations
Submissions

High-Performance Serverless Stack

How does serverless performance look like?



First-author
Collaborations
Submissions

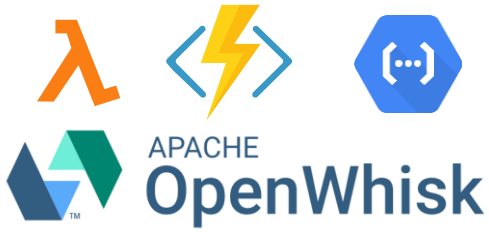
SeBS: The Serverless Benchmark Suite

ACM/IFIP Middleware' 21

SeBS: The Serverless Benchmark Suite

ACM/IFIP Middleware' 21

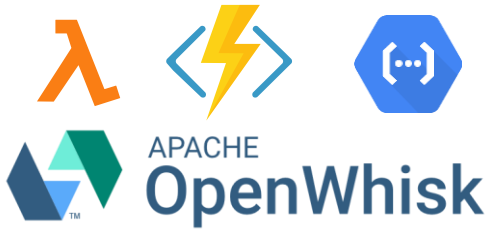
Serverless Platforms



SeBS: The Serverless Benchmark Suite

ACM/IFIP Middleware' 21

Serverless Platforms

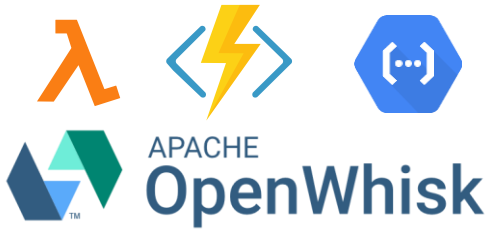


Benchmarks

SeBS: The Serverless Benchmark Suite

ACM/IFIP Middleware' 21

Serverless Platforms



Benchmarks

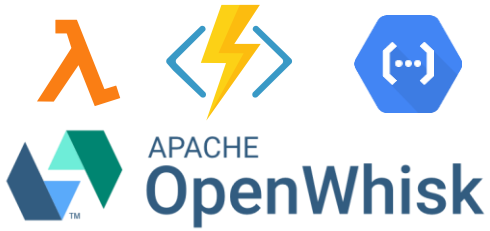
Experiments

Performance & Cost
Invocation Overhead
Container Eviction
Serverless Communication
Serverless Workflows

SeBS: The Serverless Benchmark Suite

ACM/IFIP Middleware' 21

Serverless Platforms

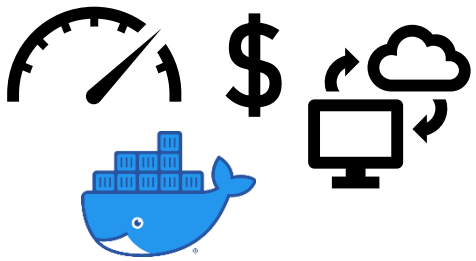


Benchmarks

Experiments

Performance & Cost
Invocation Overhead
Container Eviction
Serverless Communication
Serverless Workflows

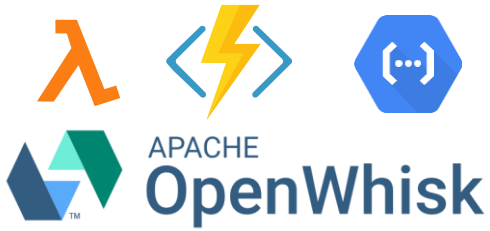
Insights



SeBS: The Serverless Benchmark Suite

ACM/IFIP Middleware' 21

Serverless Platforms

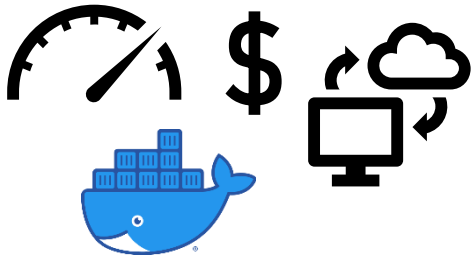


Benchmarks

Experiments

Performance & Cost
Invocation Overhead
Container Eviction
Serverless Communication
Serverless Workflows

Insights

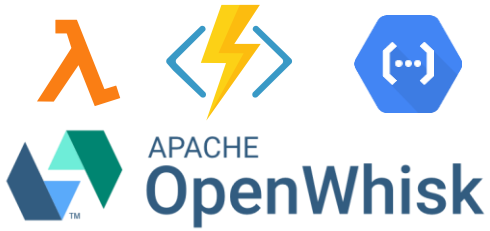


Adoption

SeBS: The Serverless Benchmark Suite

ACM/IFIP Middleware' 21

Serverless Platforms

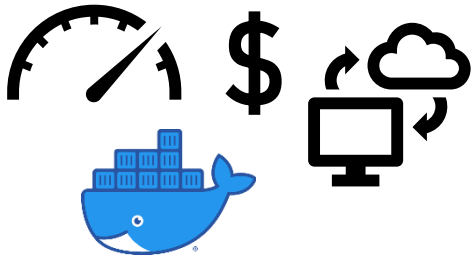


Benchmarks

Experiments

Performance & Cost
Invocation Overhead
Container Eviction
Serverless Communication
Serverless Workflows

Insights



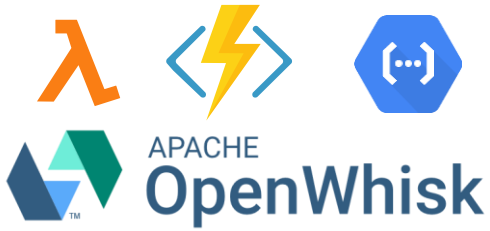
Adoption

110 stars
53 forks
15 contributors

SeBS: The Serverless Benchmark Suite

ACM/IFIP Middleware' 21

Serverless Platforms

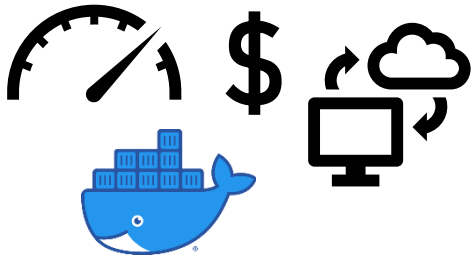


Benchmarks

Experiments

Performance & Cost
Invocation Overhead
Container Eviction
Serverless Communication
Serverless Workflows

Insights



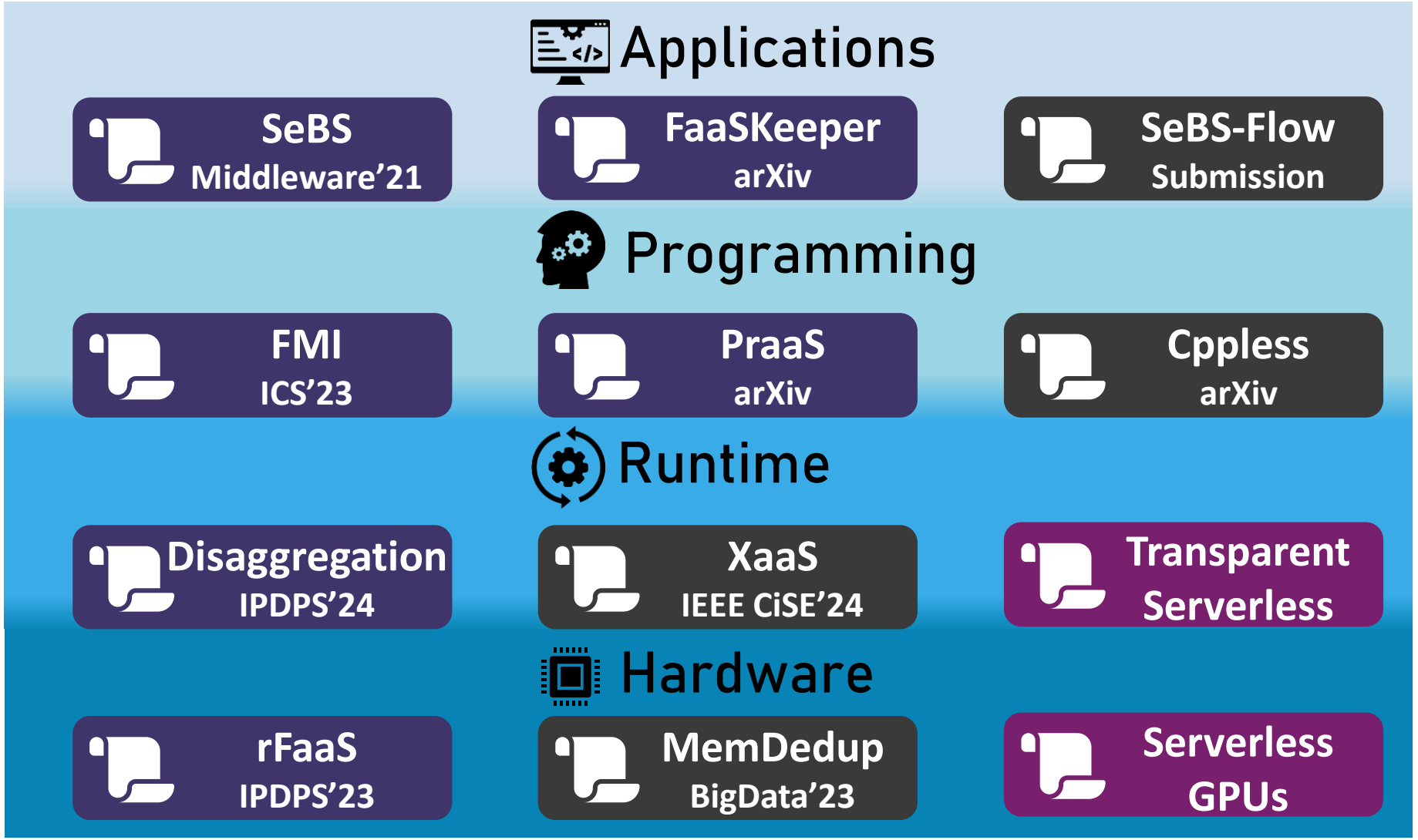
Adoption

110 stars
53 forks
15 contributors

107 citations

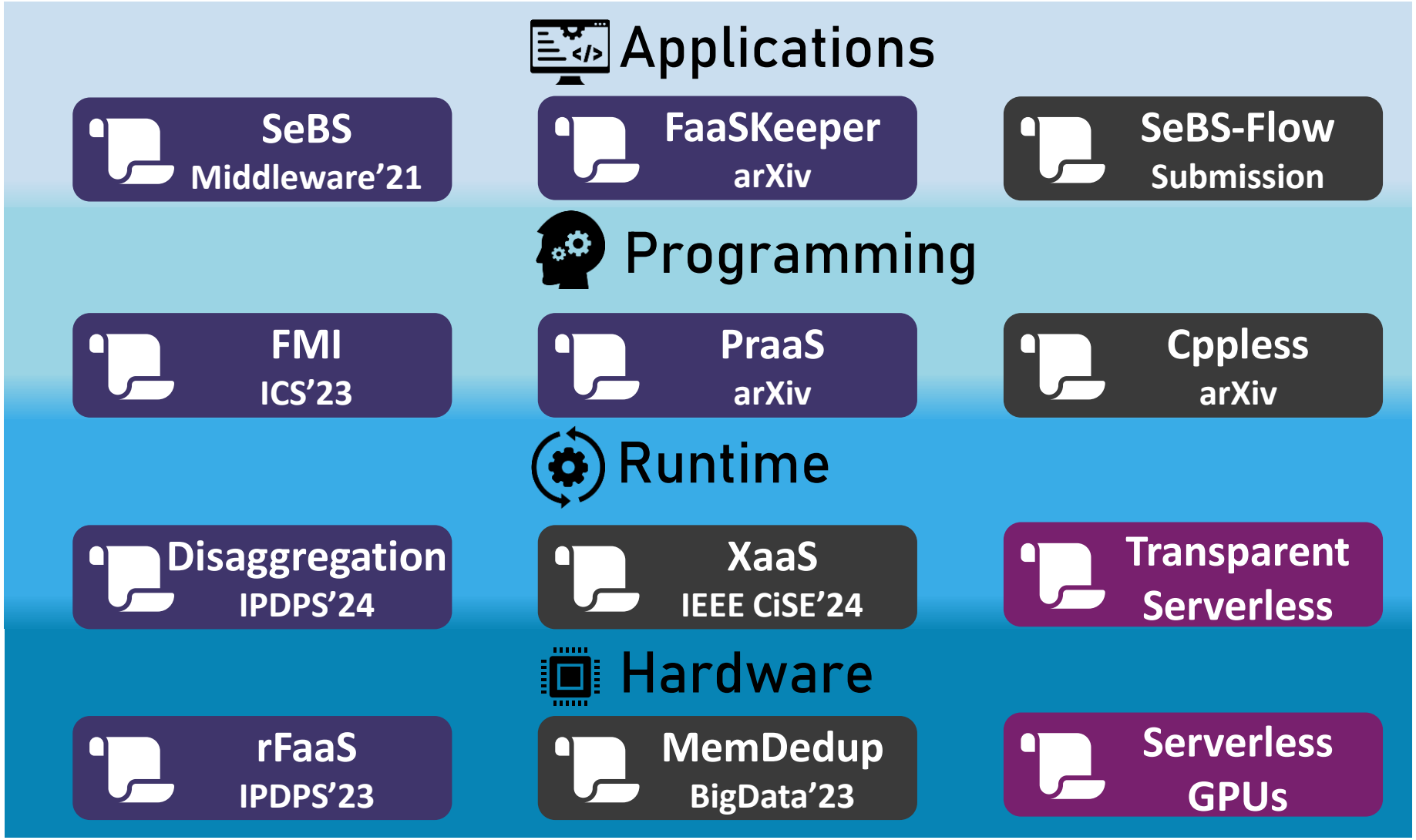
High-Performance Serverless Stack

How does serverless performance look like?



High-Performance Serverless Stack

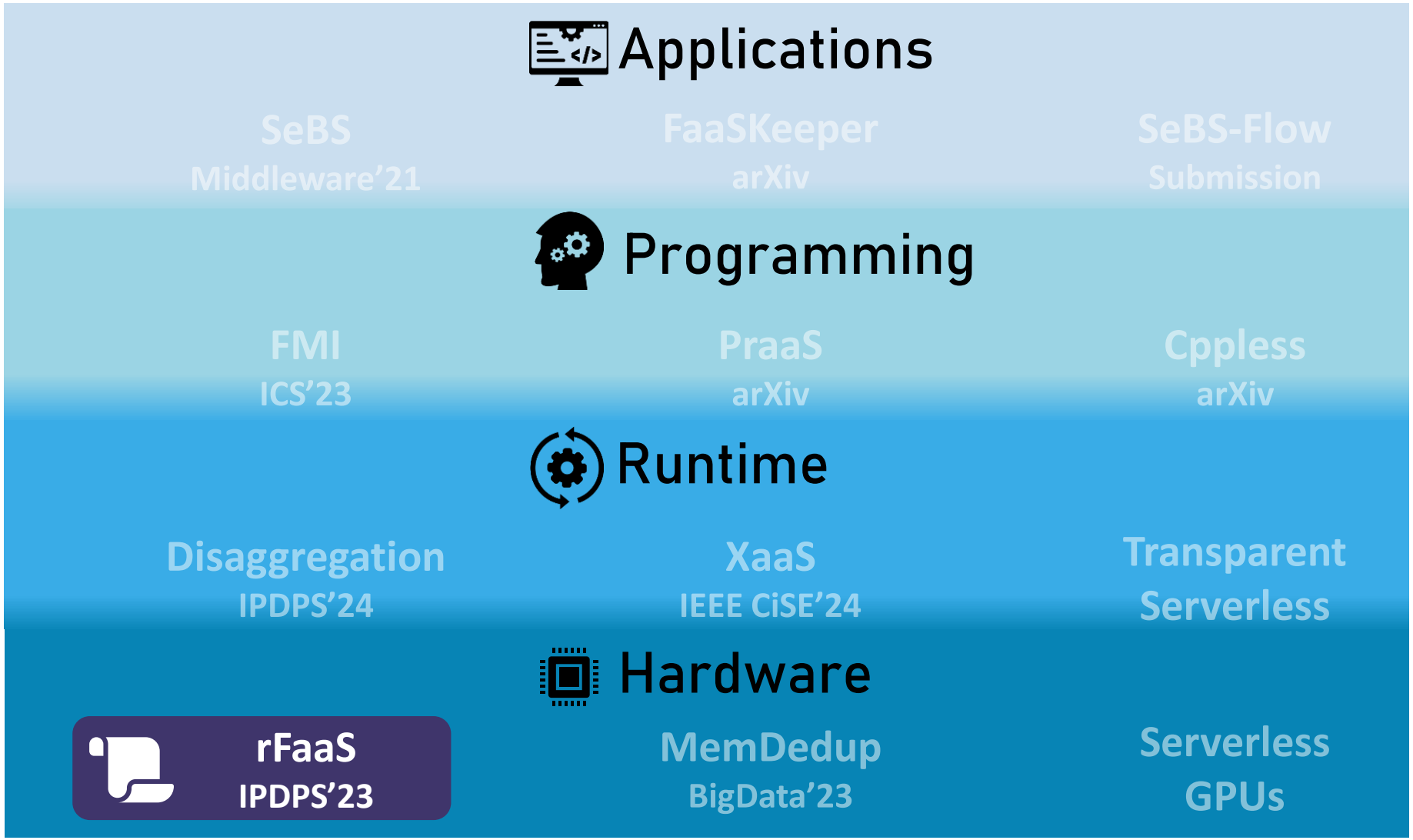
Multi-platform benchmarking suite.



High-Performance Serverless Stack

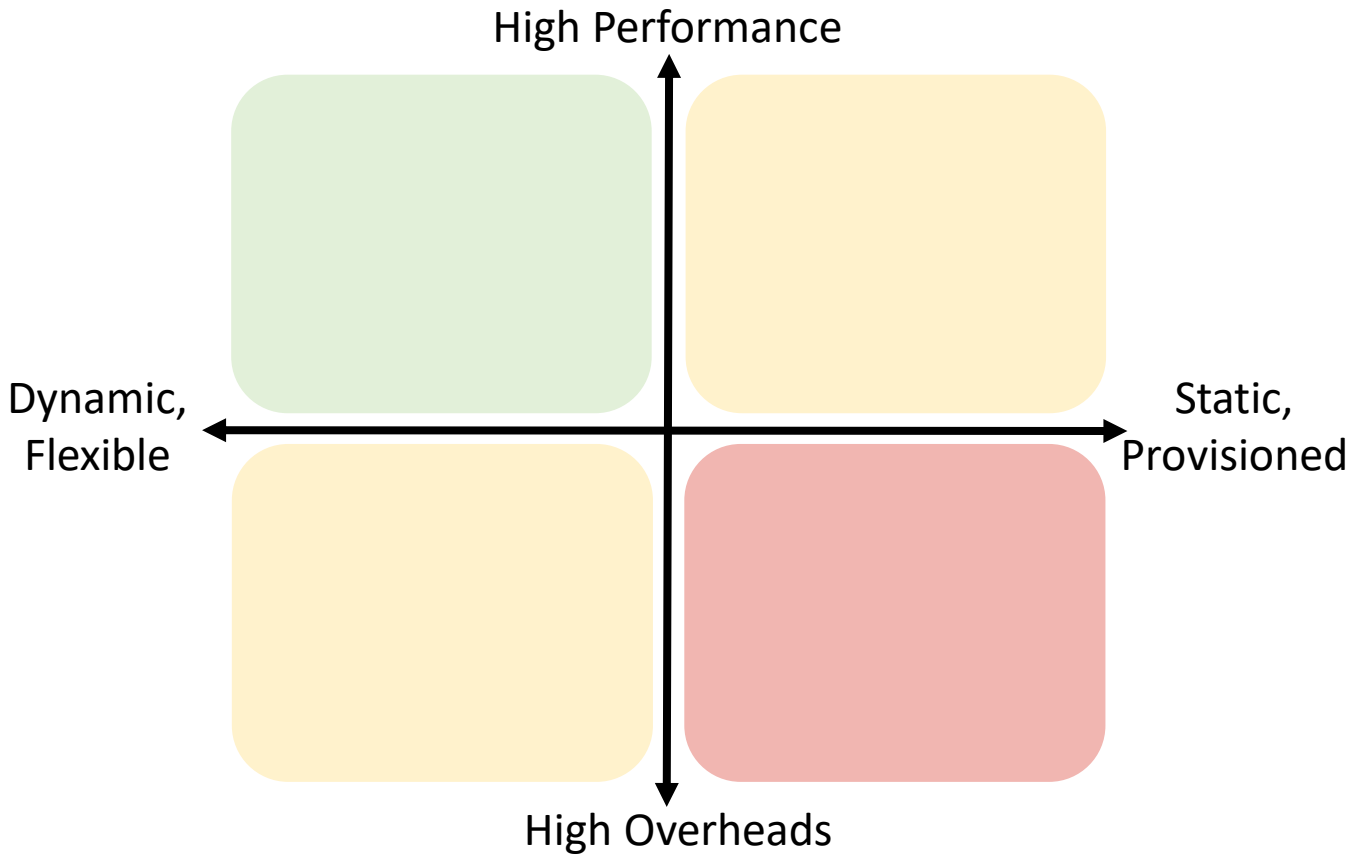
Multi-platform benchmarking suite.

Functions are expensive to invoke.



Function-as-a-Service for HPC

IEEE IPDPS 2023

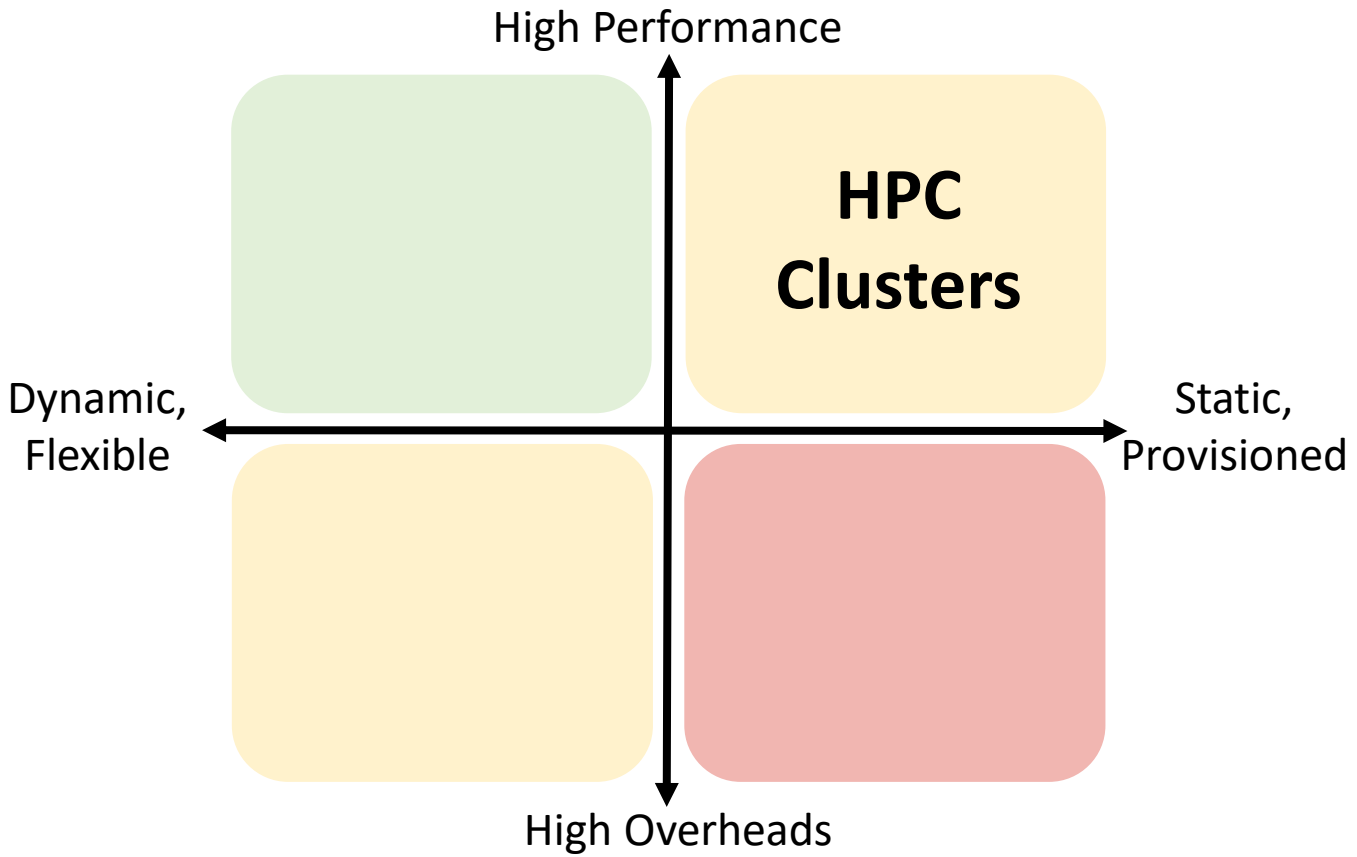


Function-as-a-Service for HPC

IEEE IPDPS 2023

Long-running jobs

Static parallelism

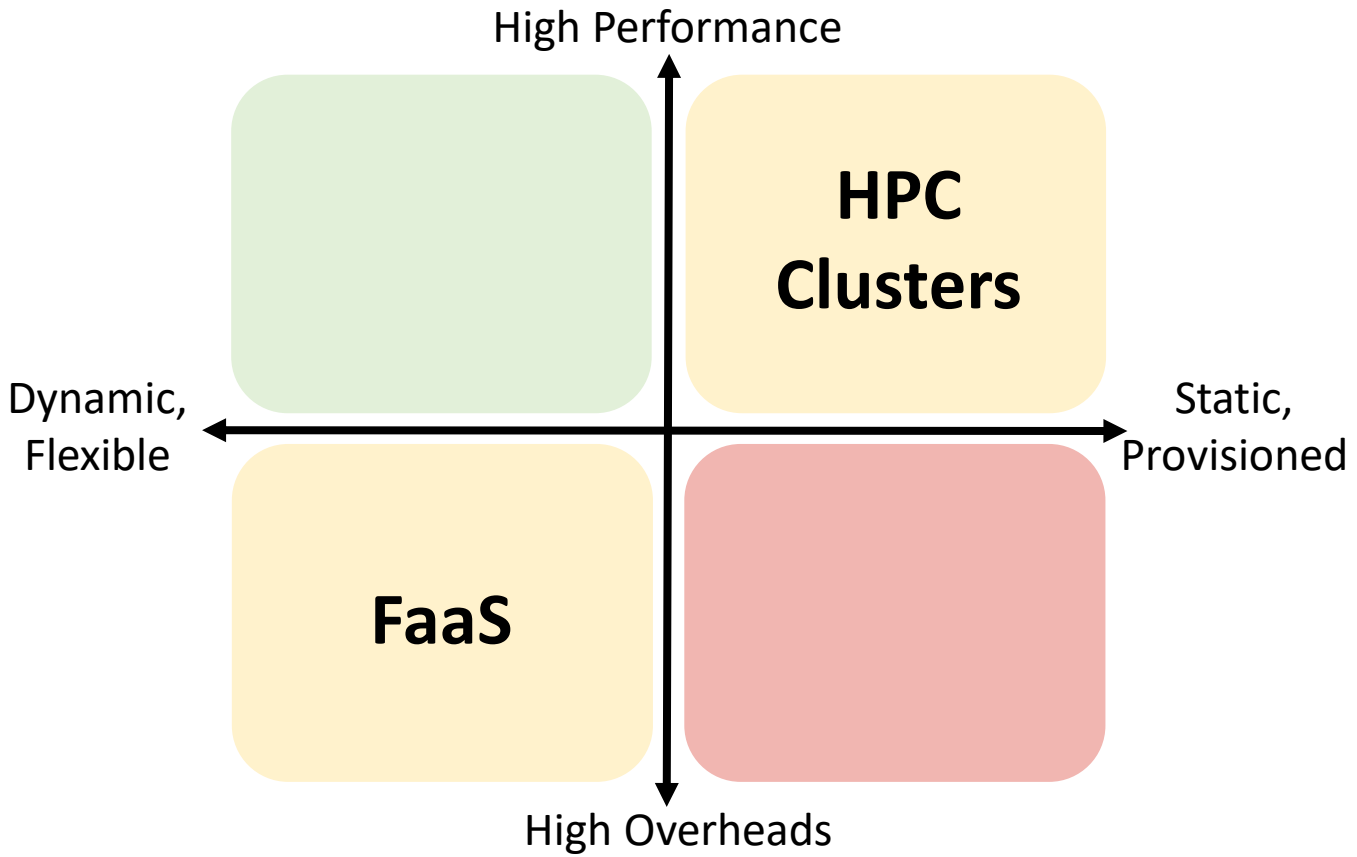


Function-as-a-Service for HPC

IEEE IPDPS 2023

Long-running jobs

Static parallelism



Malleable jobs

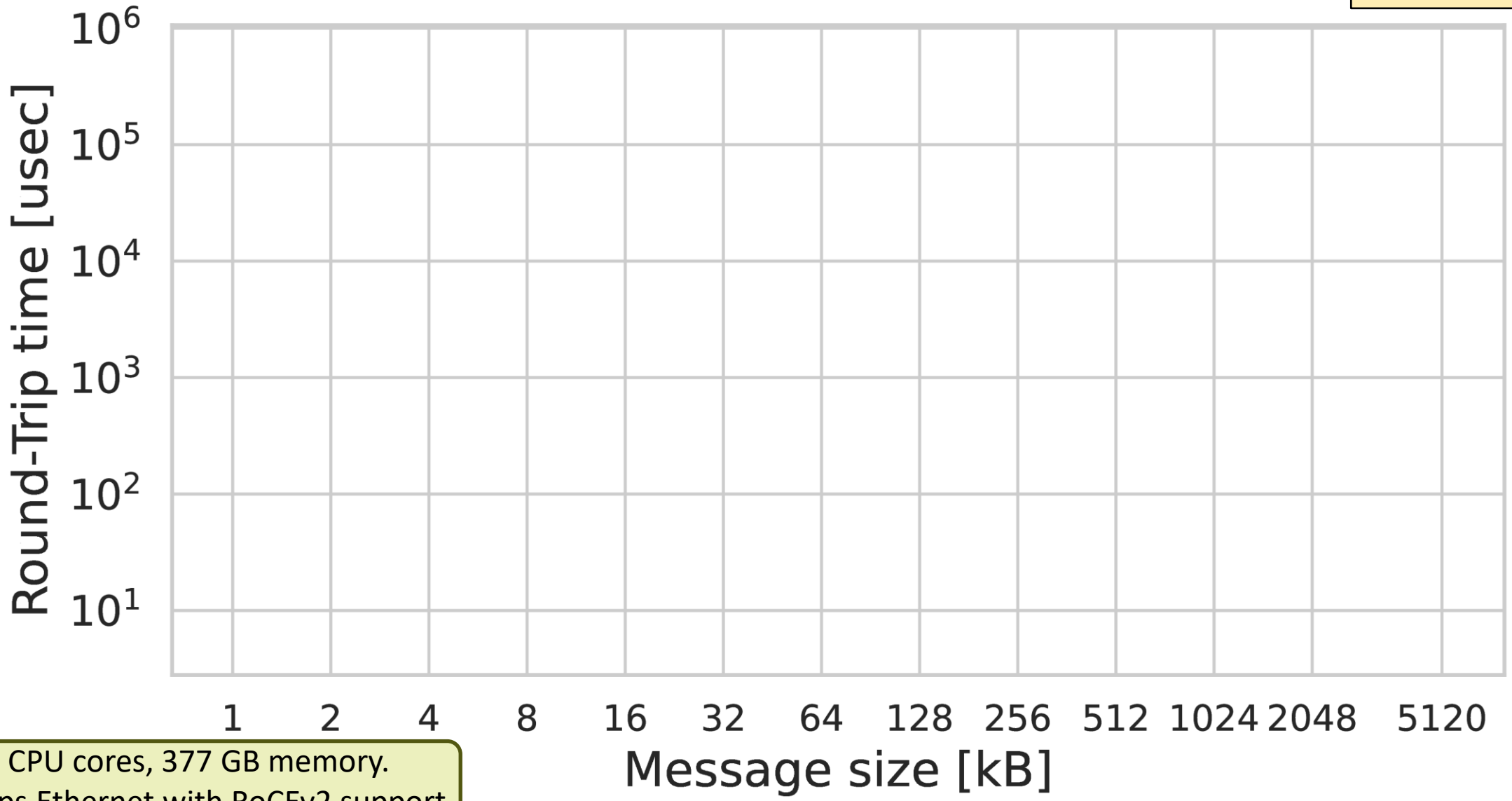
Latency-sensitive jobs

Interactive computations

Dynamic parallelism

How fast are invocations in FaaS?

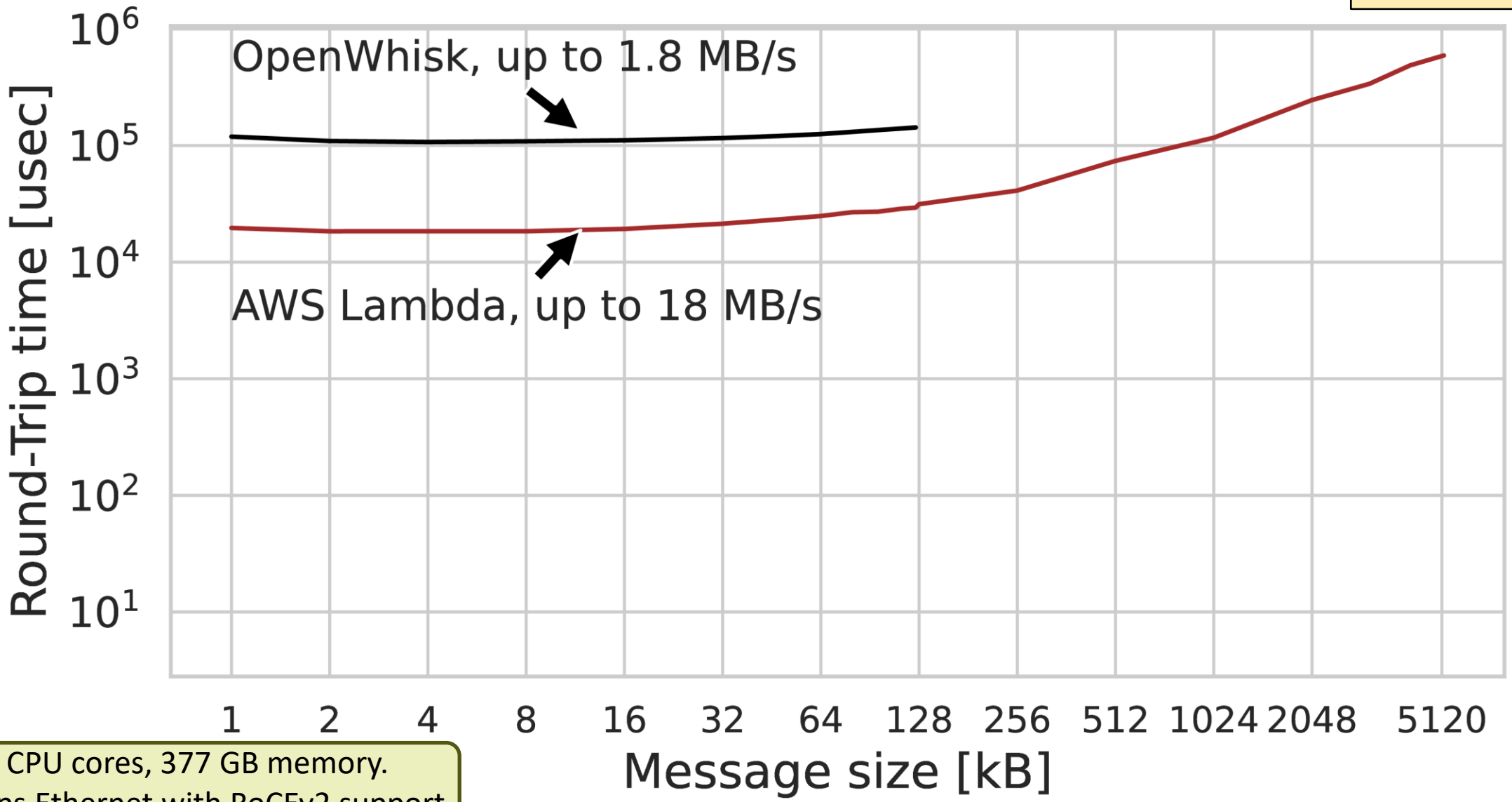
IEEE IPDPS 2023



36 CPU cores, 377 GB memory.
100 Gbps Ethernet with RoCEv2 support.

How fast are invocations in FaaS?

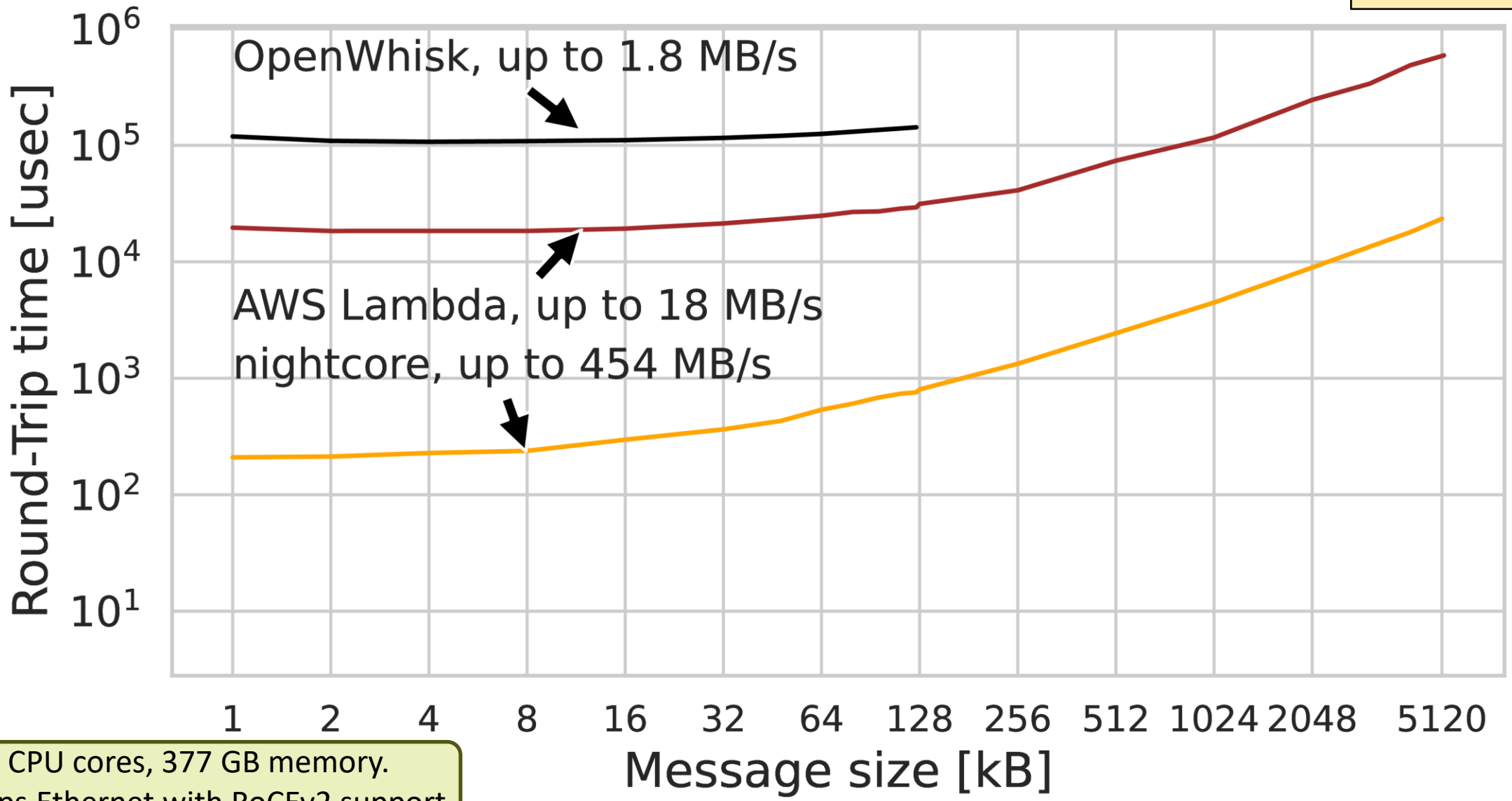
IEEE IPDPS 2023



36 CPU cores, 377 GB memory.
100 Gbps Ethernet with RoCEv2 support.

How fast are invocations in FaaS?

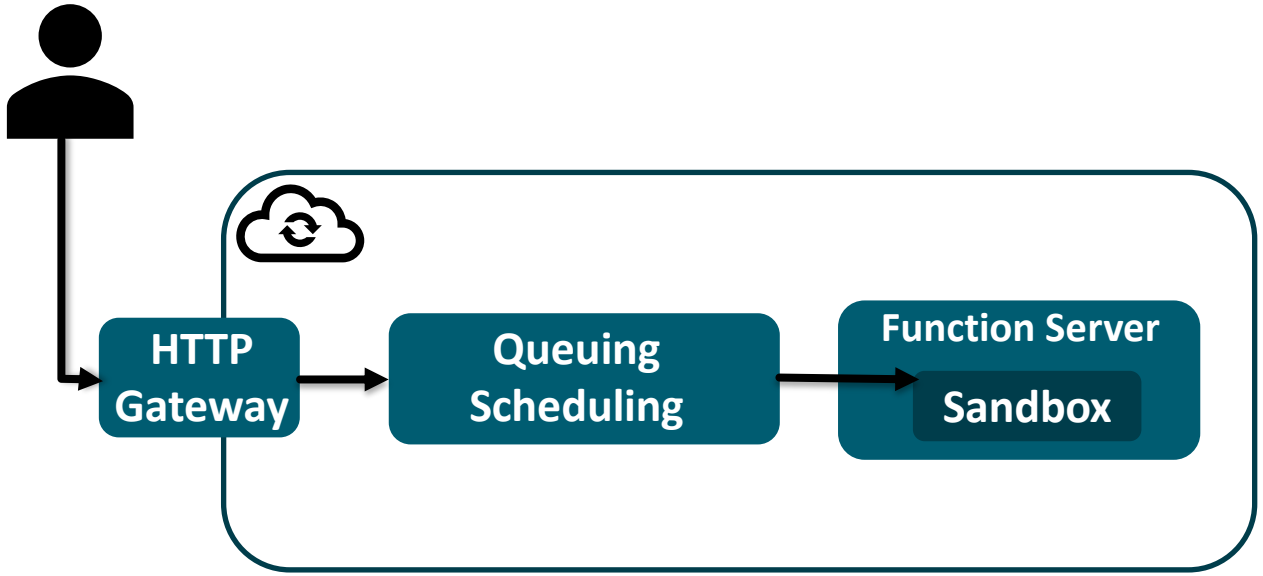
IEEE IPDPS 2023



36 CPU cores, 377 GB memory.
100 Gbps Ethernet with RoCEv2 support.

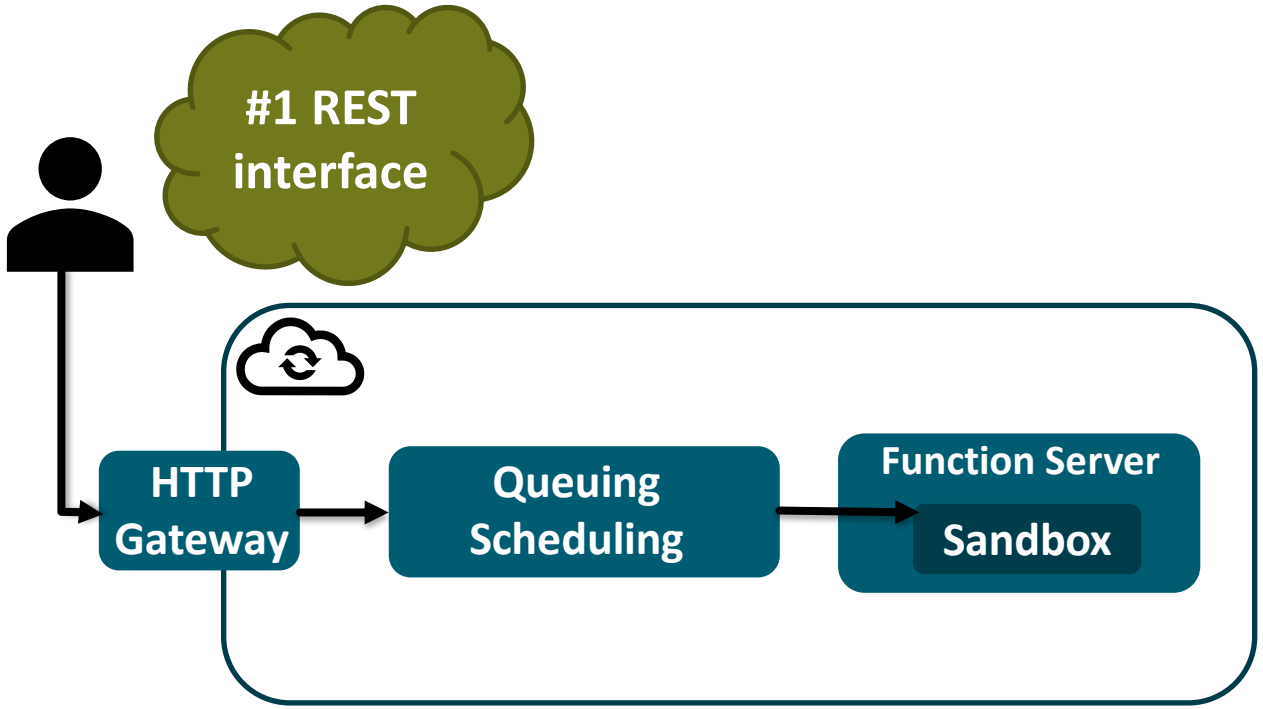
Why is FaaS slow?

IEEE IPDPS 2023



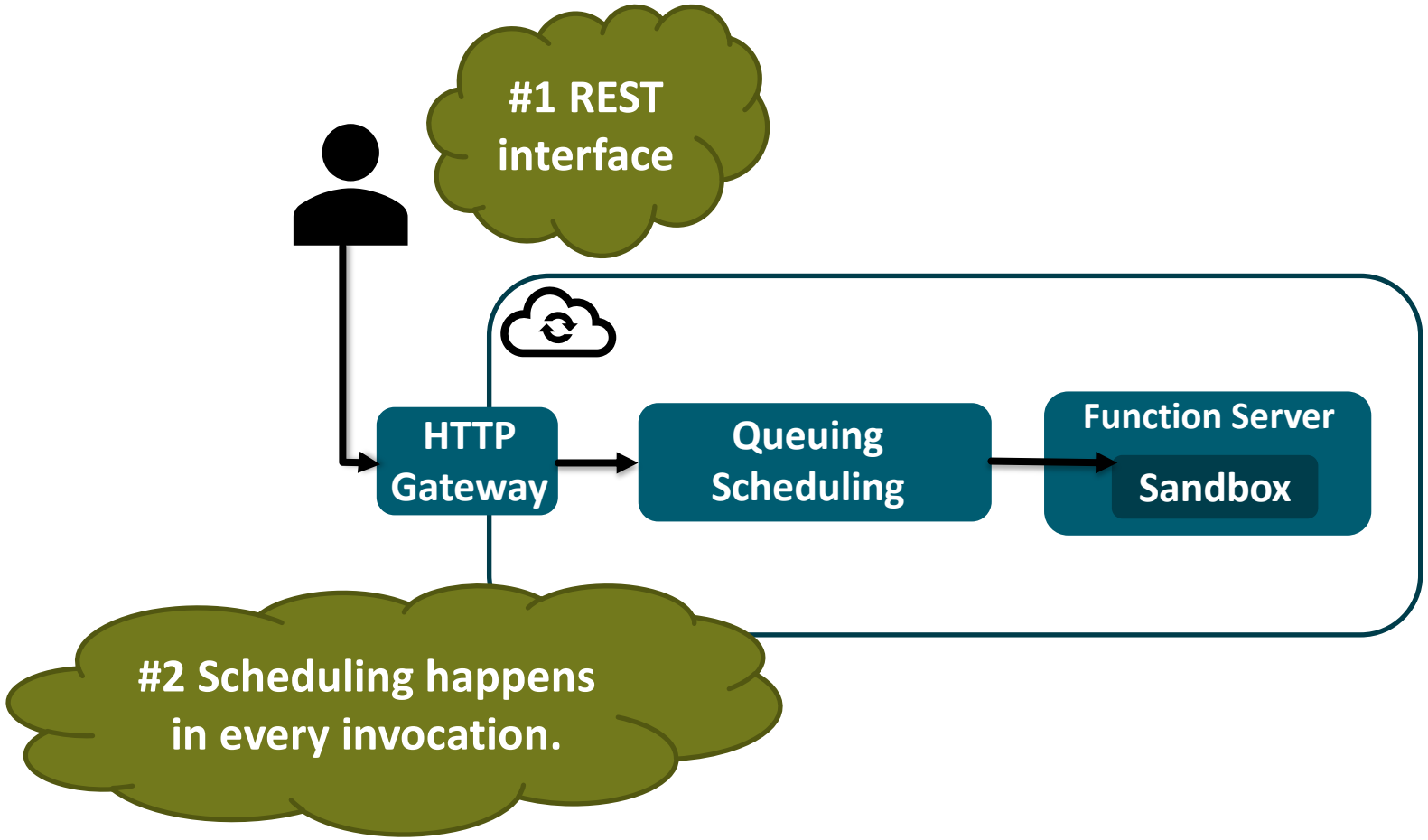
Why is FaaS slow?

IEEE IPDPS 2023



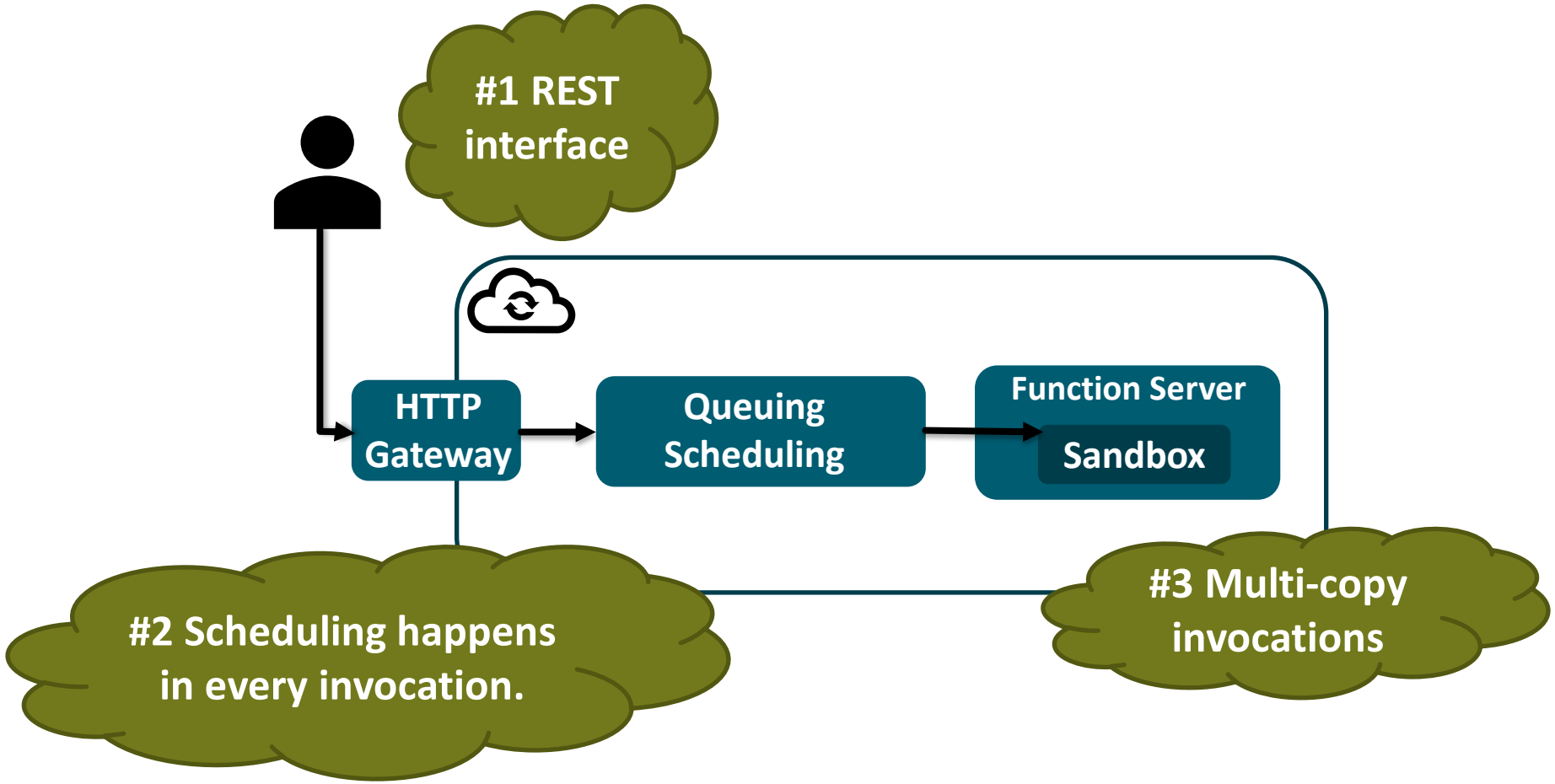
Why is FaaS slow?

IEEE IPDPS 2023



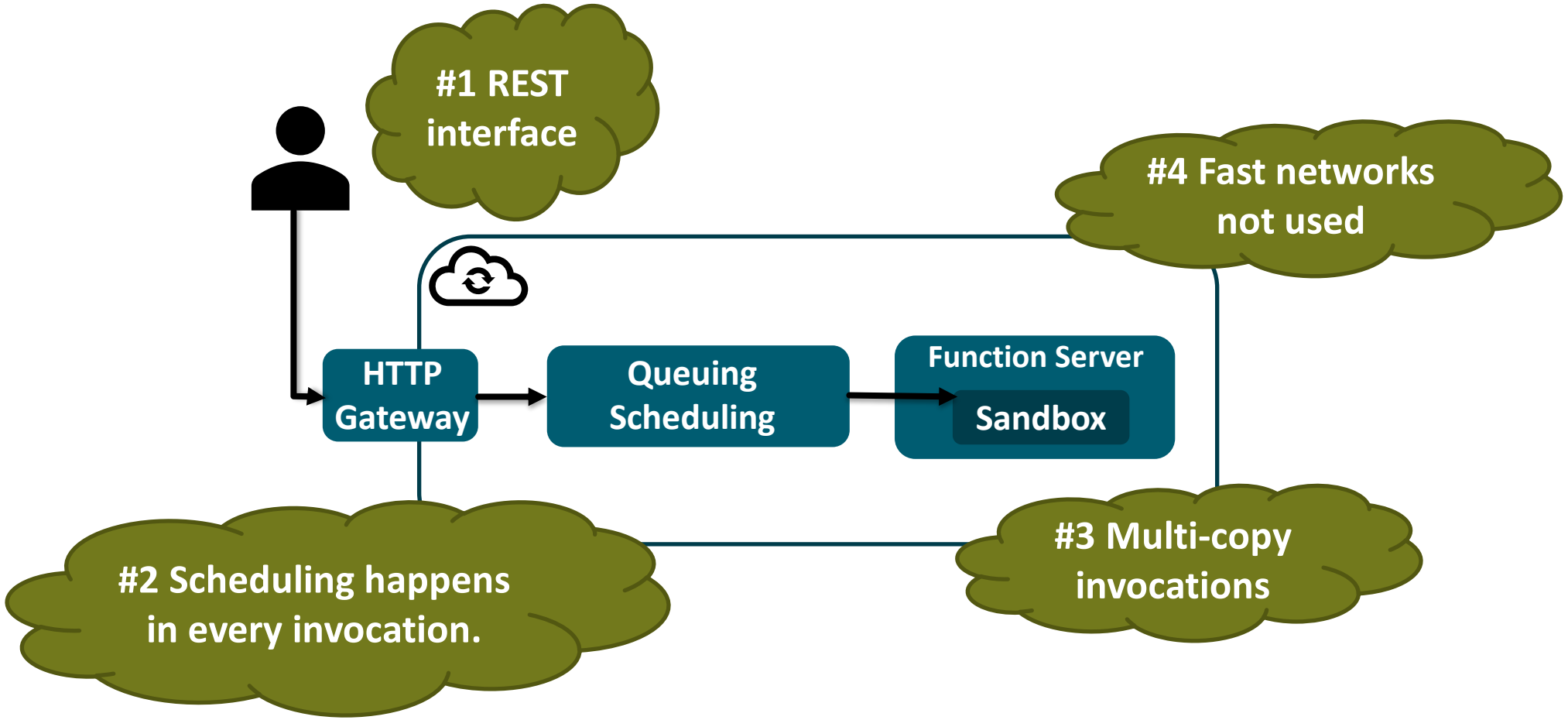
Why is FaaS slow?

IEEE IPDPS 2023



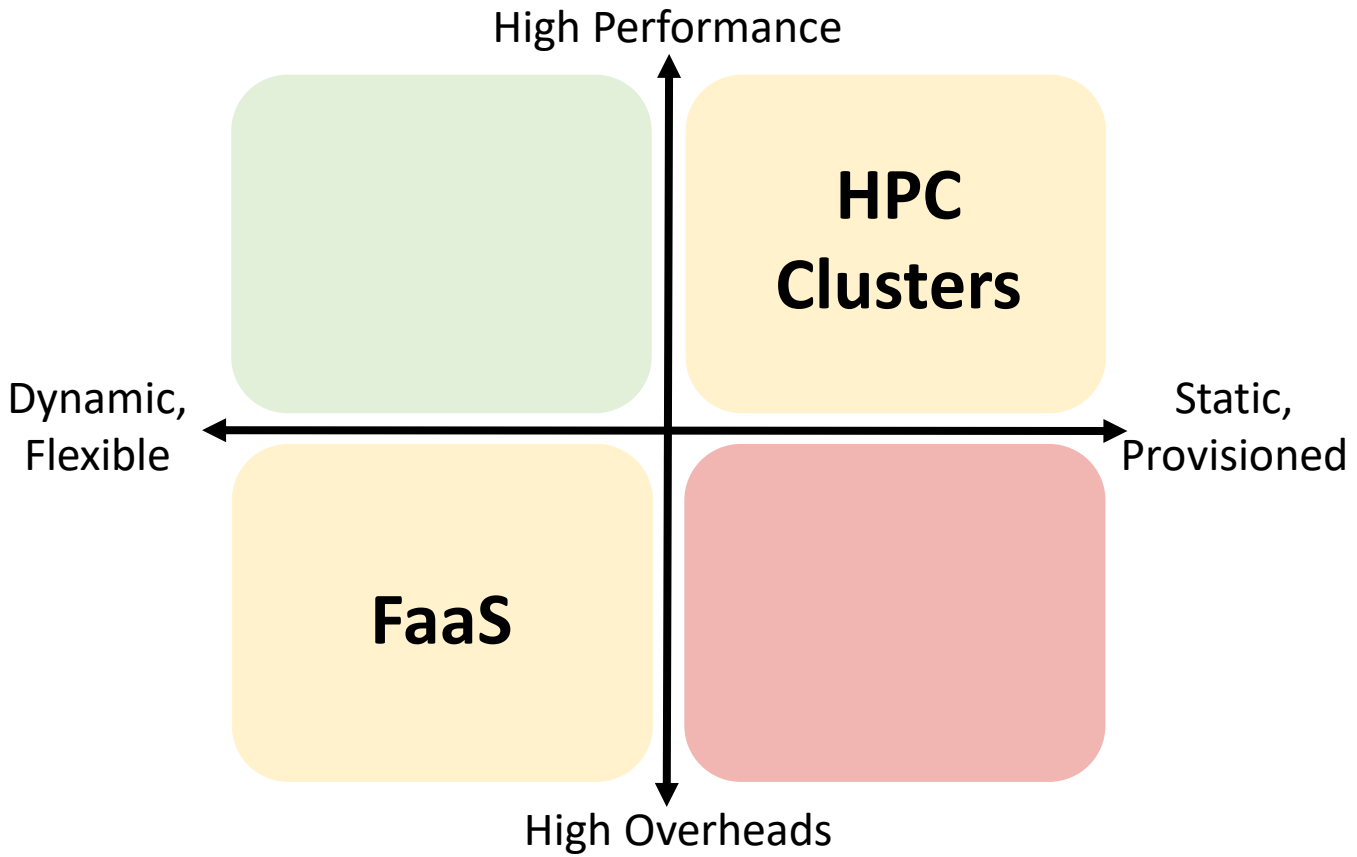
Why is FaaS slow?

IEEE IPDPS 2023



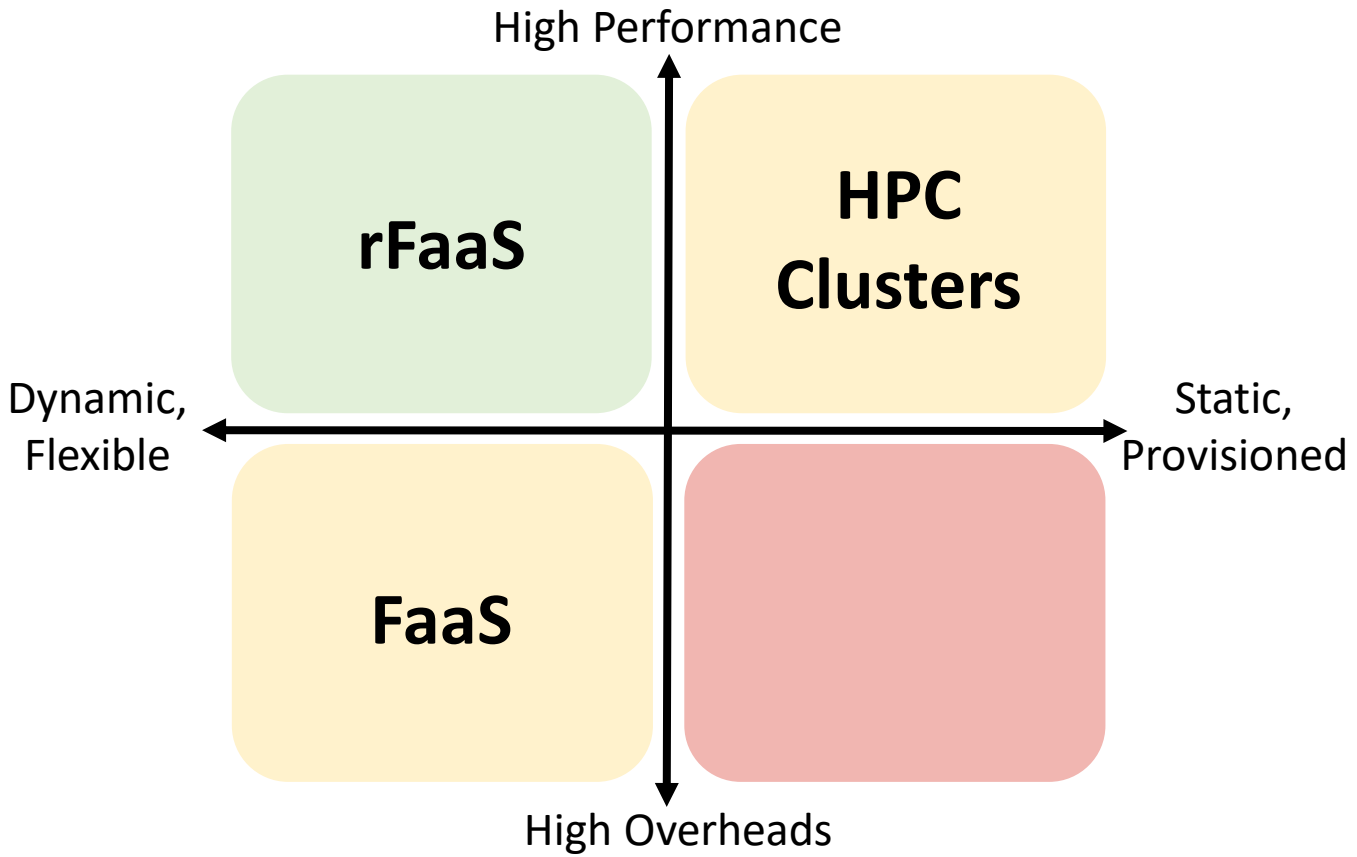
Function-as-a-Service for HPC

IEEE IPDPS 2023



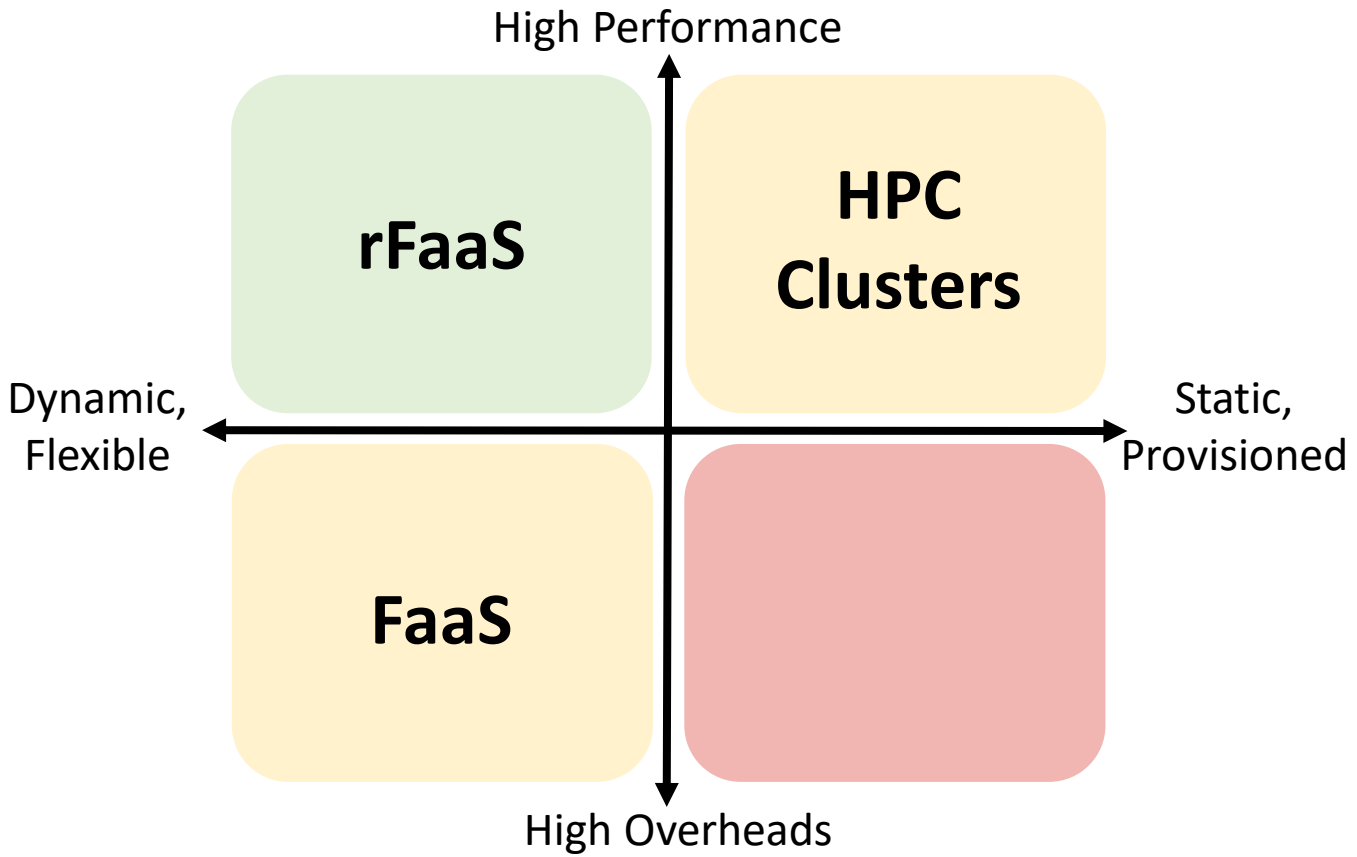
Function-as-a-Service for HPC

IEEE IPDPS 2023



Function-as-a-Service for HPC

IEEE IPDPS 2023



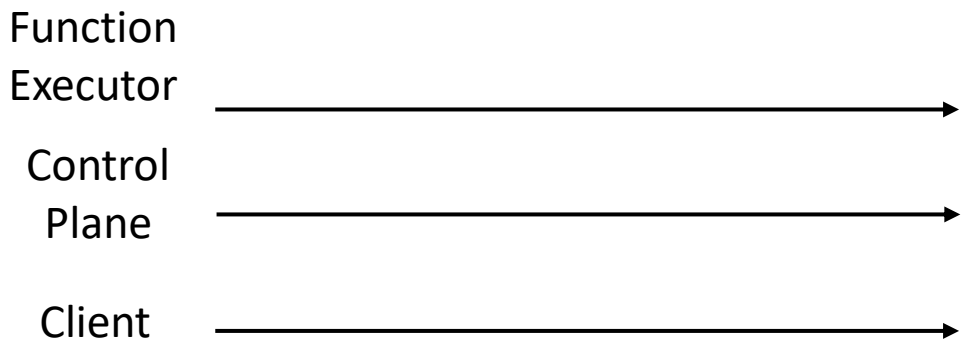
Reduced invocation critical path

Zero-copy RDMA networking

Invocations in FaaS and rFaaS

IEEE IPDPS 2023

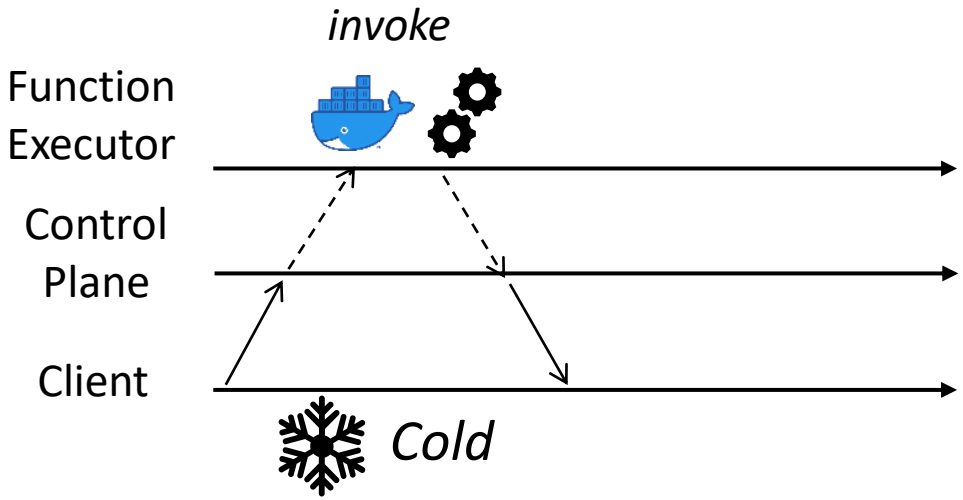
FaaS



Invocations in FaaS and rFaaS

IEEE IPDPS 2023

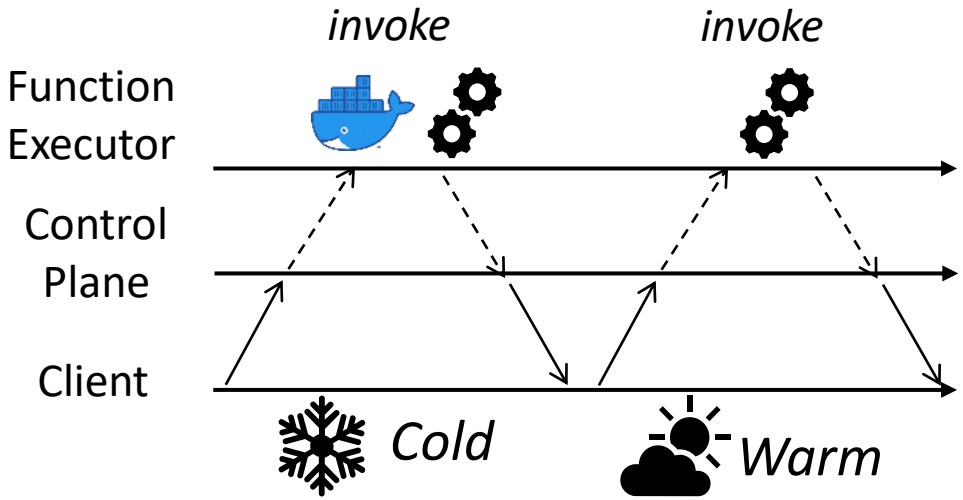
FaaS



Invocations in FaaS and rFaaS

IEEE IPDPS 2023

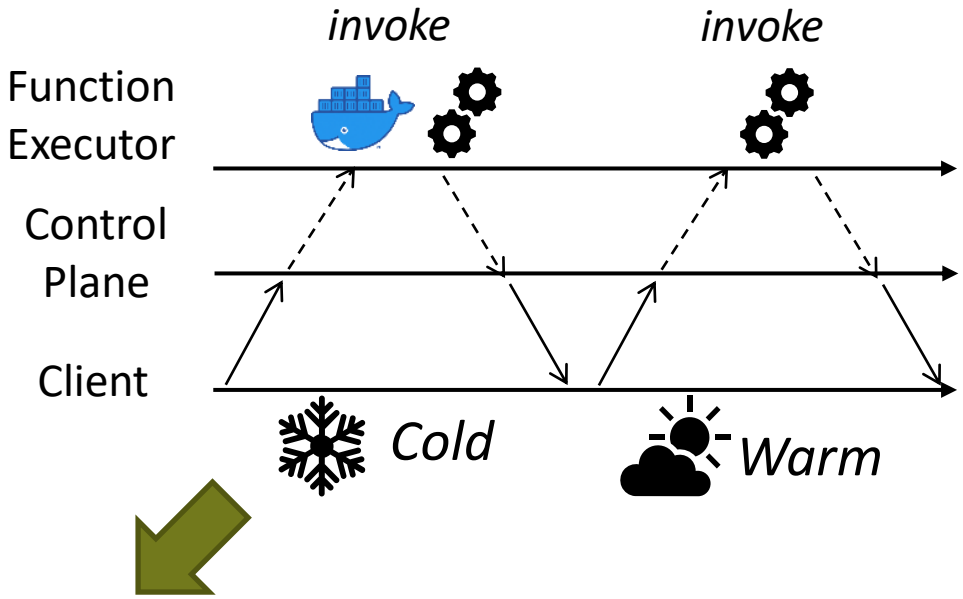
FaaS



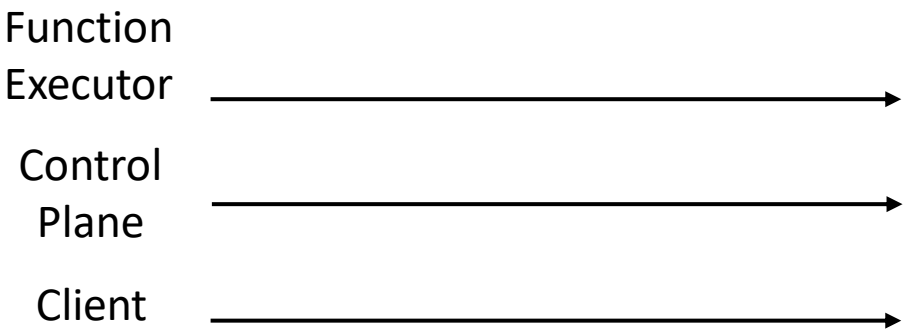
Invocations in FaaS and rFaaS

IEEE IPDPS 2023

FaaS



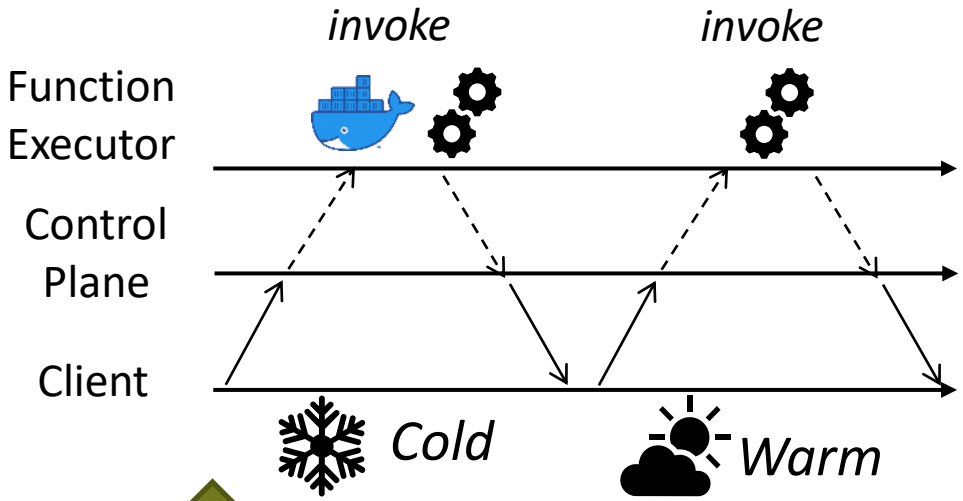
rFaaS



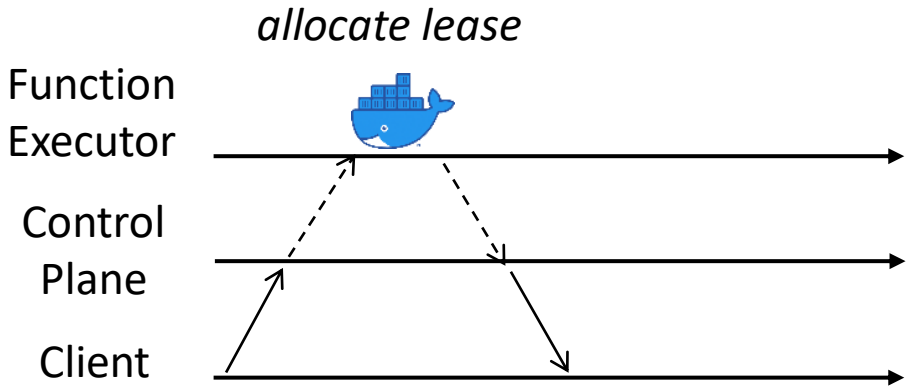
Invocations in FaaS and rFaaS

IEEE IPDPS 2023

FaaS



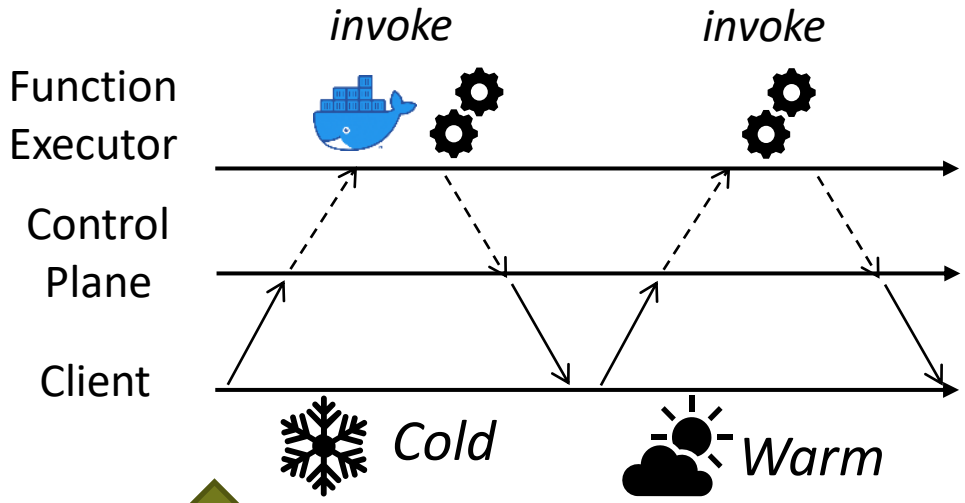
rFaaS



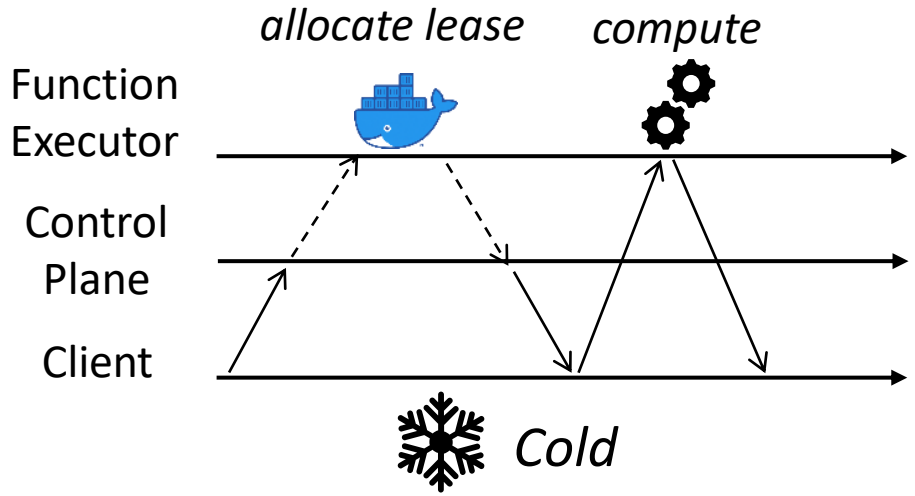
Invocations in FaaS and rFaaS

IEEE IPDPS 2023

FaaS



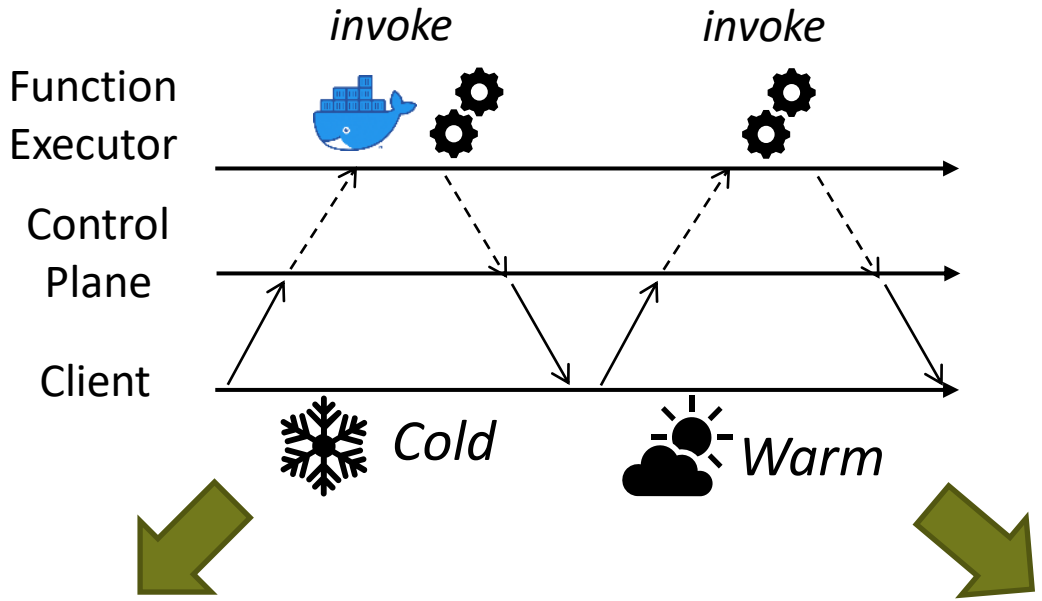
rFaaS



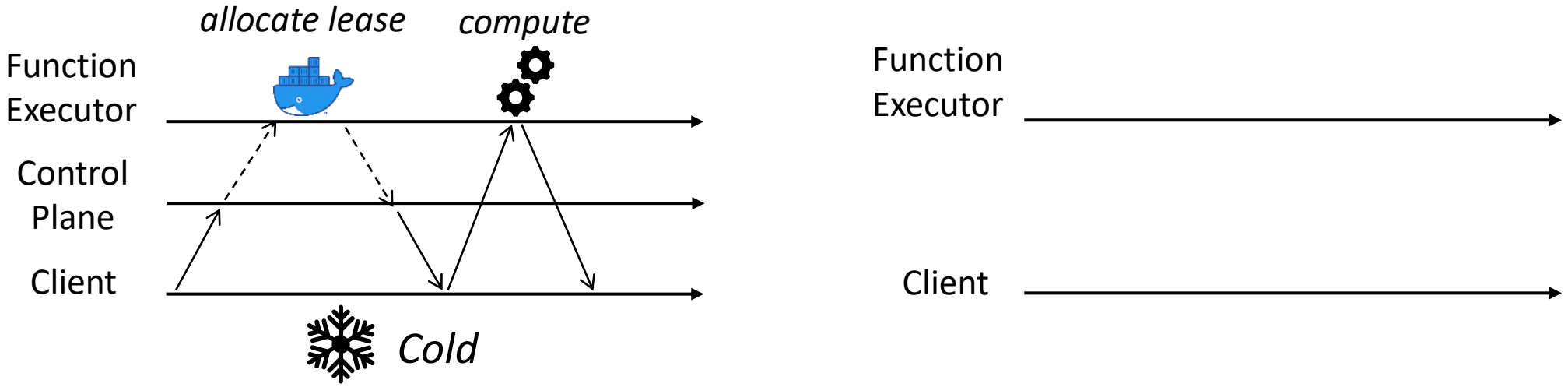
Invocations in FaaS and rFaaS

IEEE IPDPS 2023

FaaS



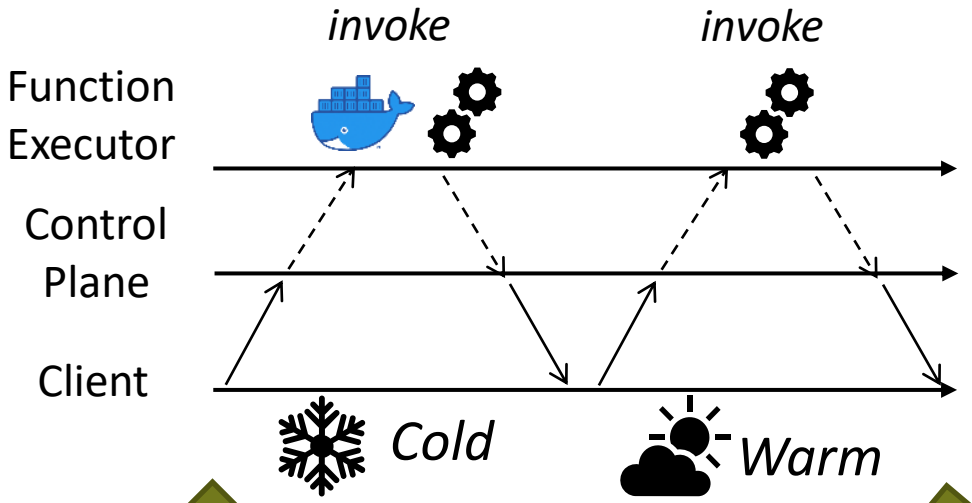
rFaaS



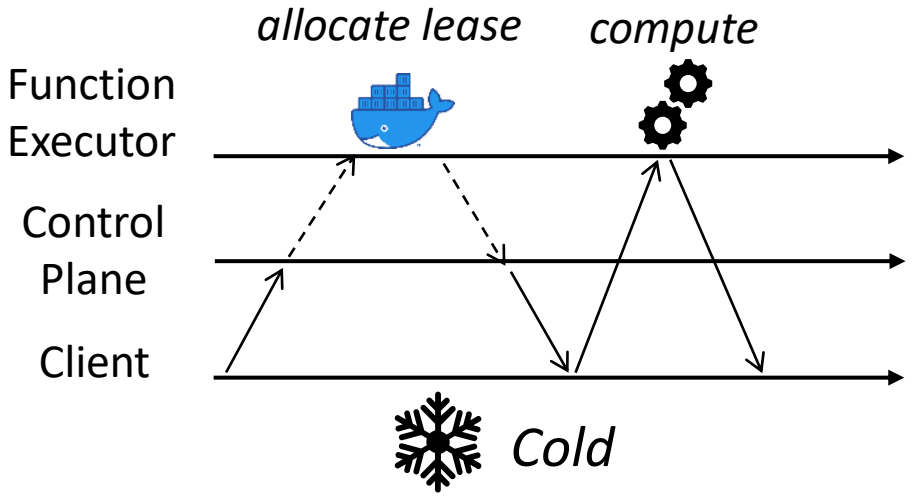
Invocations in FaaS and rFaaS

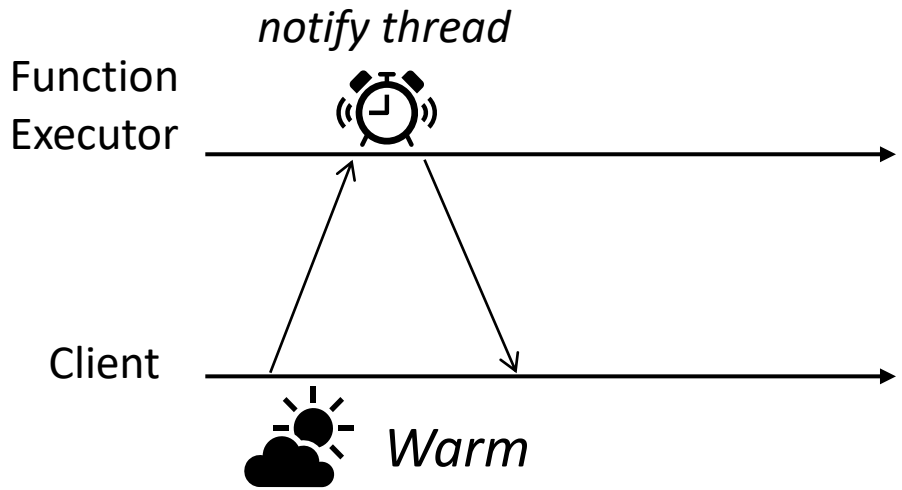
IEEE IPDPS 2023

FaaS



rFaaS

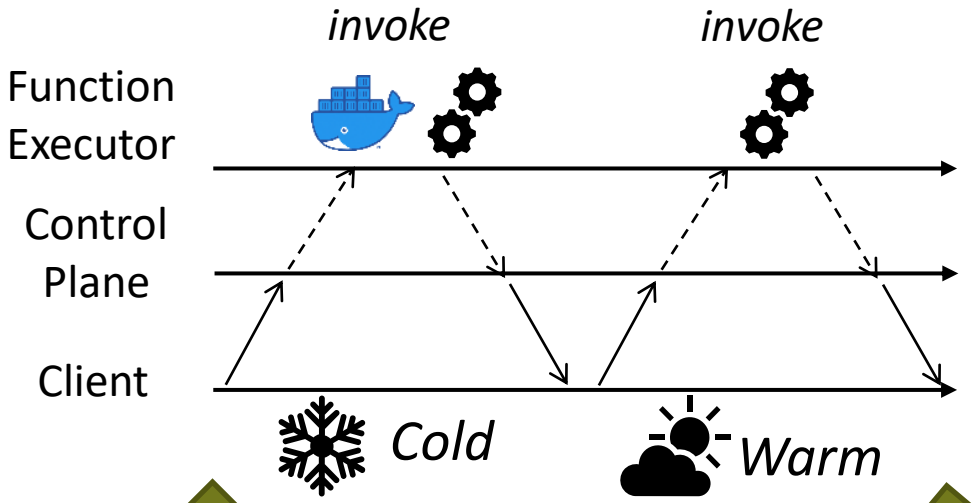




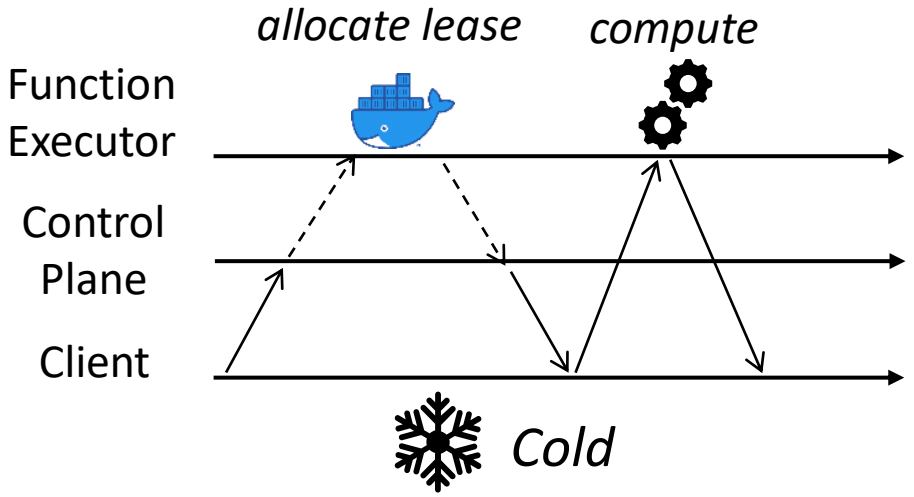
Invocations in FaaS and rFaaS

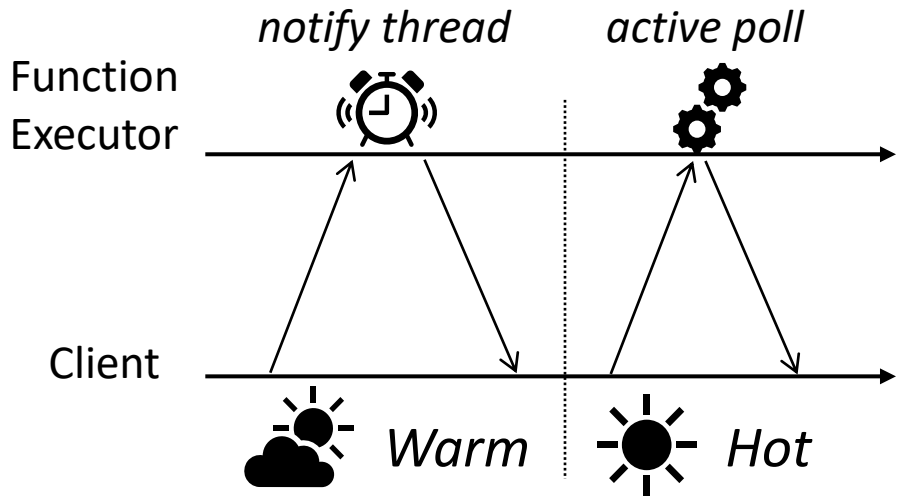
IEEE IPDPS 2023

FaaS



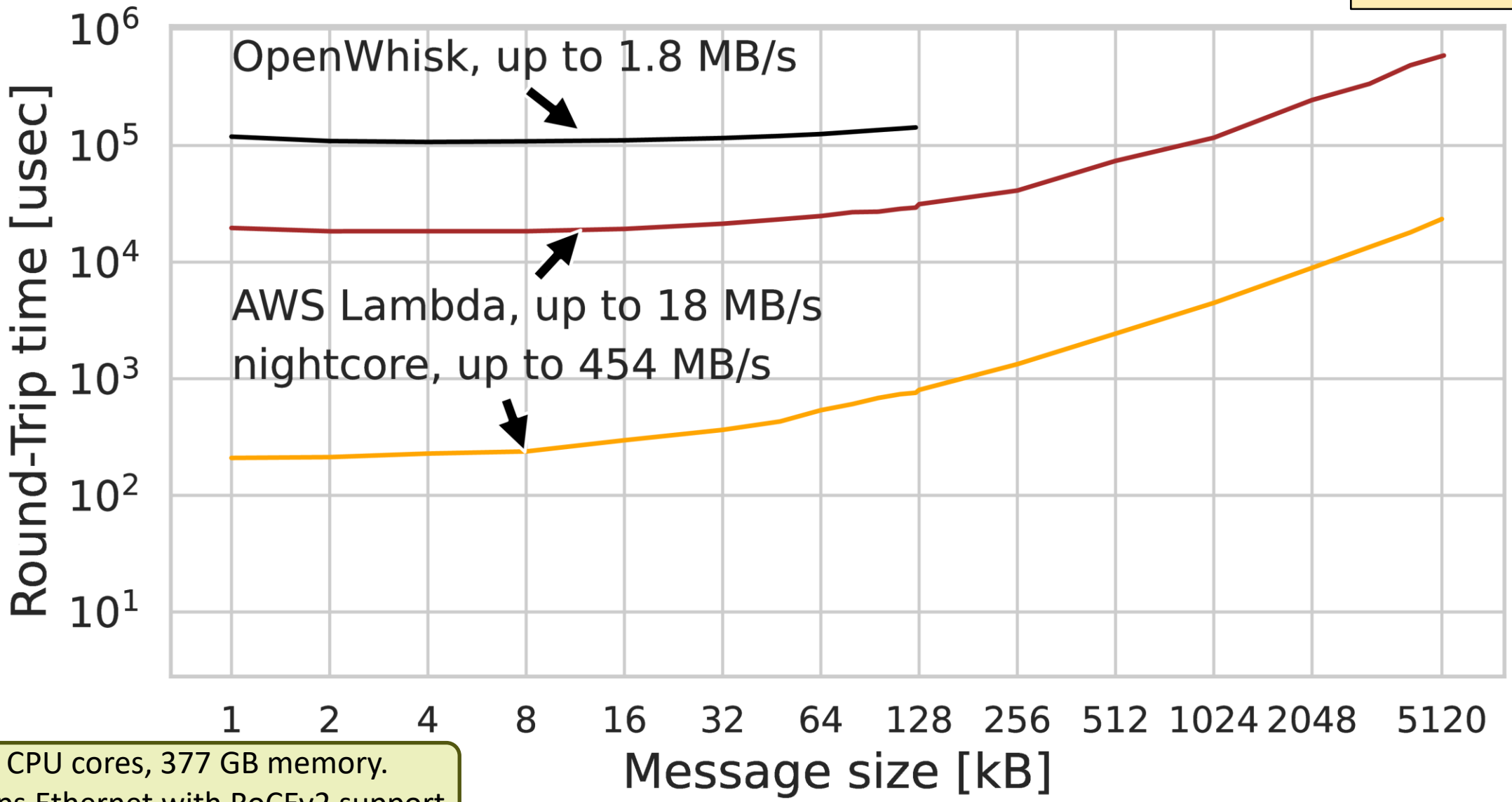
rFaaS





How fast are invocations in FaaS?

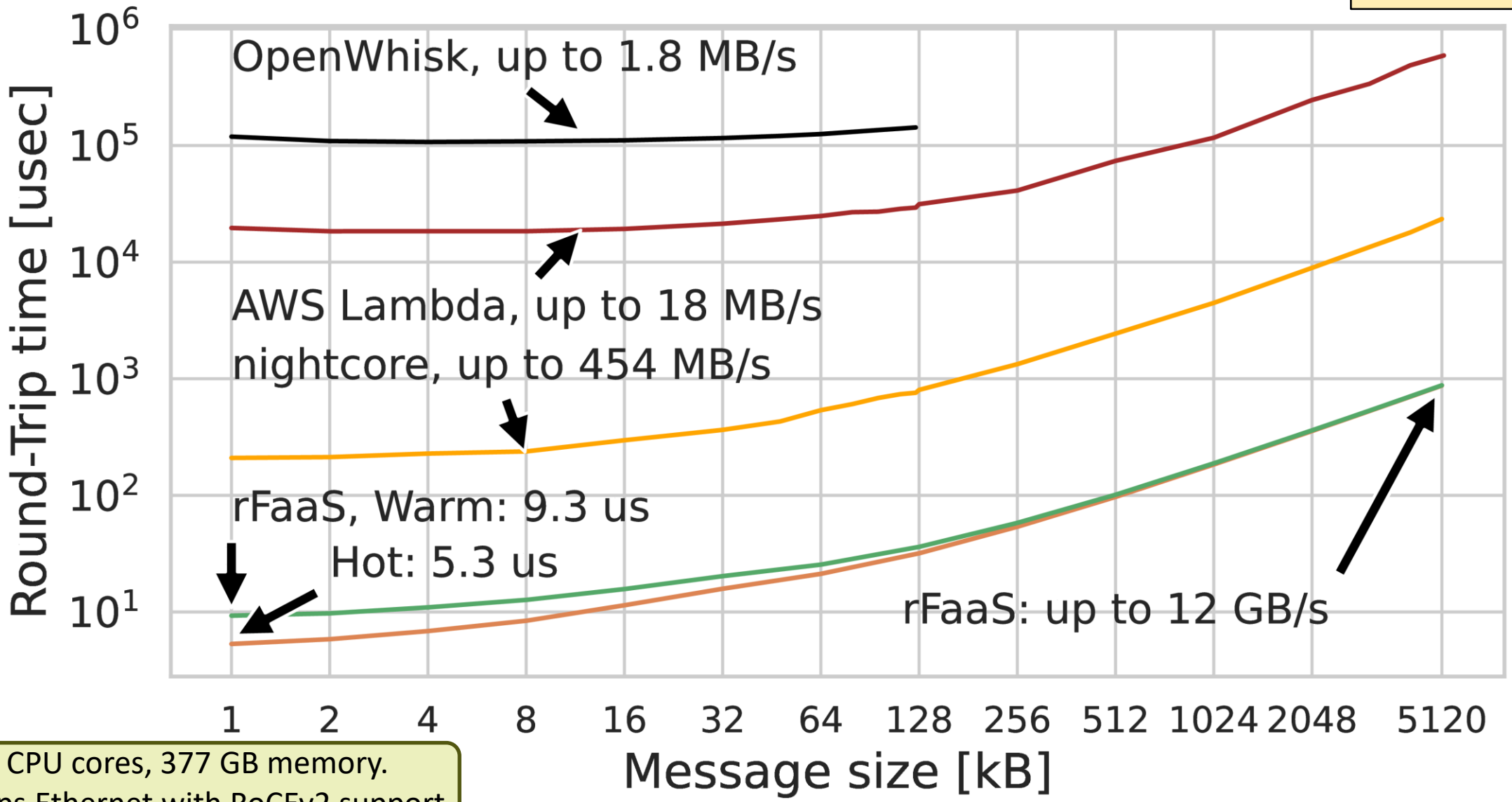
IEEE IPDPS 2023



36 CPU cores, 377 GB memory.
100 Gbps Ethernet with RoCEv2 support.

How fast are invocations in FaaS?

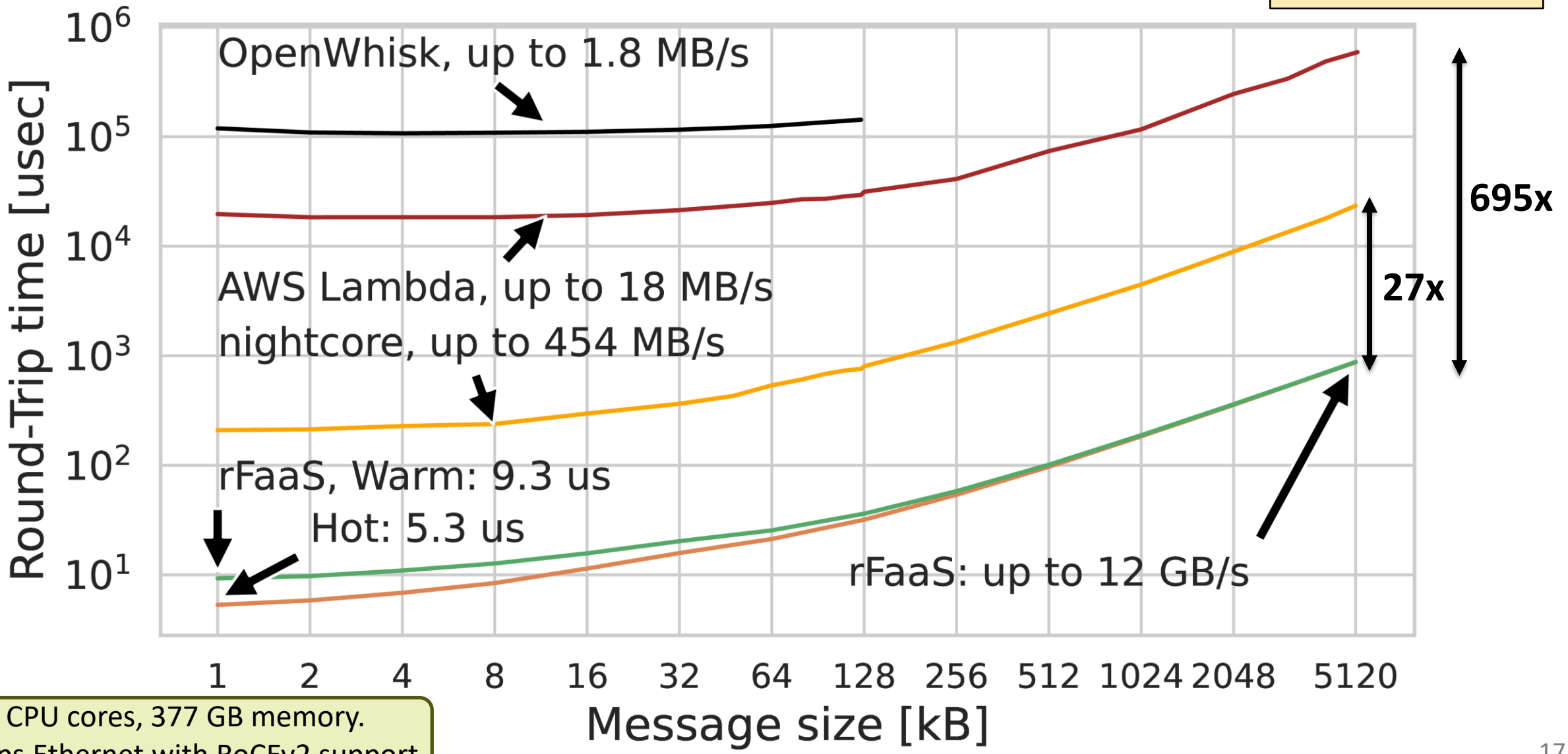
IEEE IPDPS 2023



36 CPU cores, 377 GB memory.
100 Gbps Ethernet with RoCEv2 support.

How fast are invocations in FaaS?

IEEE IPDPS 2023

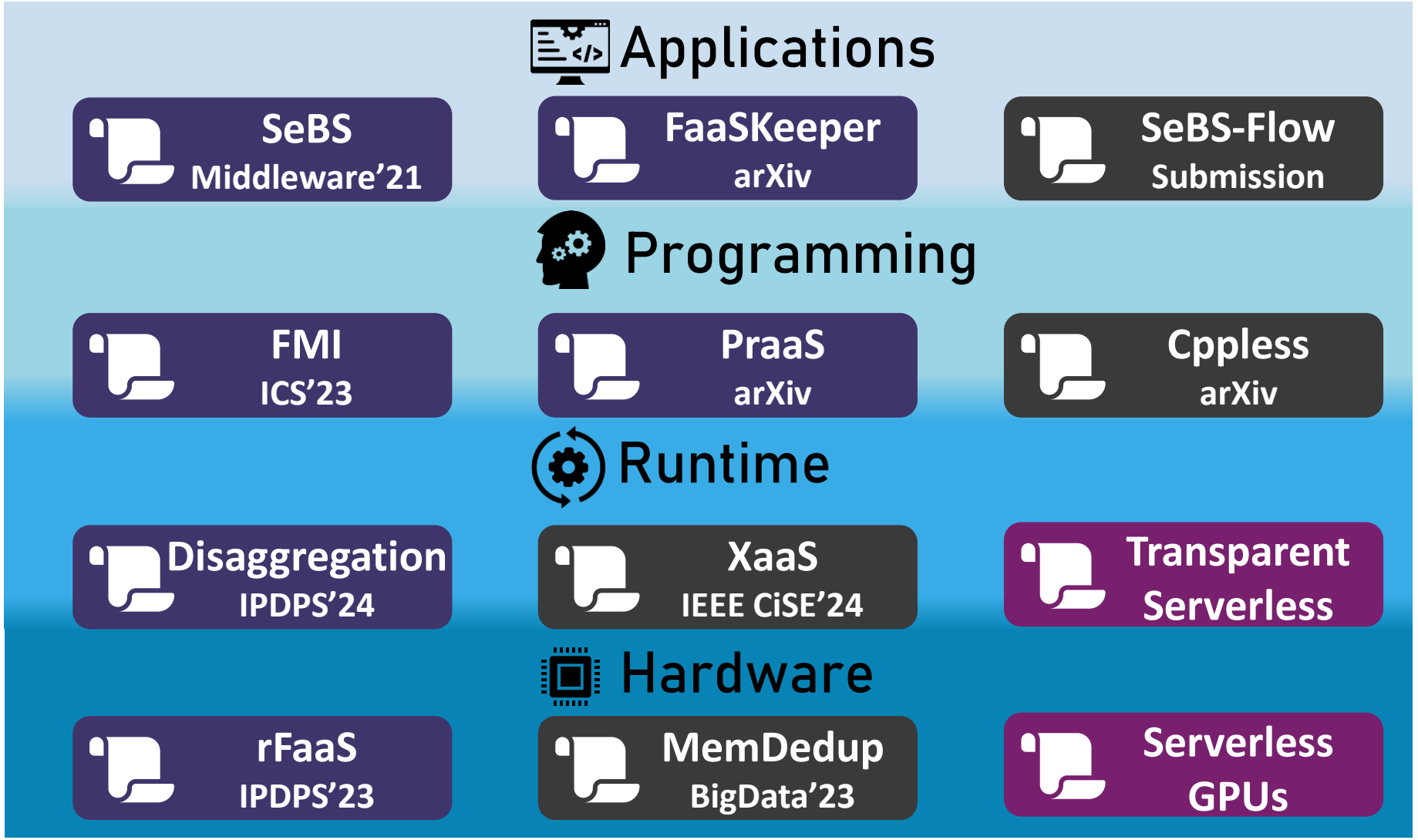


36 CPU cores, 377 GB memory.
100 Gbps Ethernet with RoCEv2 support.

High-Performance Serverless Stack

Multi-platform benchmarking suite.

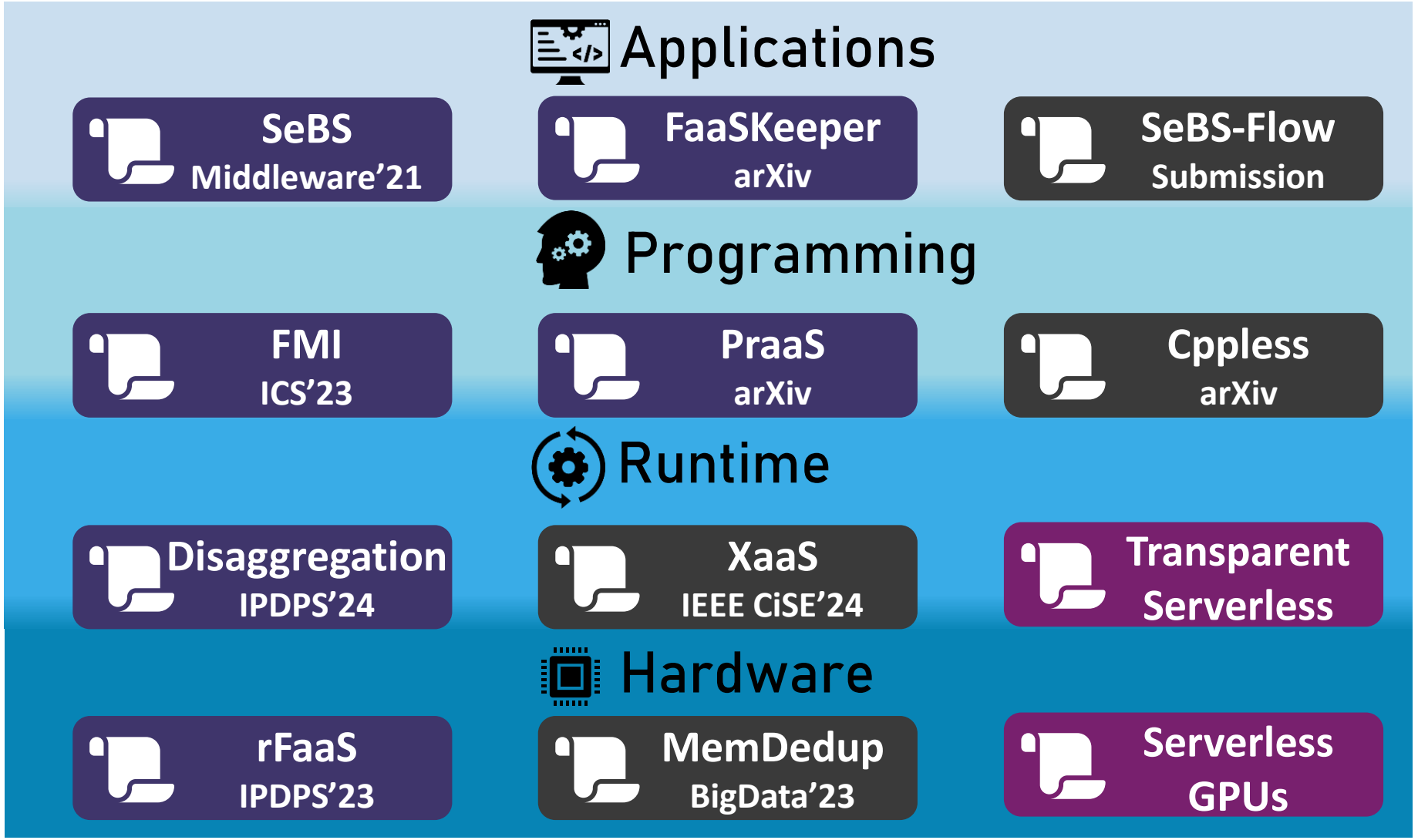
Functions are expensive to invoke.



High-Performance Serverless Stack

Multi-platform benchmarking suite.

Fast invocations with RDMA acceleration.

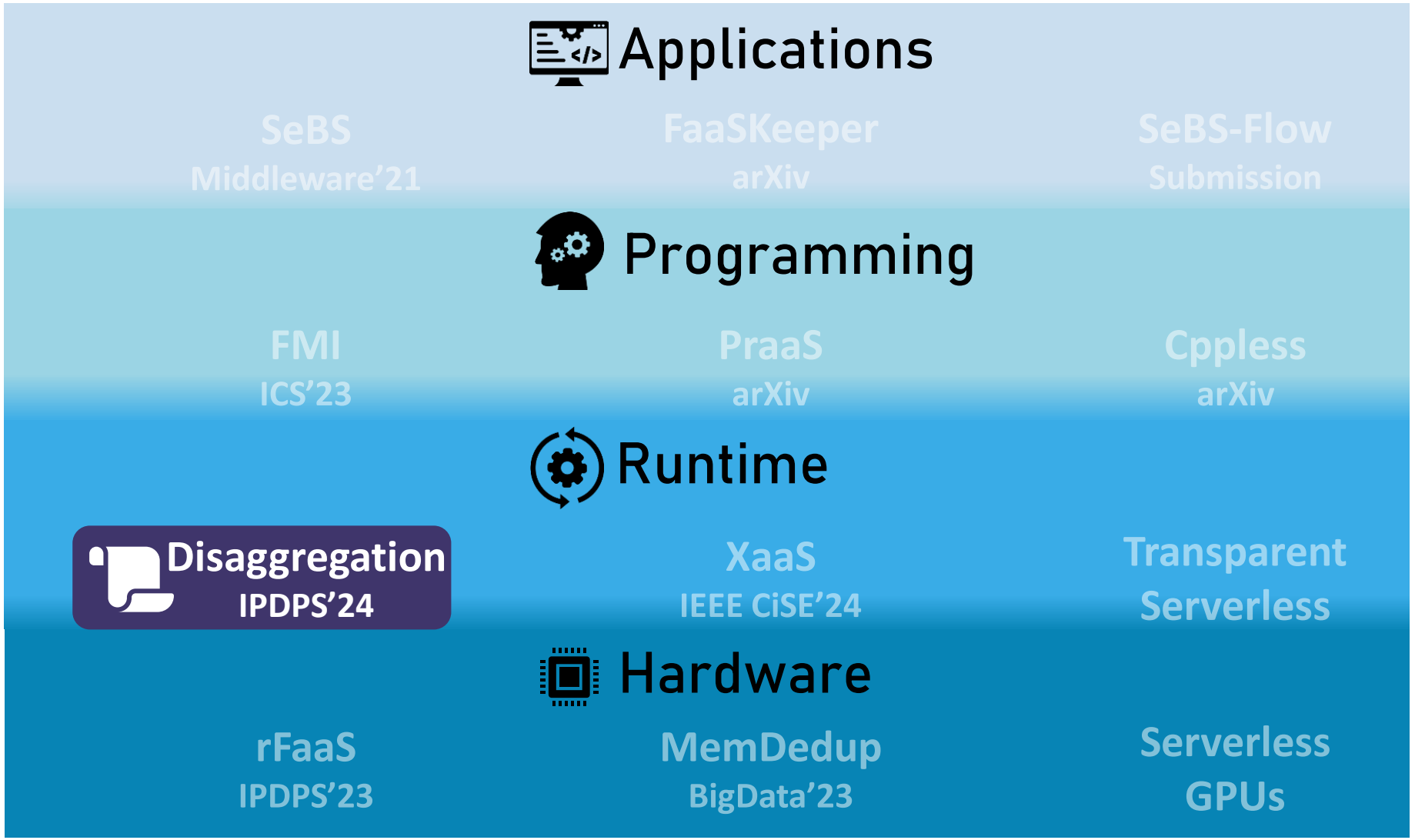


High-Performance Serverless Stack

Multi-platform benchmarking suite.

Fast invocations with RDMA acceleration.

How can serverless improve HPC?



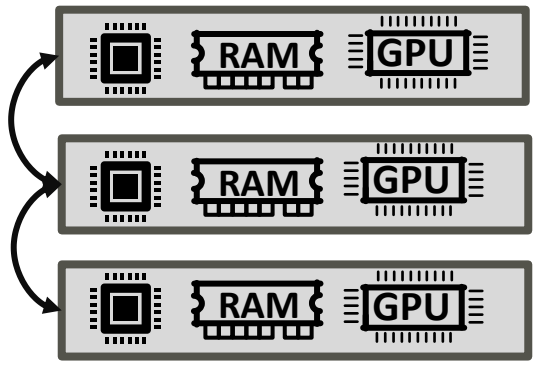
Software Solution

IEEE IPDPS 2024
ACM SRC @ SC 22

Software Solution

IEEE IPDPS 2024
ACM SRC @ SC 22

Standard HPC Node

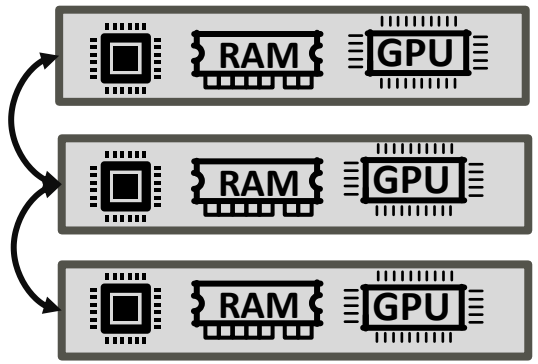


- High performance
- Inflexible architecture

Software Solution

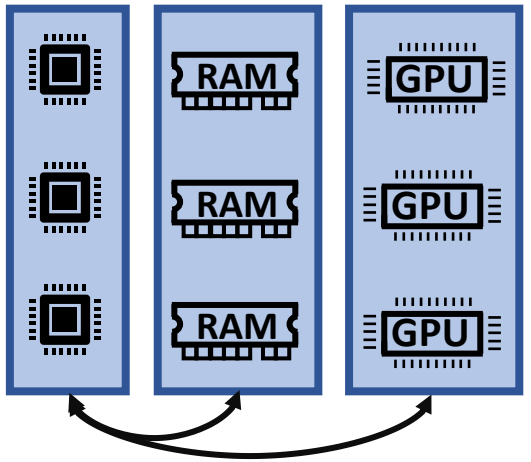
IEEE IPDPS 2024
ACM SRC @ SC 22

Standard HPC Node



- High performance
- Inflexible architecture

Hardware Disaggregation

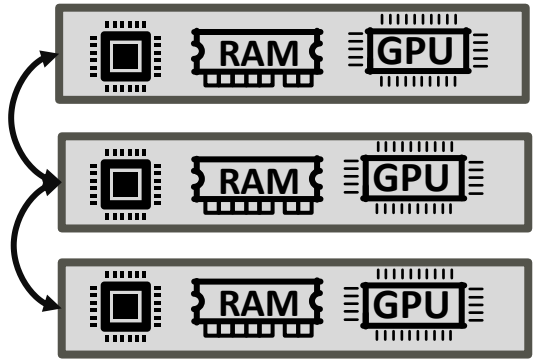


- High utilization
- Cost, performance penalty

Software Solution

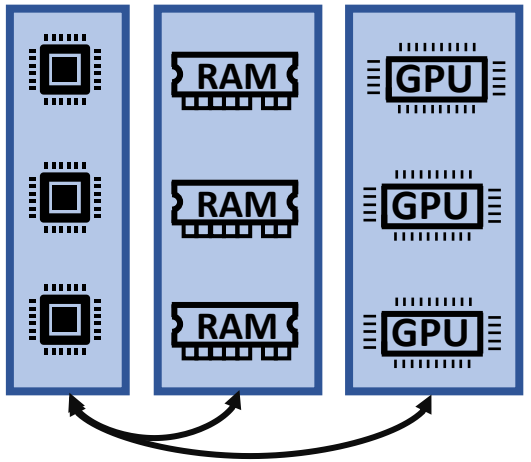
IEEE IPDPS 2024
ACM SRC @ SC 22

Standard HPC Node



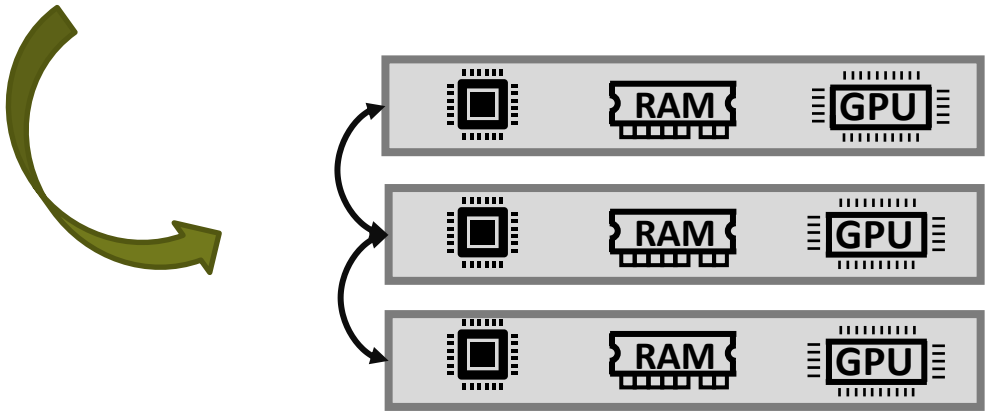
- High performance
- Inflexible architecture

Hardware Disaggregation



- High utilization
- Cost, performance penalty

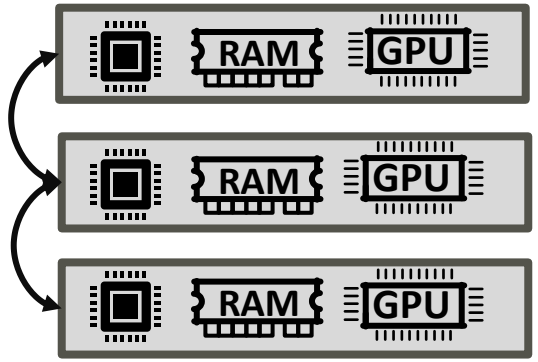
Existing Coupled Hardware Systems



Software Solution

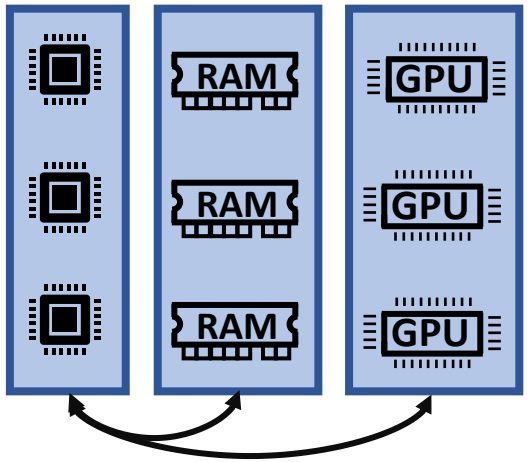
IEEE IPDPS 2024
ACM SRC @ SC 22

Standard HPC Node



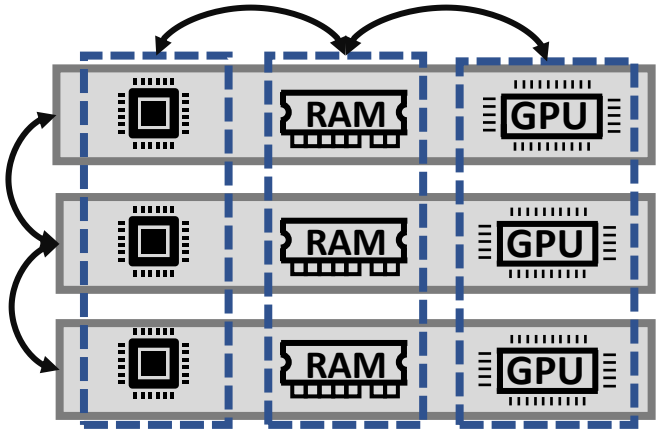
- High performance
- Inflexible architecture

Hardware Disaggregation



- High utilization
- Cost, performance penalty

Existing Coupled Hardware Systems

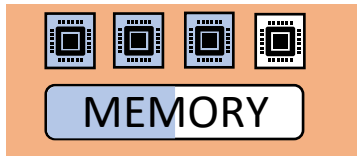


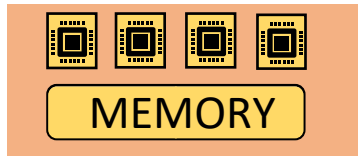
Software Abstraction for Disaggregation

Serving Remote Memory

IEEE IPDPS 2024
ACM SRC @ SC 22

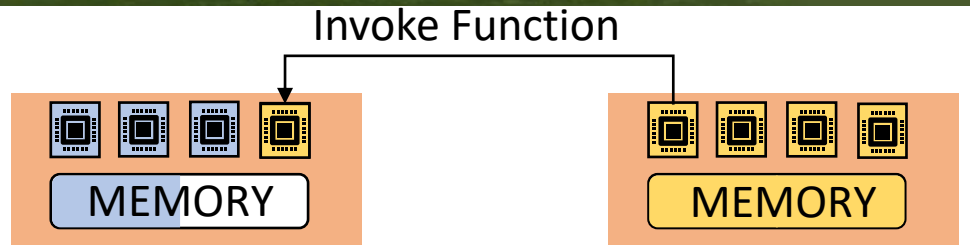
Serving Remote Memory





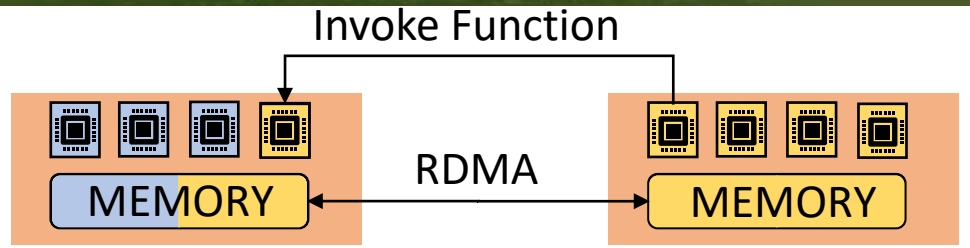
IEEE IPDPS 2024
ACM SRC @ SC 22

Serving Remote Memory



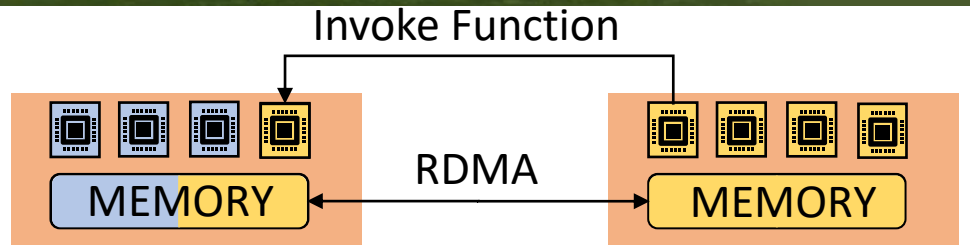
IEEE IPDPS 2024
ACM SRC @ SC 22

Serving Remote Memory



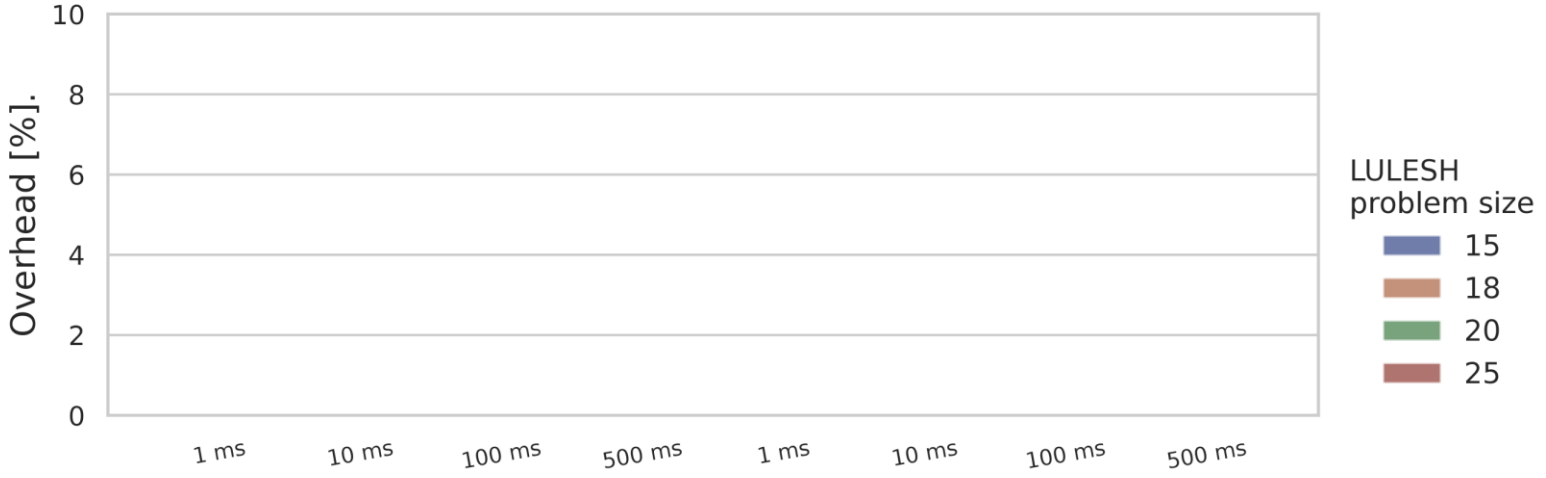
IEEE IPDPS 2024
ACM SRC @ SC 22

Serving Remote Memory

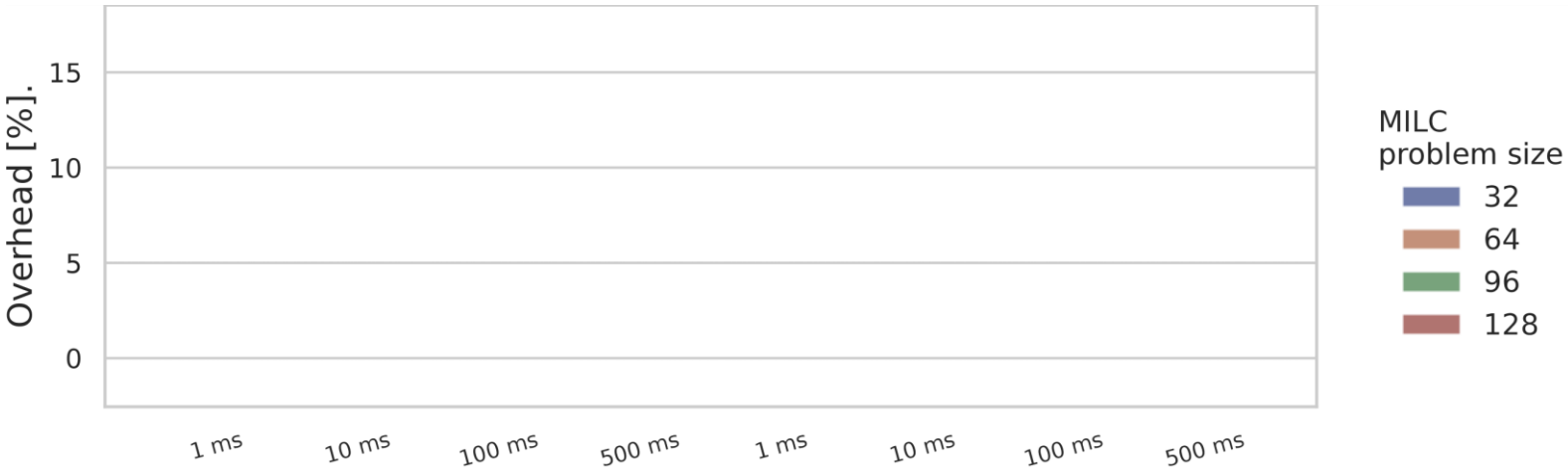


IEEE IPDPS 2024
ACM SRC @ SC 22

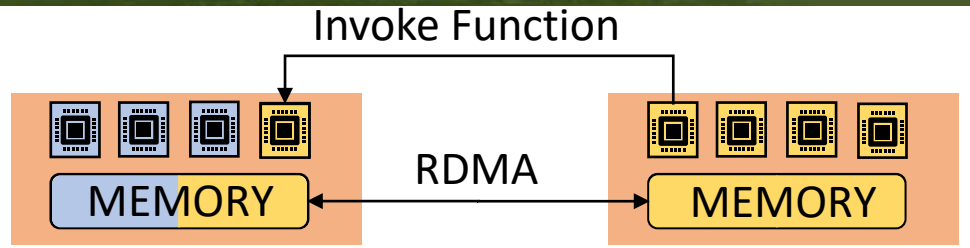
LULESH
125 ranks



MILC
32 ranks

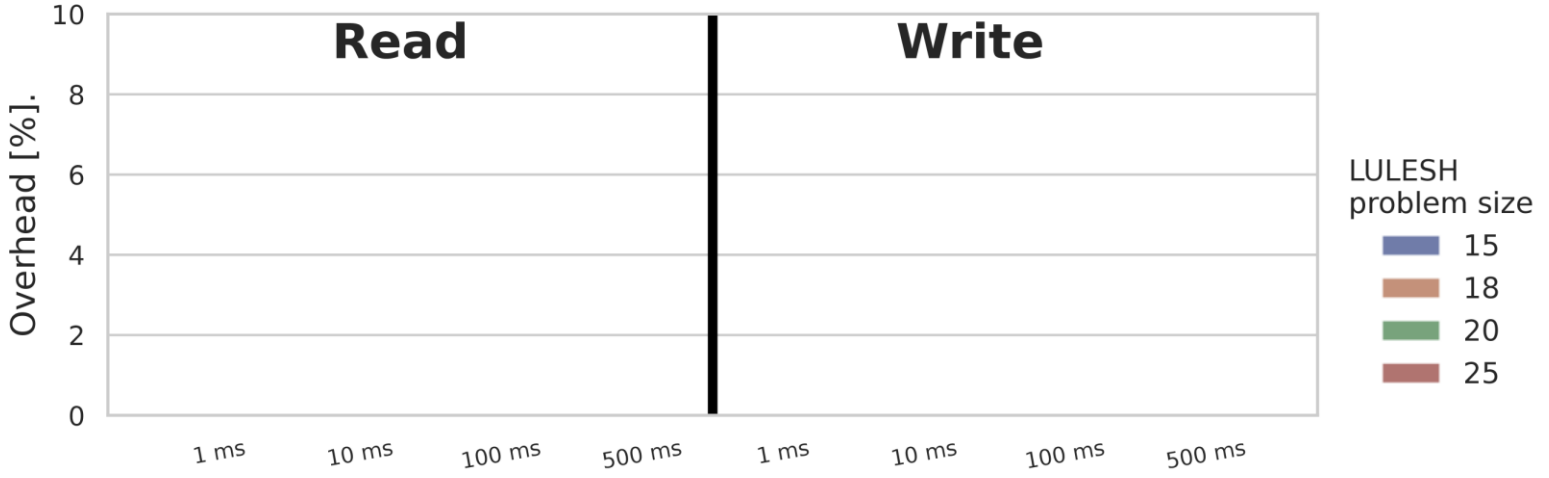


Serving Remote Memory

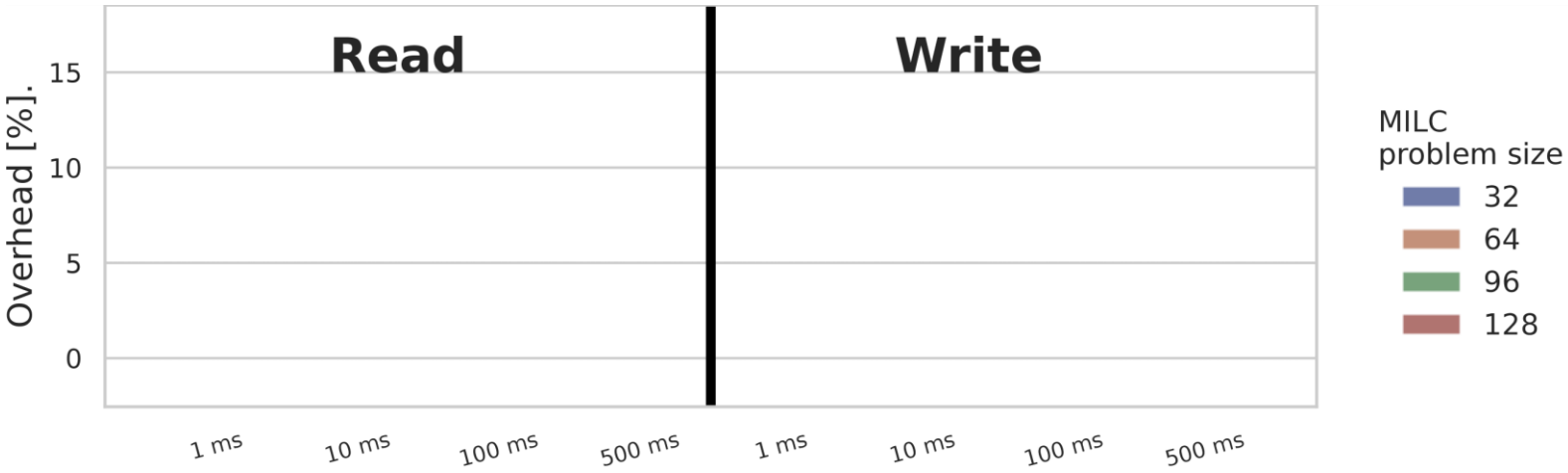


IEEE IPDPS 2024
ACM SRC @ SC 22

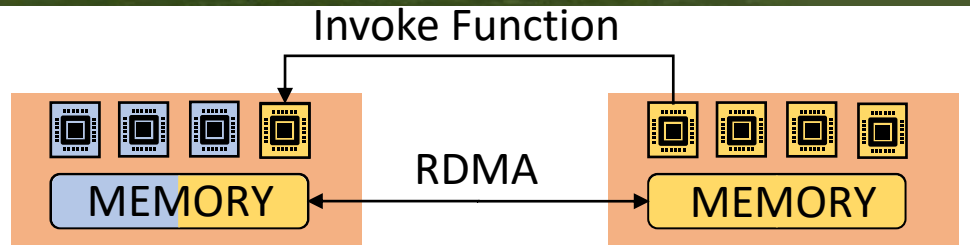
LULESH
125 ranks



MILC
32 ranks

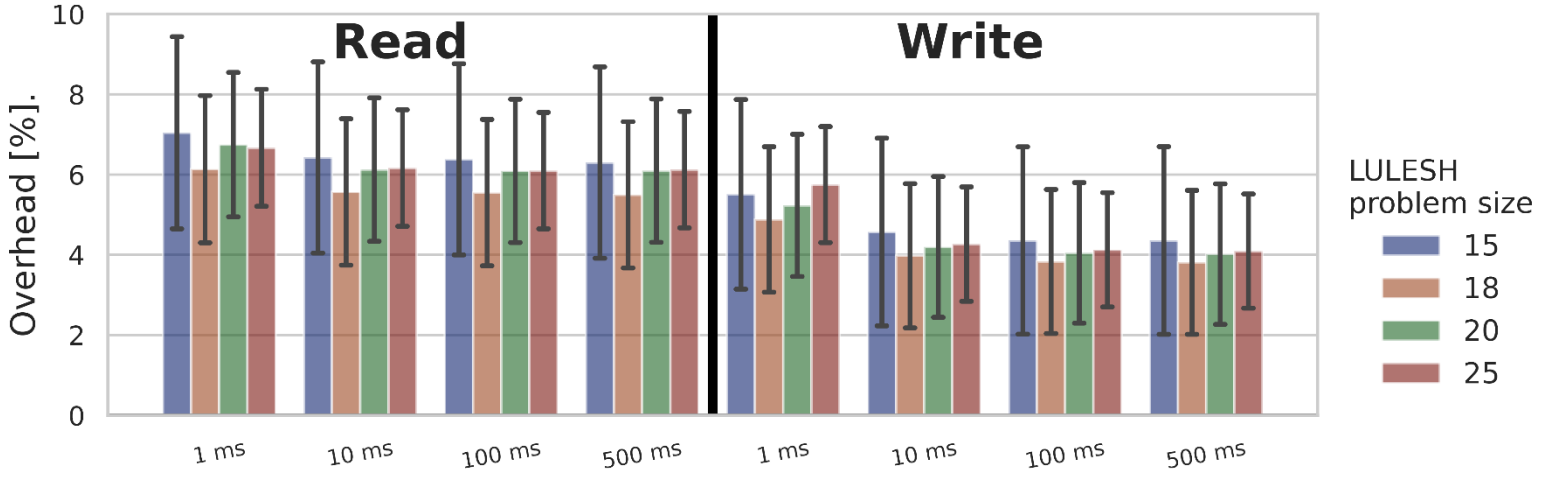


Serving Remote Memory

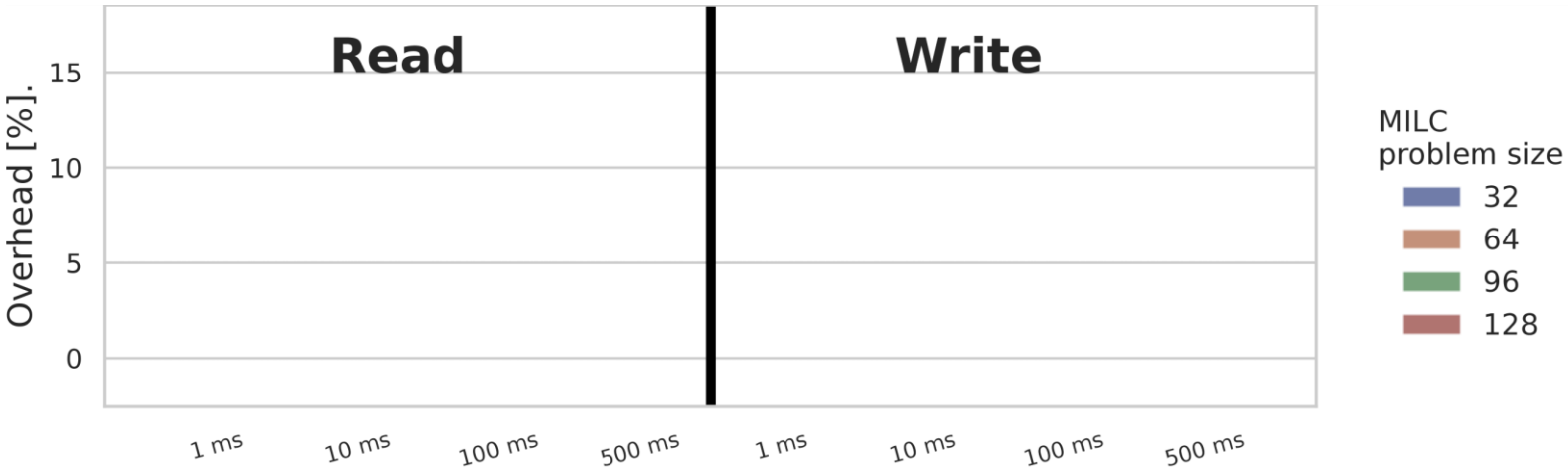


IEEE IPDPS 2024
ACM SRC @ SC 22

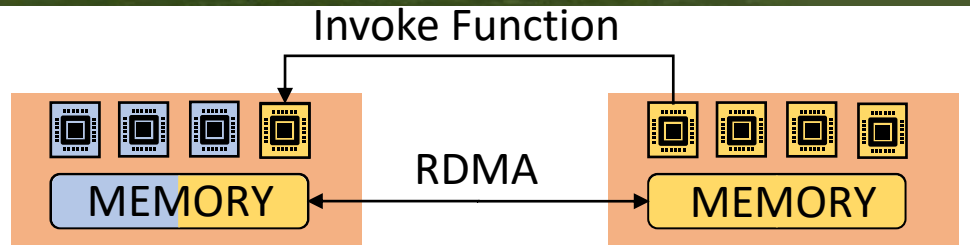
LULESH
125 ranks



MILC
32 ranks

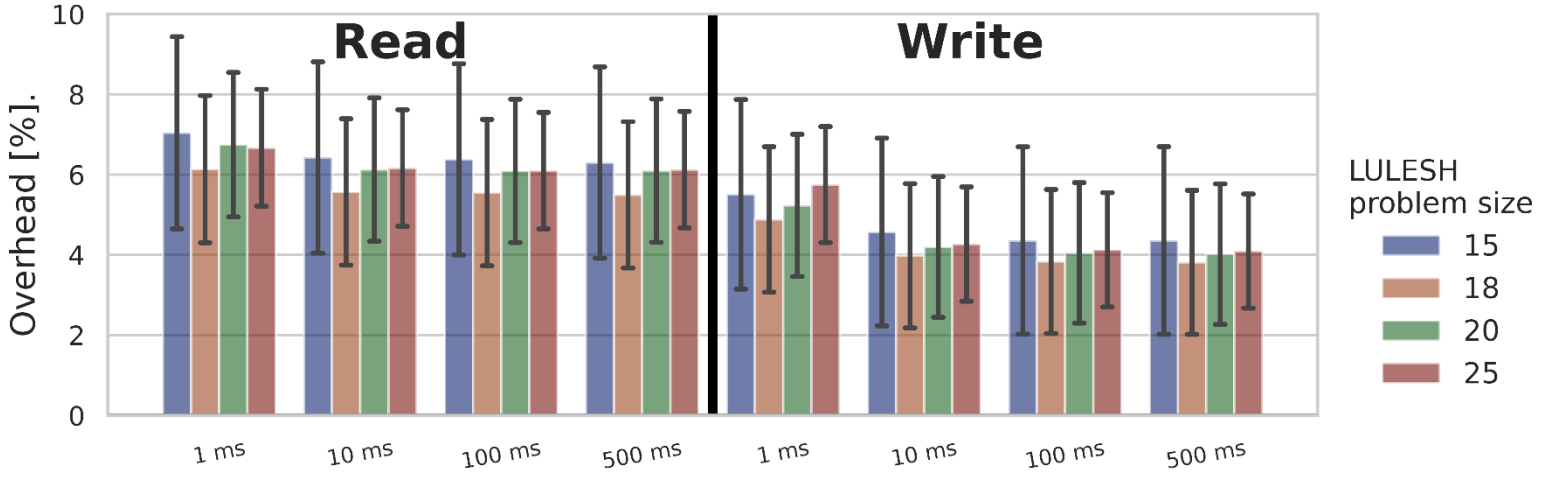


Serving Remote Memory

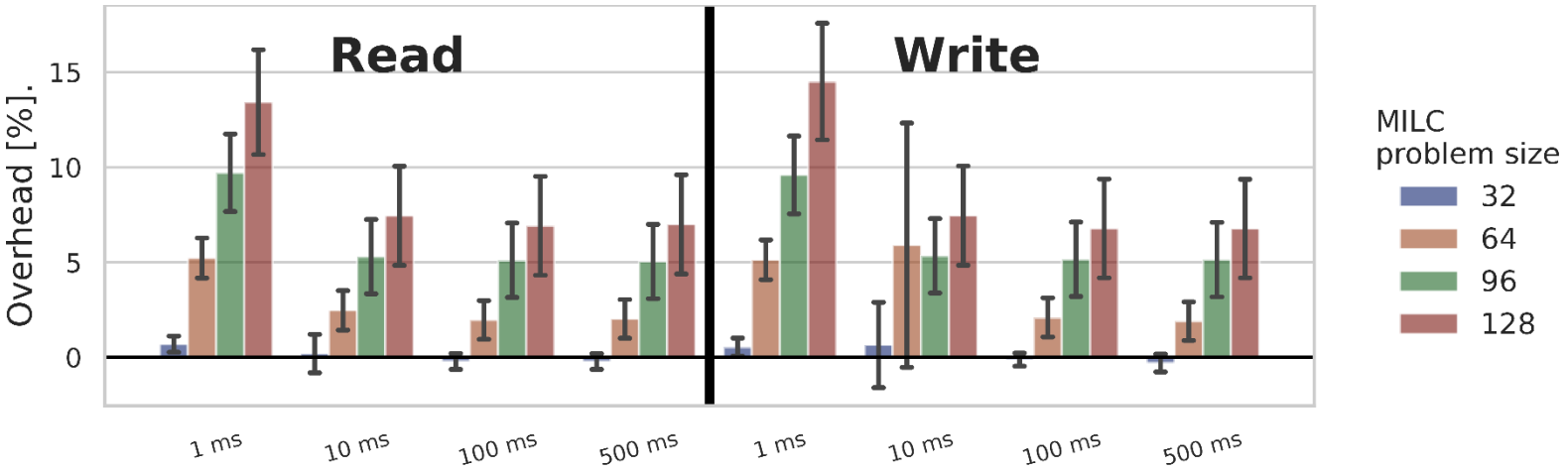


IEEE IPDPS 2024
ACM SRC @ SC 22

LULESH
125 ranks



MILC
32 ranks

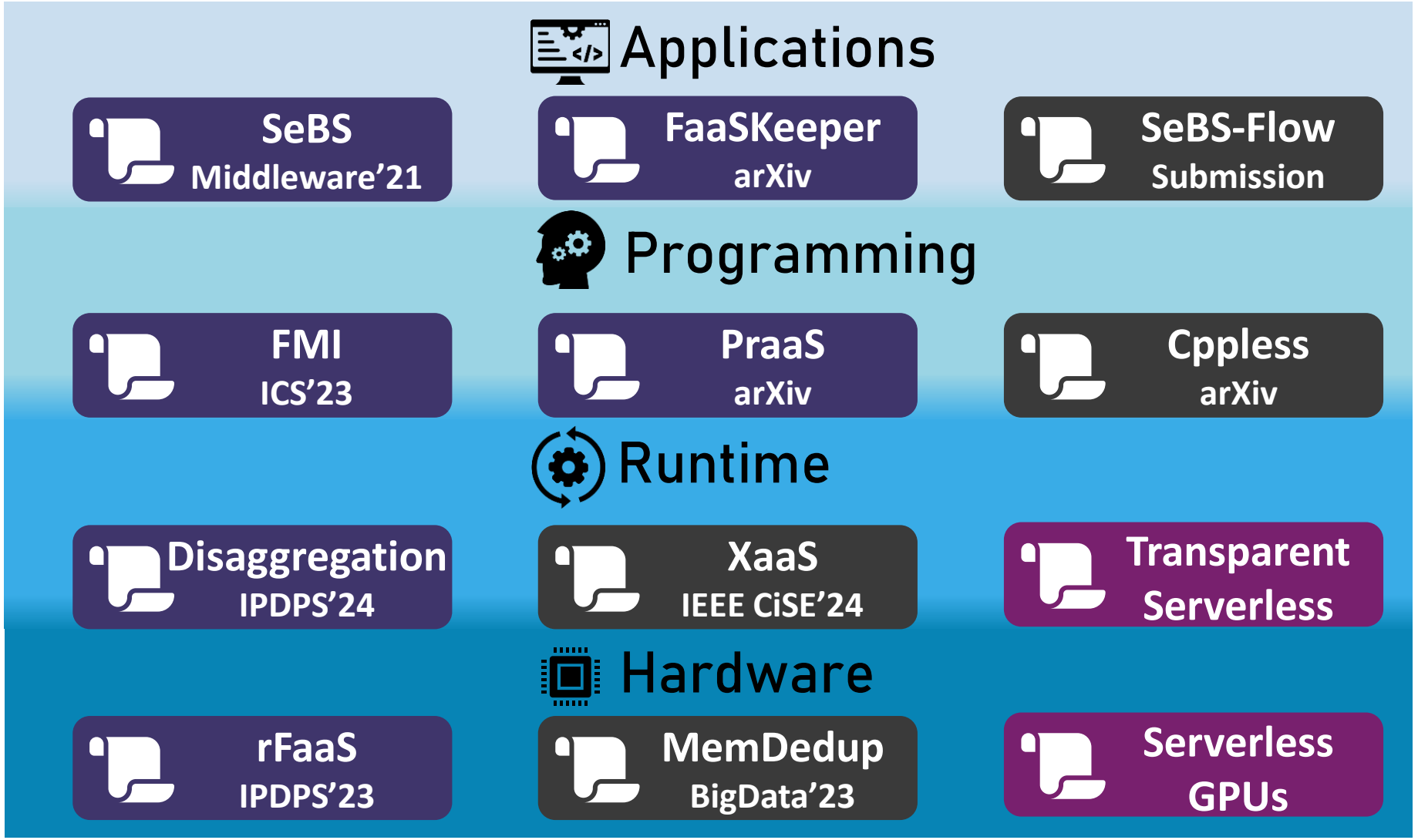


High-Performance Serverless Stack

Multi-platform benchmarking suite.

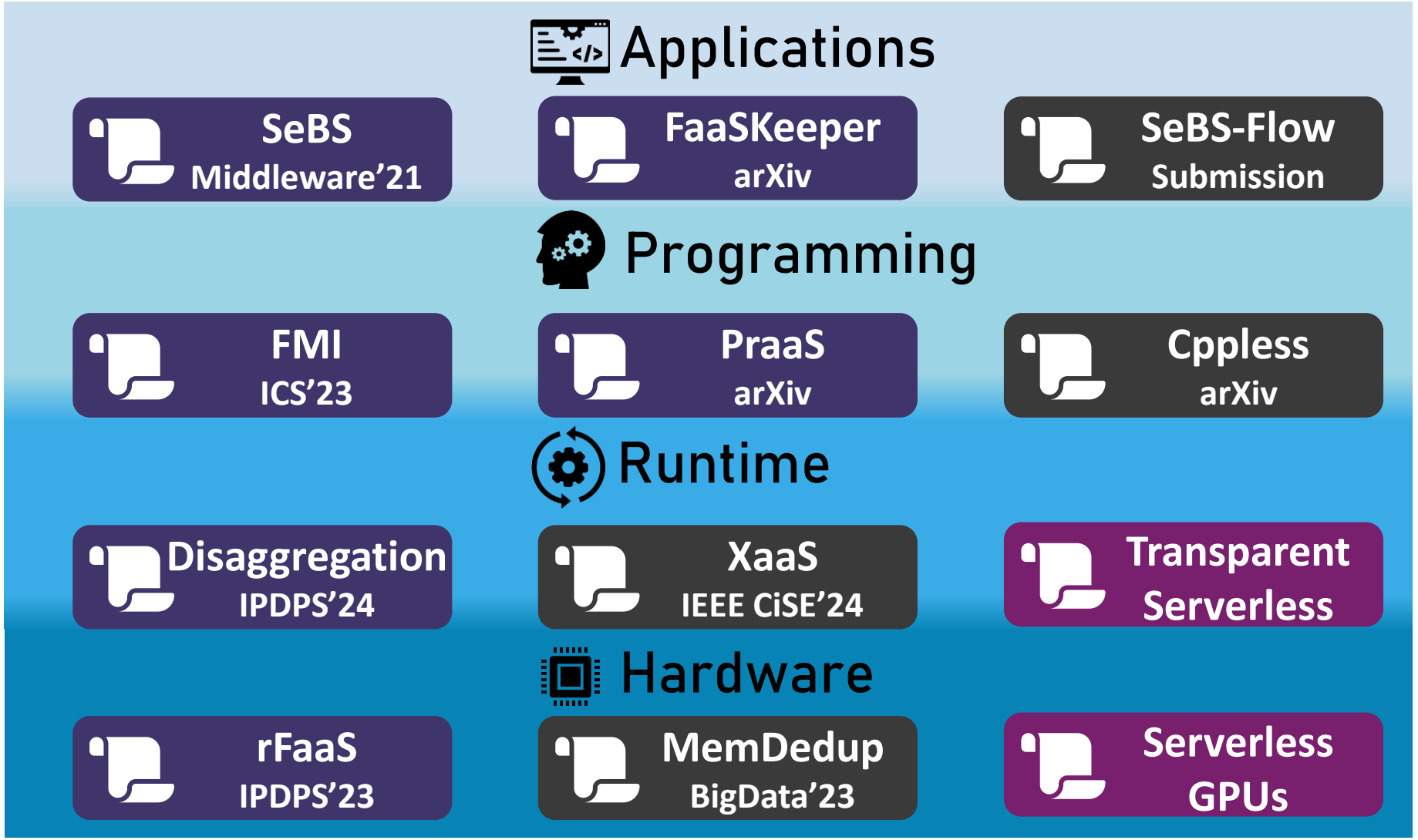
Fast invocations with RDMA acceleration.

How can serverless improve HPC?



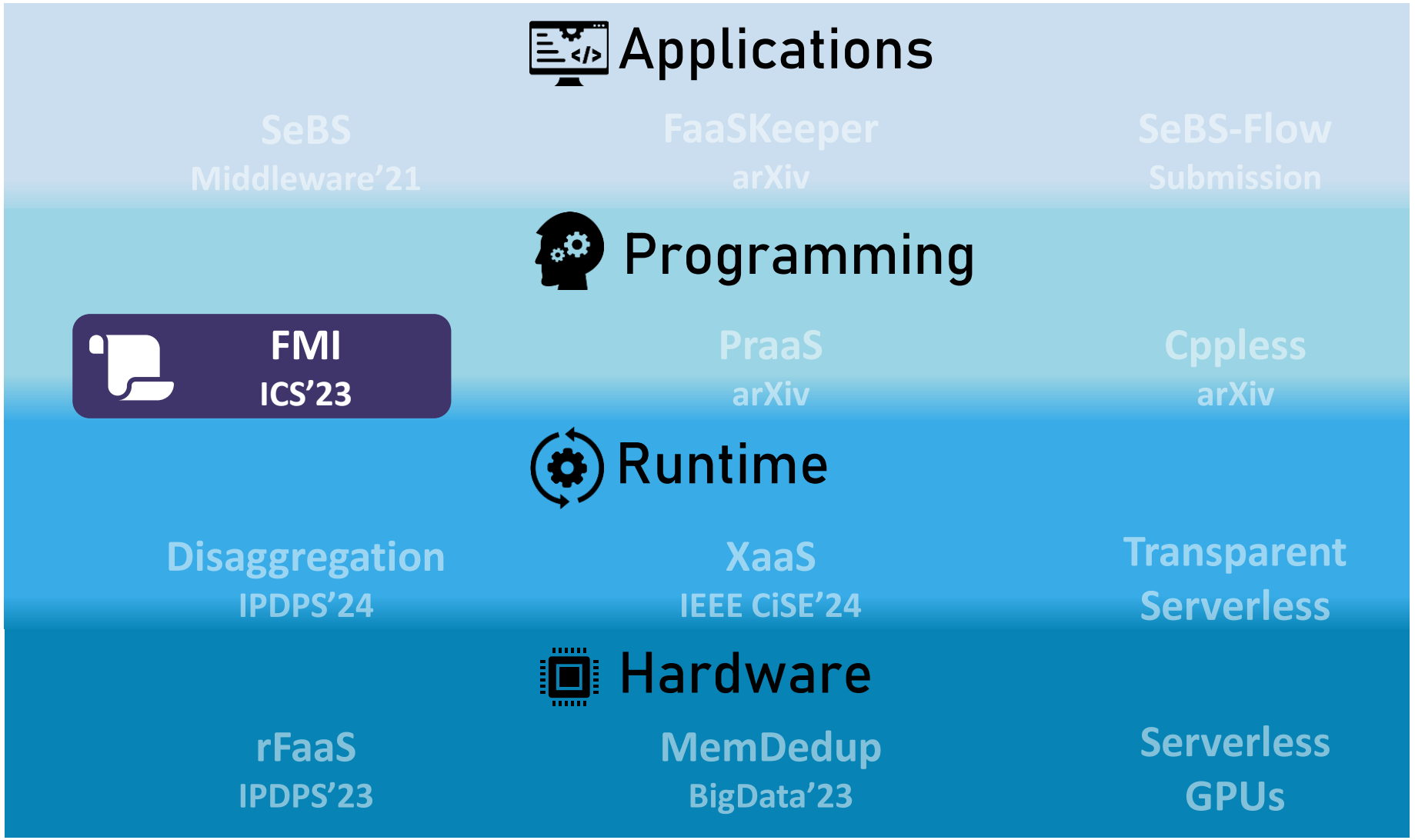
High-Performance Serverless Stack

- Multi-platform benchmarking suite.
- Fast invocations with RDMA acceleration.
- Improved utilization with software disaggregation.



High-Performance Serverless Stack

- Multi-platform benchmarking suite.
- Fast invocations with RDMA acceleration.
- Improved utilization with software disaggregation.
- Communication is slow and restricted.



Communication in serverless

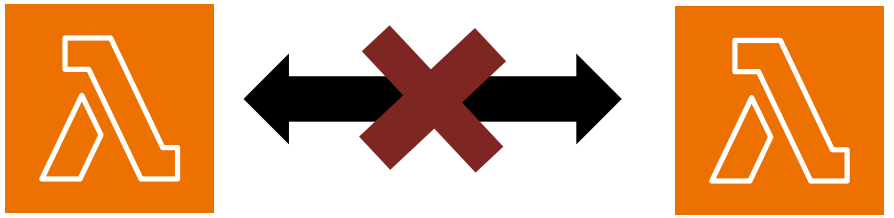
ACM ICS 2023

Communication in serverless

ACM ICS 2023

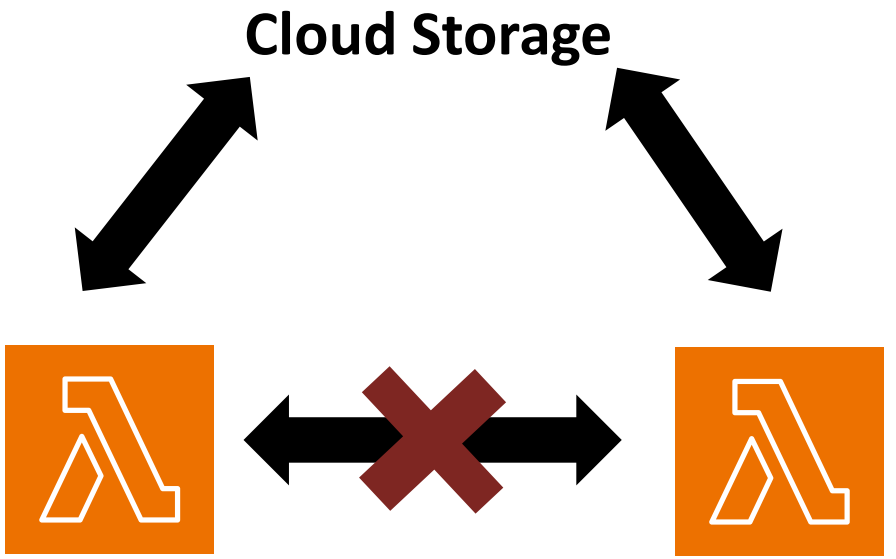
Communication in serverless

ACM ICS 2023



Communication in serverless

ACM ICS 2023



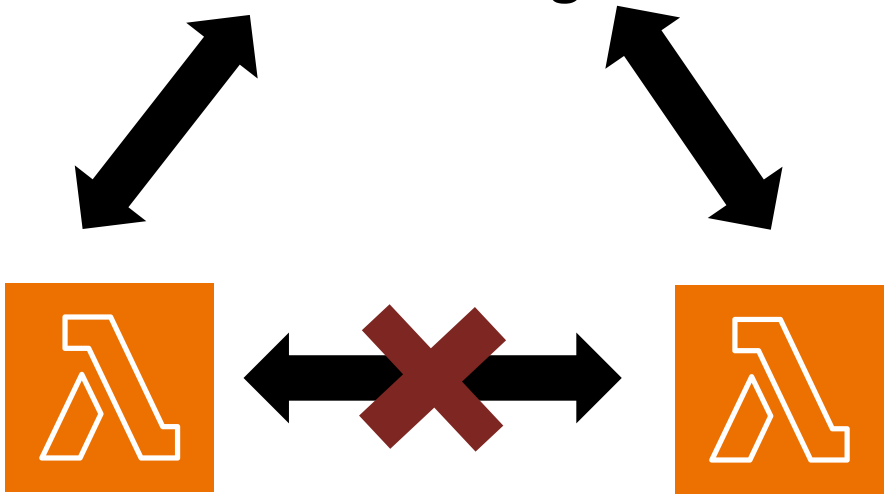
Communication in serverless

ACM ICS 2023

High Latency
For Small Messages

S3

Cloud Storage



Communication in serverless

ACM ICS 2023

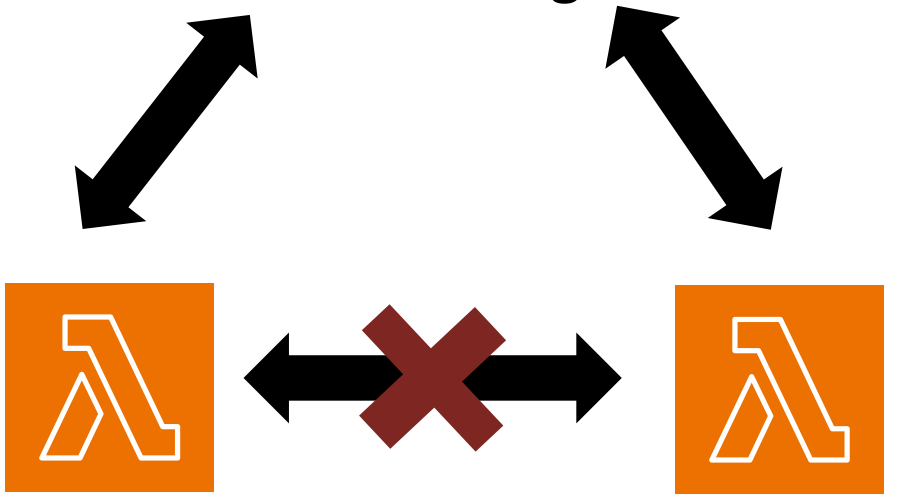
High Latency
For Small Messages

S3

Expensive for
Large Messages

DynamoDB

Cloud Storage



Communication in serverless

ACM ICS 2023

High Latency
For Small Messages

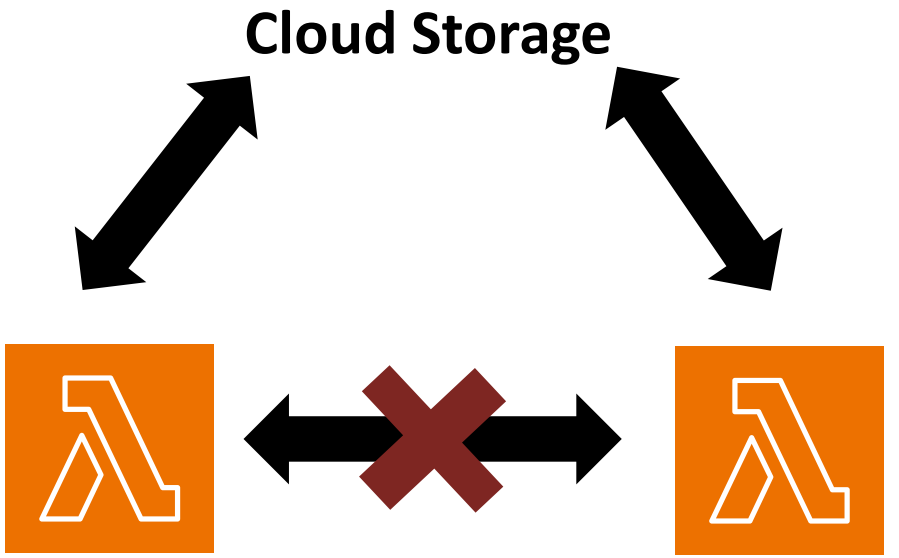
S3

Expensive for
Large Messages

DynamoDB

Not Serverless

Redis

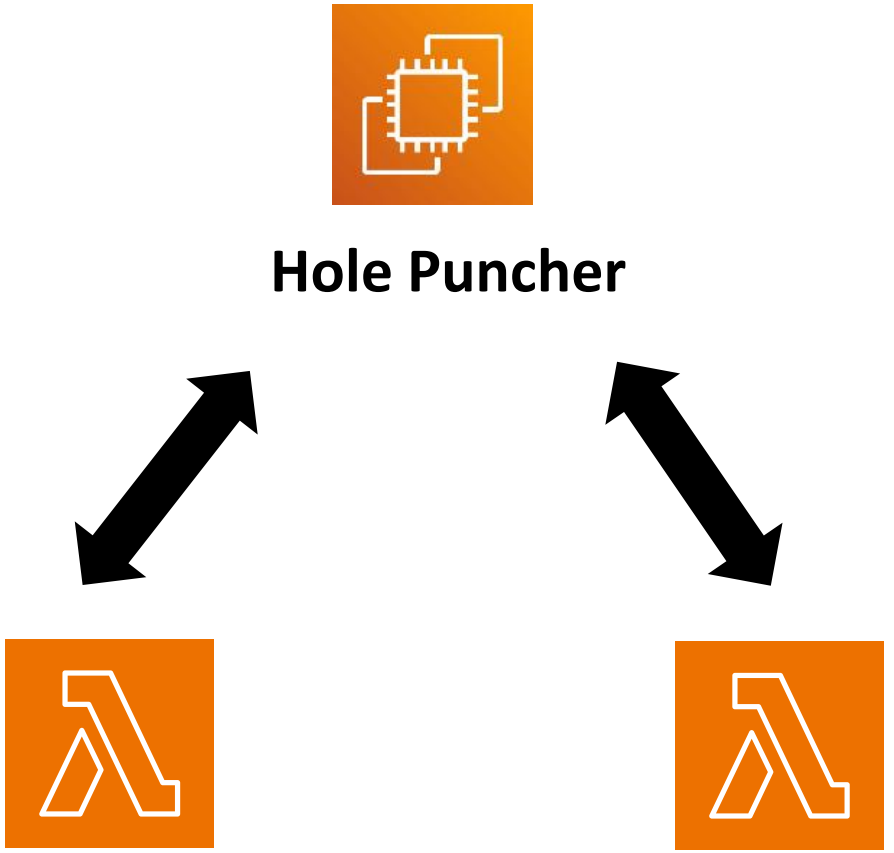


Communication in serverless

ACM ICS 2023

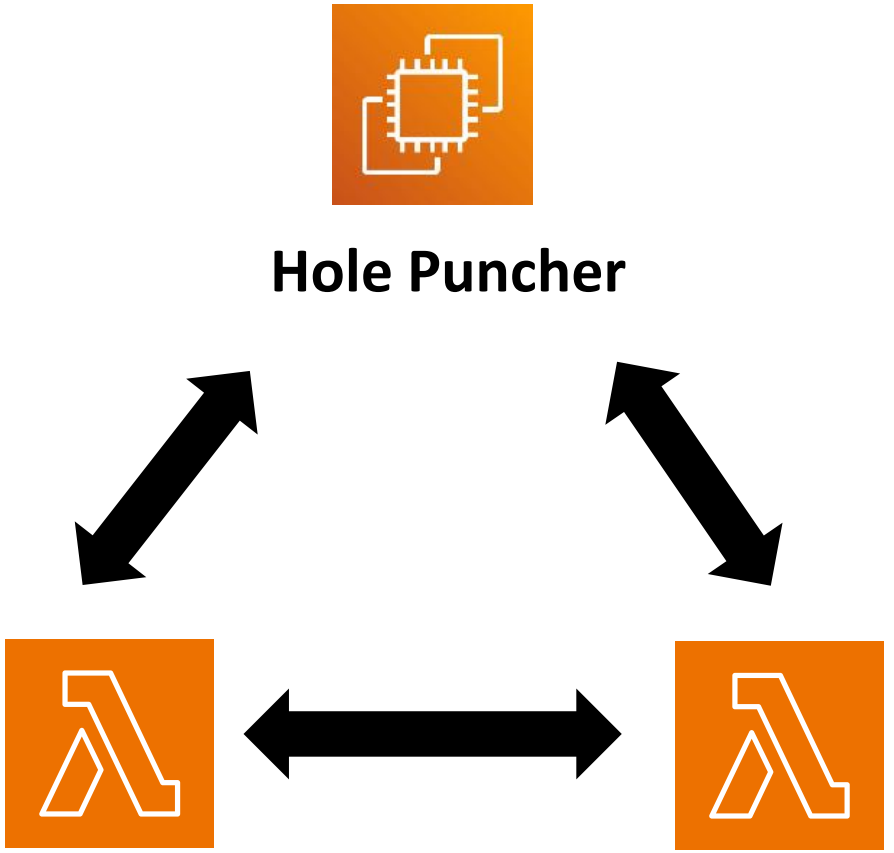
Communication in serverless

ACM ICS 2023



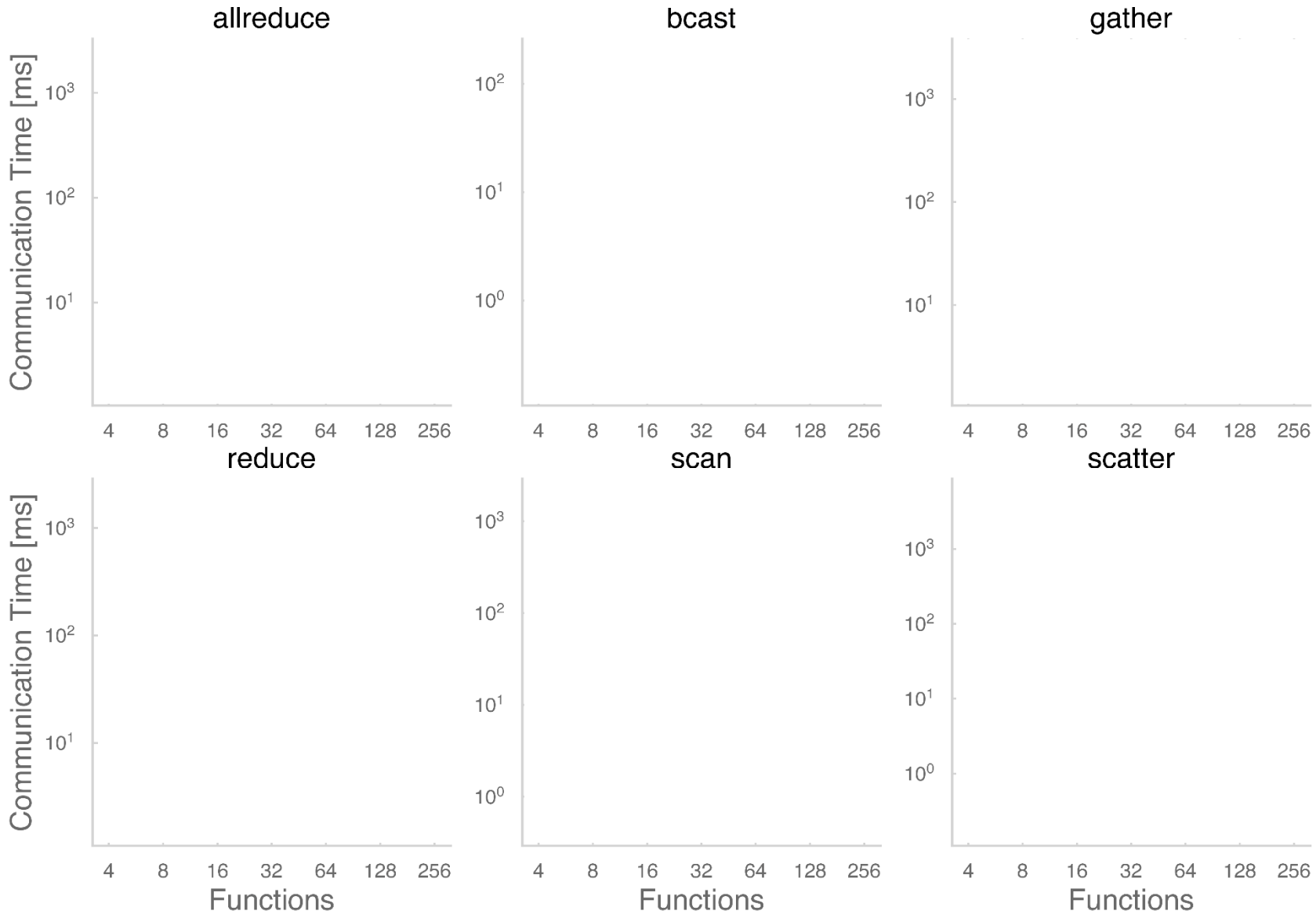
Communication in serverless

ACM ICS 2023



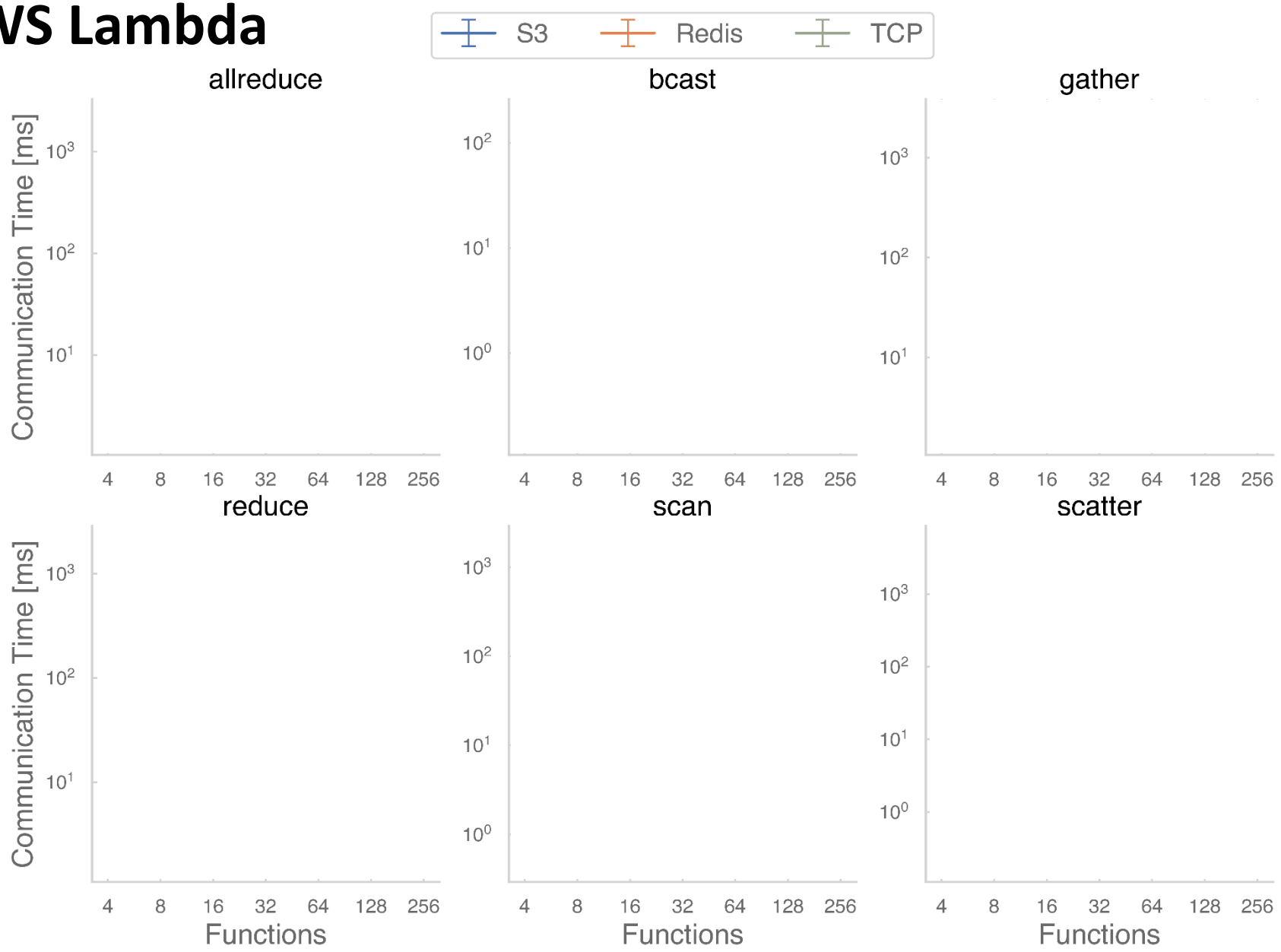
FMI on AWS Lambda

ACM ICS 2023



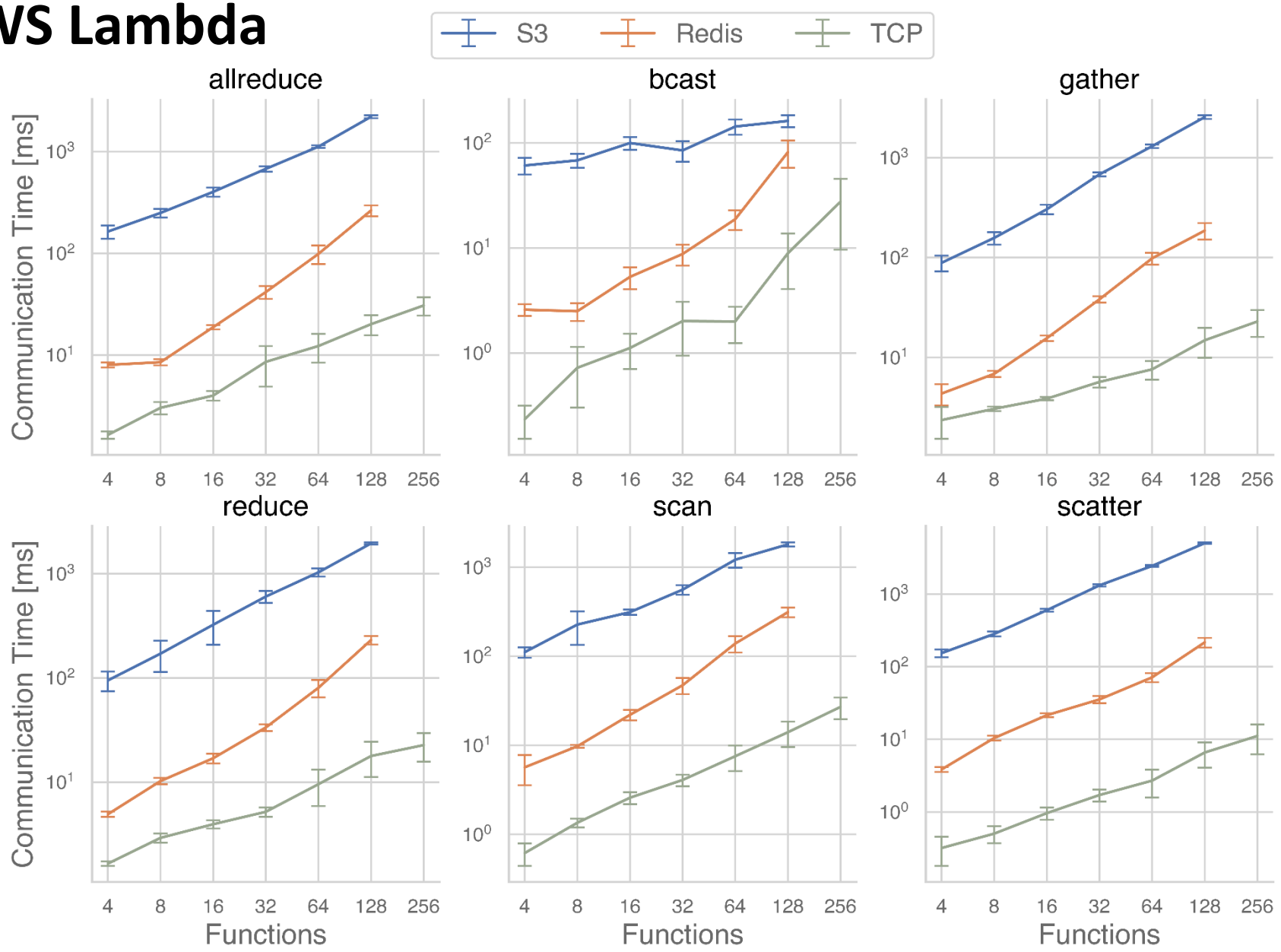
FMI on AWS Lambda

ACM ICS 2023



FMI on AWS Lambda

ACM ICS 2023



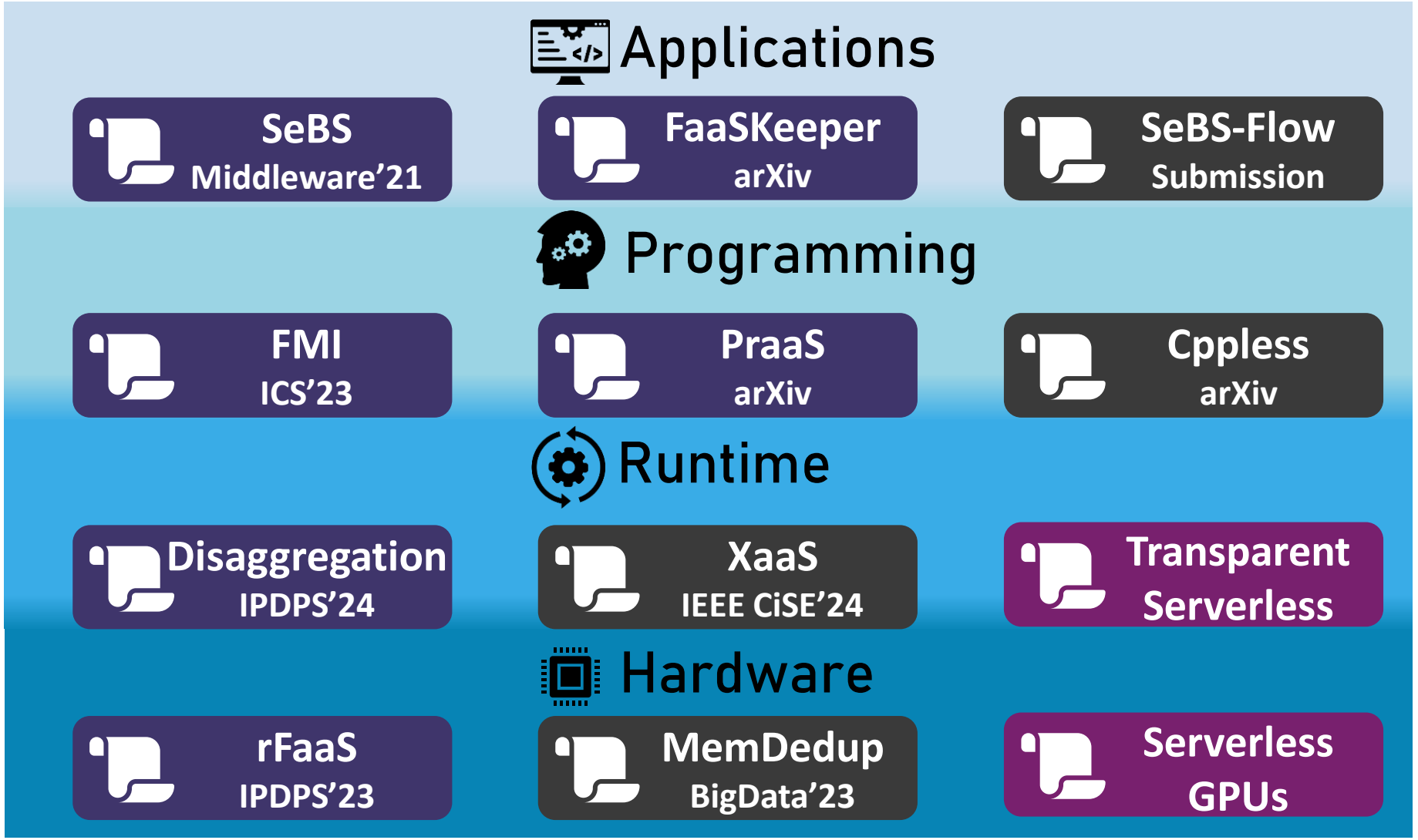
High-Performance Serverless Stack

Multi-platform benchmarking suite.

Fast invocations with RDMA acceleration.

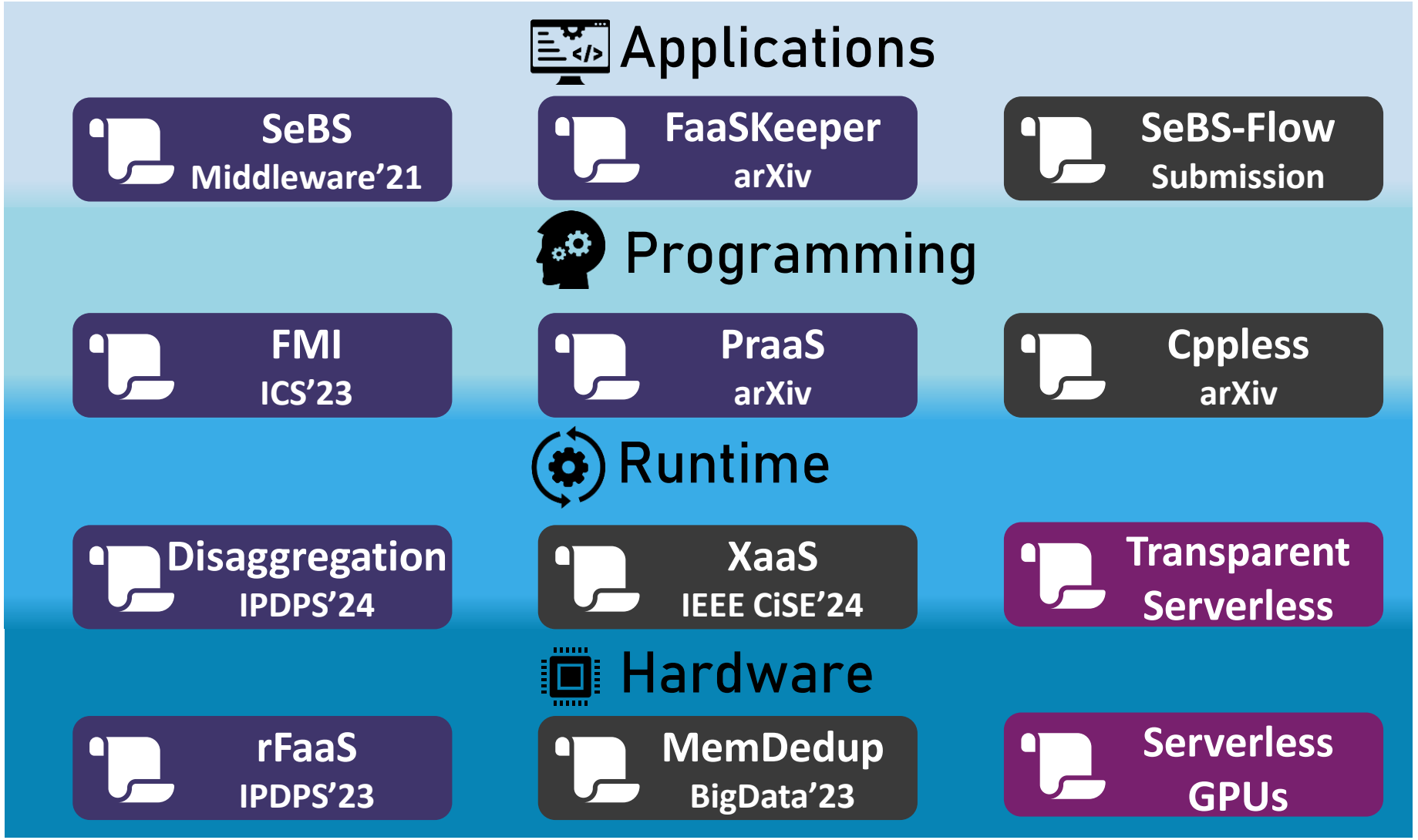
Improved utilization with software disaggregation.

Communication is slow and restricted.



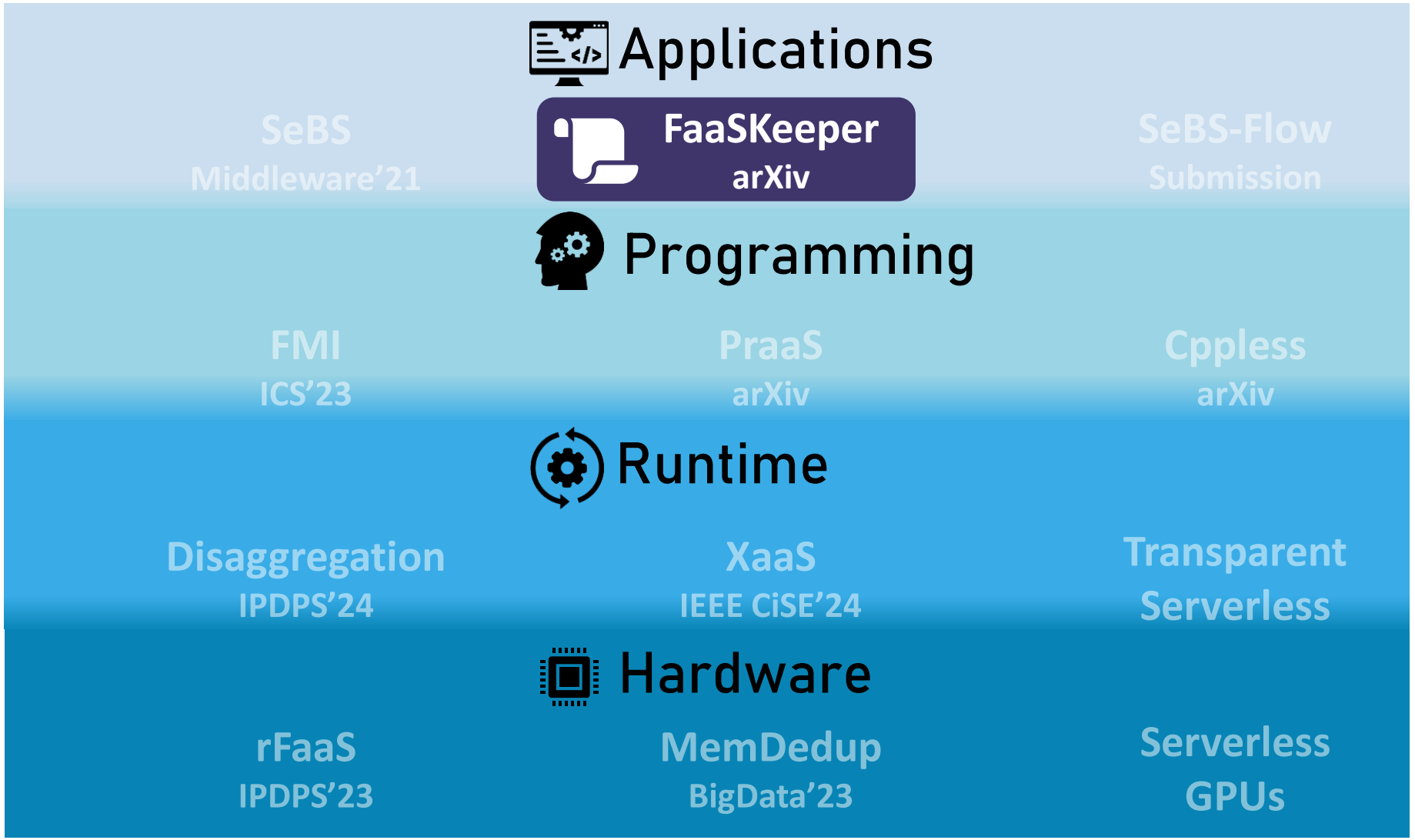
High-Performance Serverless Stack

- Multi-platform benchmarking suite.
- Fast invocations with RDMA acceleration.
- Improved utilization with software disaggregation.
- Fast communication with hole punching.



High-Performance Serverless Stack

- Multi-platform benchmarking suite.
- Fast invocations with RDMA acceleration.
- Improved utilization with software disaggregation.
- Fast communication with hole punching.
- How to port existing and complex systems?



From ZooKeeper to FaaSKeeper

From ZooKeeper to FaaSKeeper

APACHE ZooKeeper™

APACHE HBASE

APACHE Spark™

Apache BookKeeper™

From ZooKeeper to FaaSKeeper

APACHE ZooKeeper™

Infrequently used

From ZooKeeper to FaaSKeeper

APACHE ZooKeeper™

Infrequently used

High read-to-write ratio

From ZooKeeper to FaaSKeeper

APACHE ZooKeeper™

Infrequently used

High read-to-write ratio

Server-centric design

From ZooKeeper to FaaSKeeper

APACHE ZooKeeper™

Infrequently used

High read-to-write ratio

Server-centric design

Complex data model

From ZooKeeper to FaaSKeeper

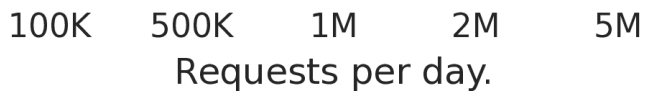
Cost ratio of ZooKeeper and FaaSKeeper, 90% reads.

Cost ratio of ZooKeeper and FaaSKeeper, 80% reads.

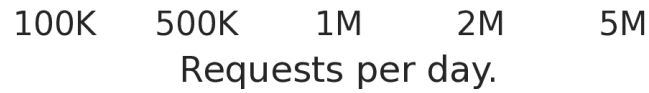
ZooKeeper – constant cost for VMs.
FaaSKeeper – pay per each request.

From ZooKeeper to FaaSKeeper

Cost ratio of ZooKeeper and FaaSKeeper, 90% reads.

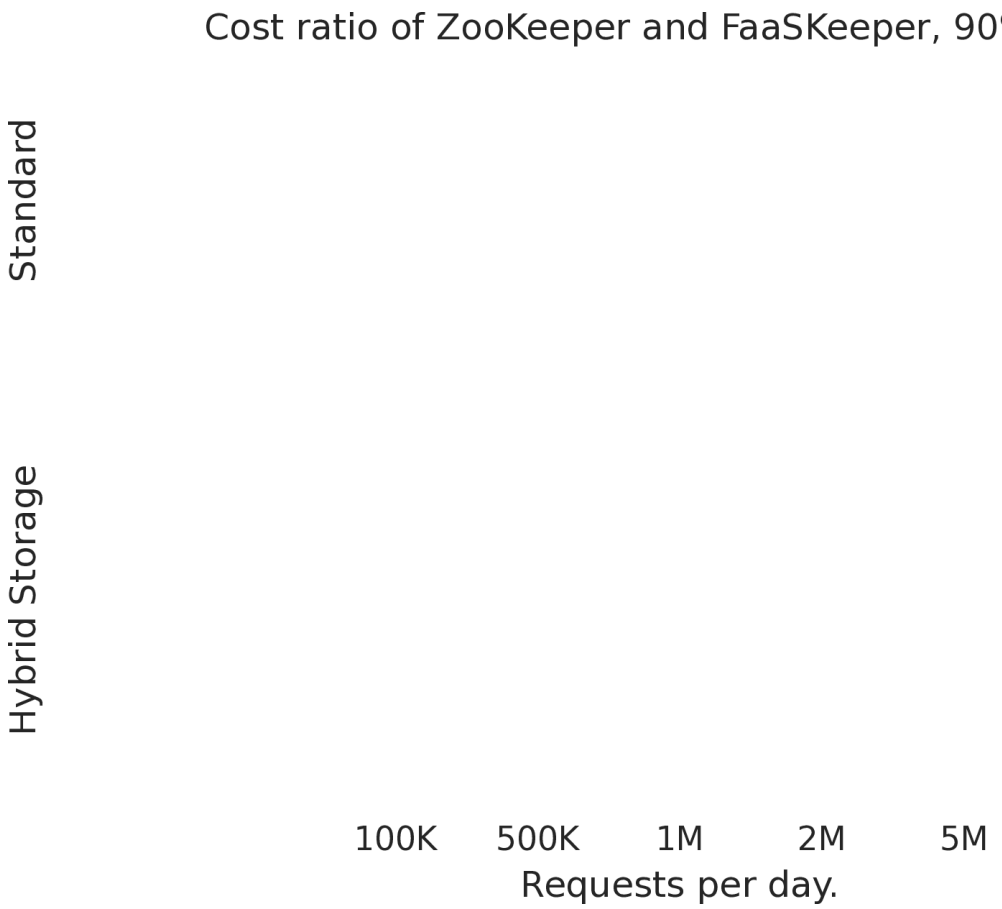


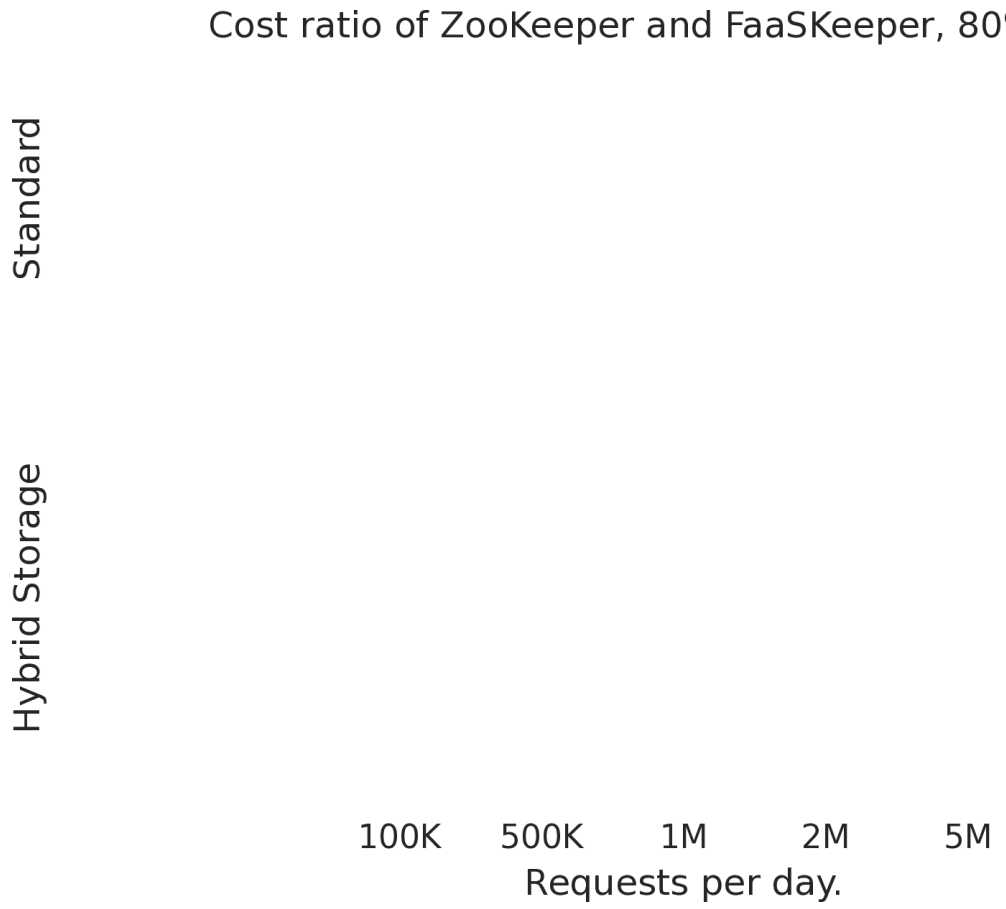
Cost ratio of ZooKeeper and FaaSKeeper, 80% reads.



ZooKeeper – constant cost for VMs.
FaaSKeeper – pay per each request.

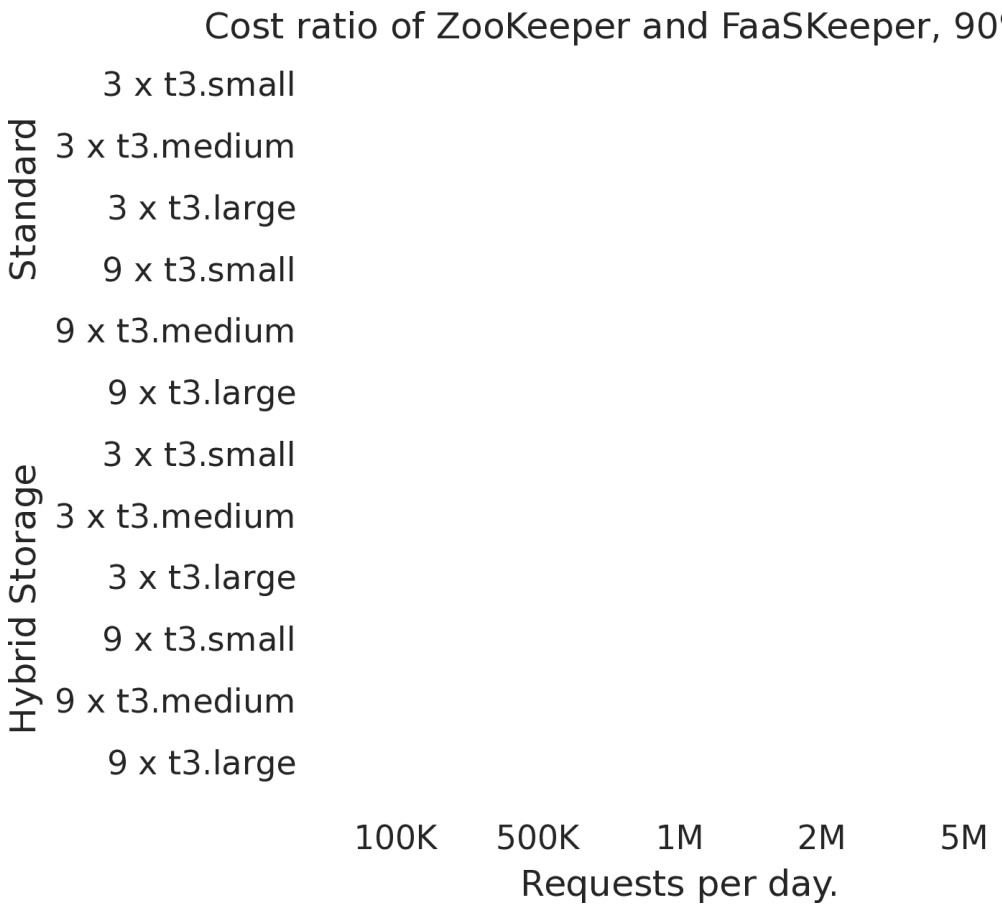
From ZooKeeper to FaaSKeeper

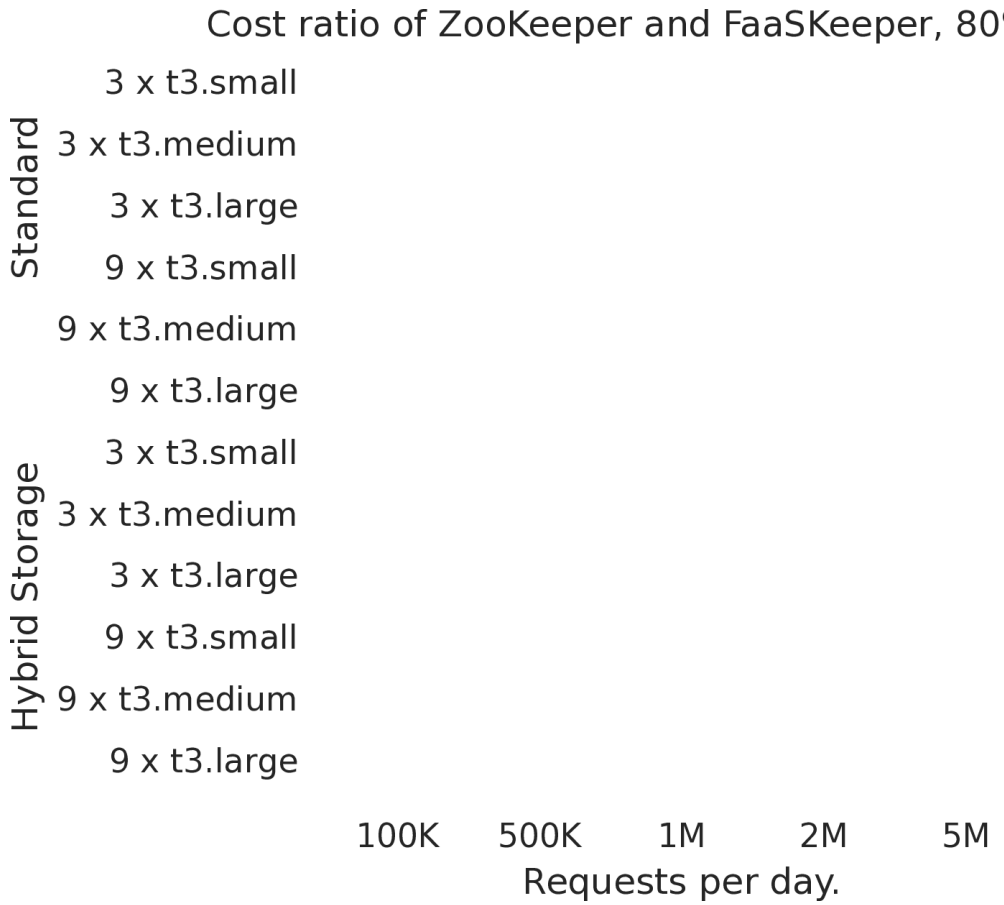




ZooKeeper – constant cost for VMs.
FaaSKeeper – pay per each request.

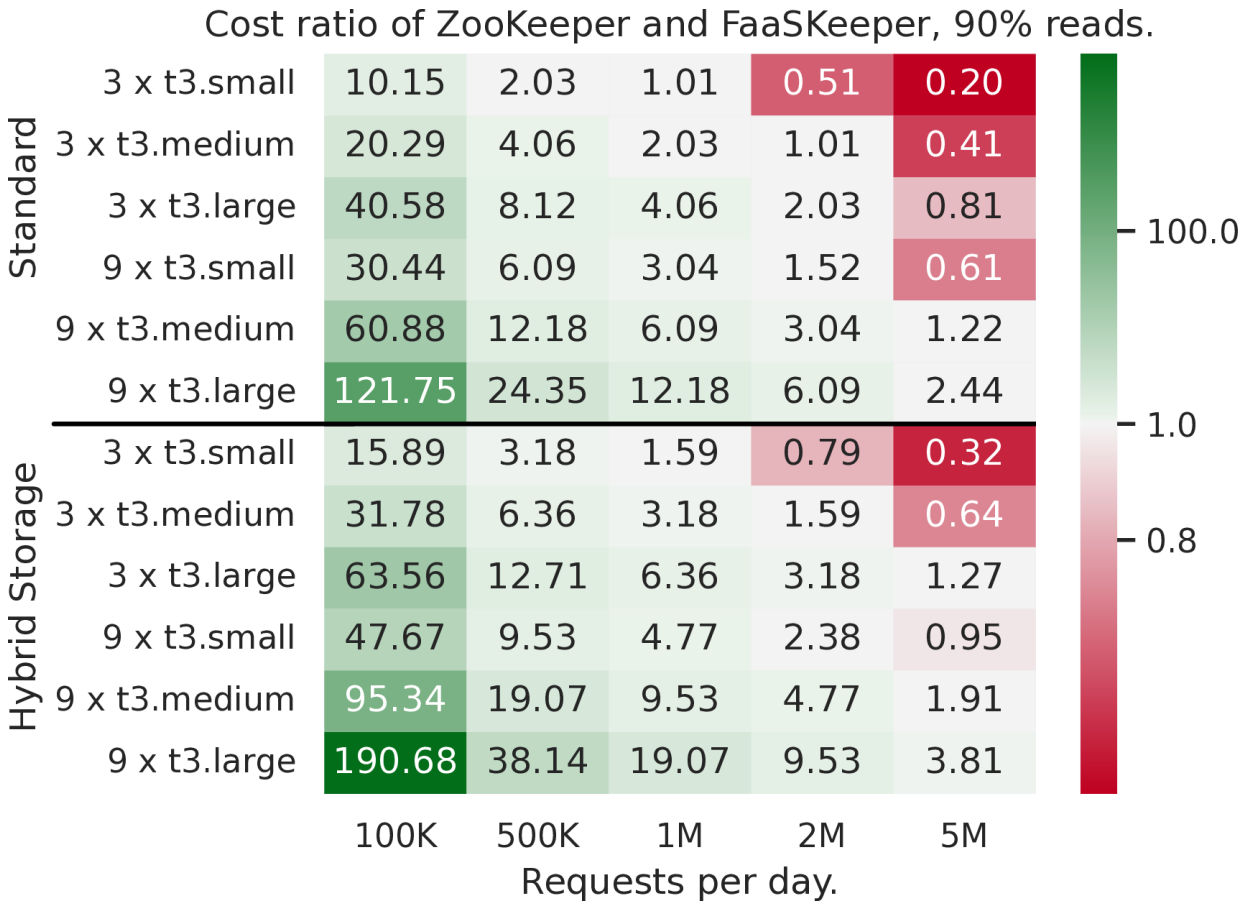
From ZooKeeper to FaaSKeeper

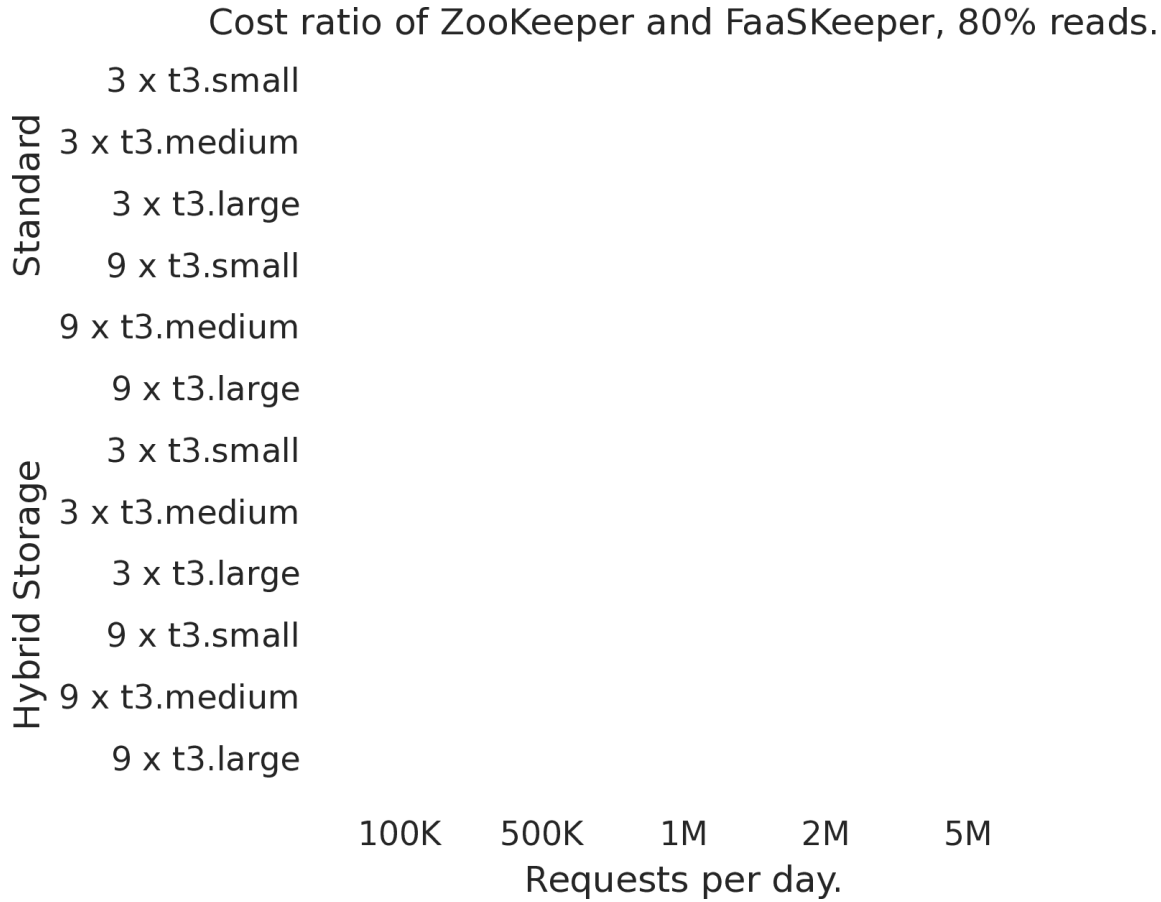




ZooKeeper – constant cost for VMs.
FaaSKeeper – pay per each request.

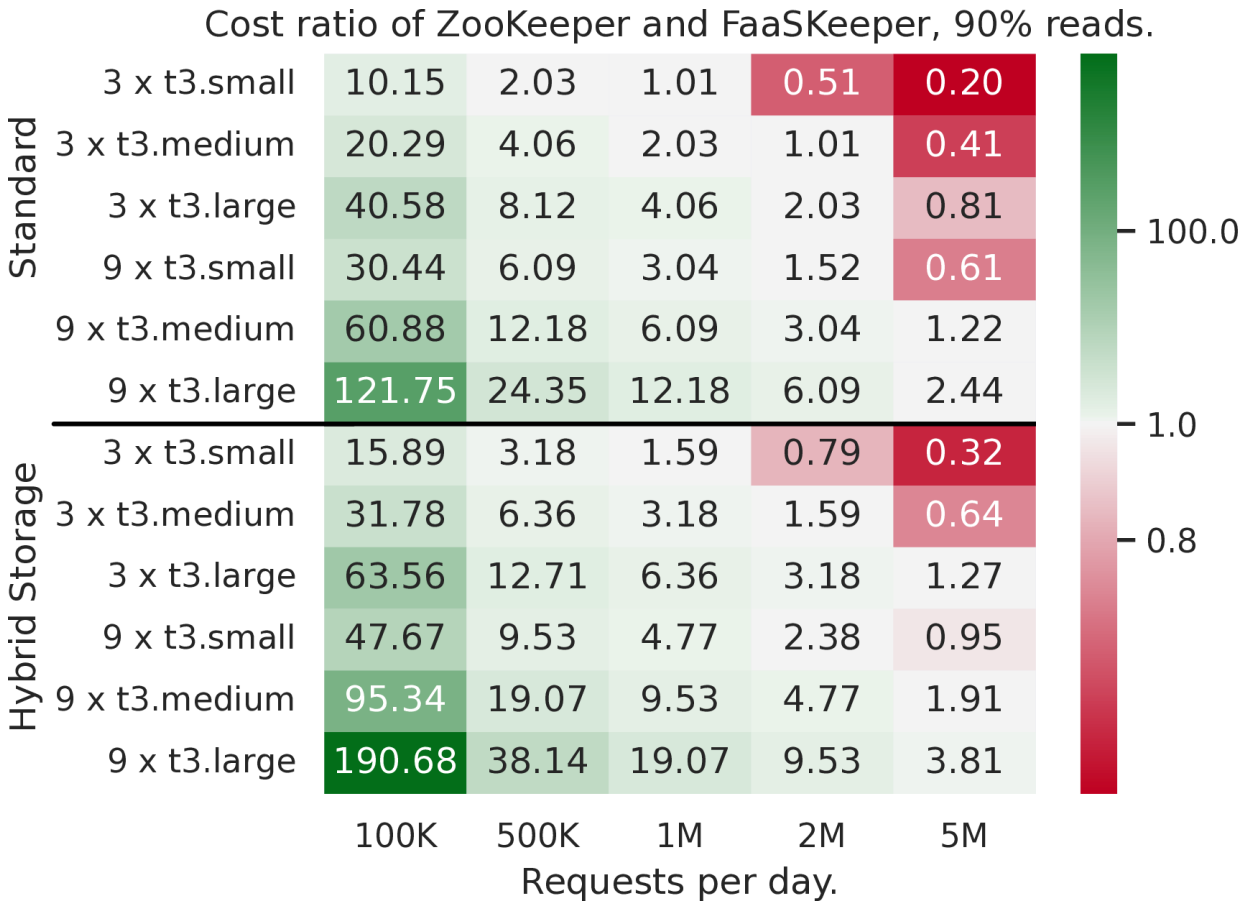
From ZooKeeper to FaaSKeeper

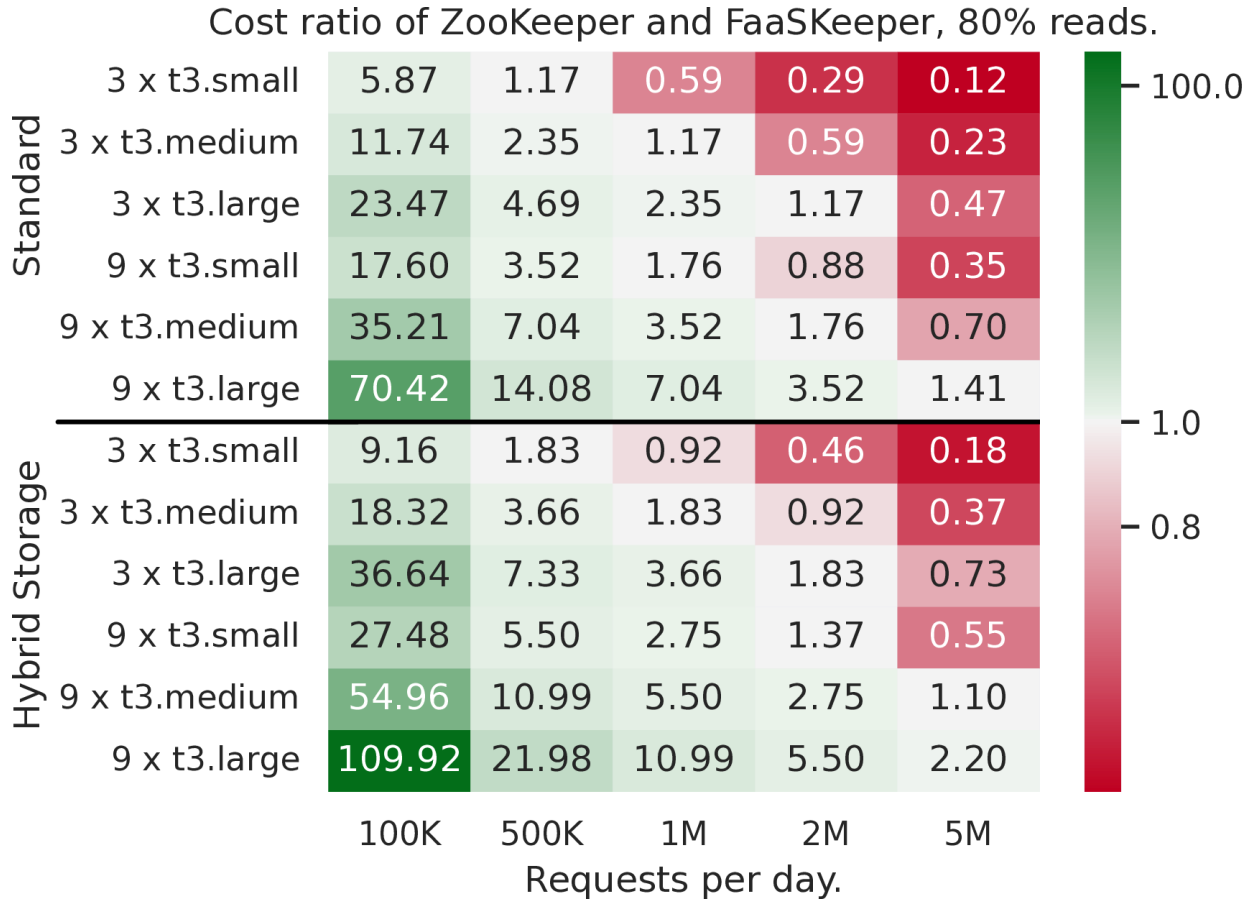




ZooKeeper – constant cost for VMs.
FaaSKeeper – pay per each request.

From ZooKeeper to FaaSKeeper

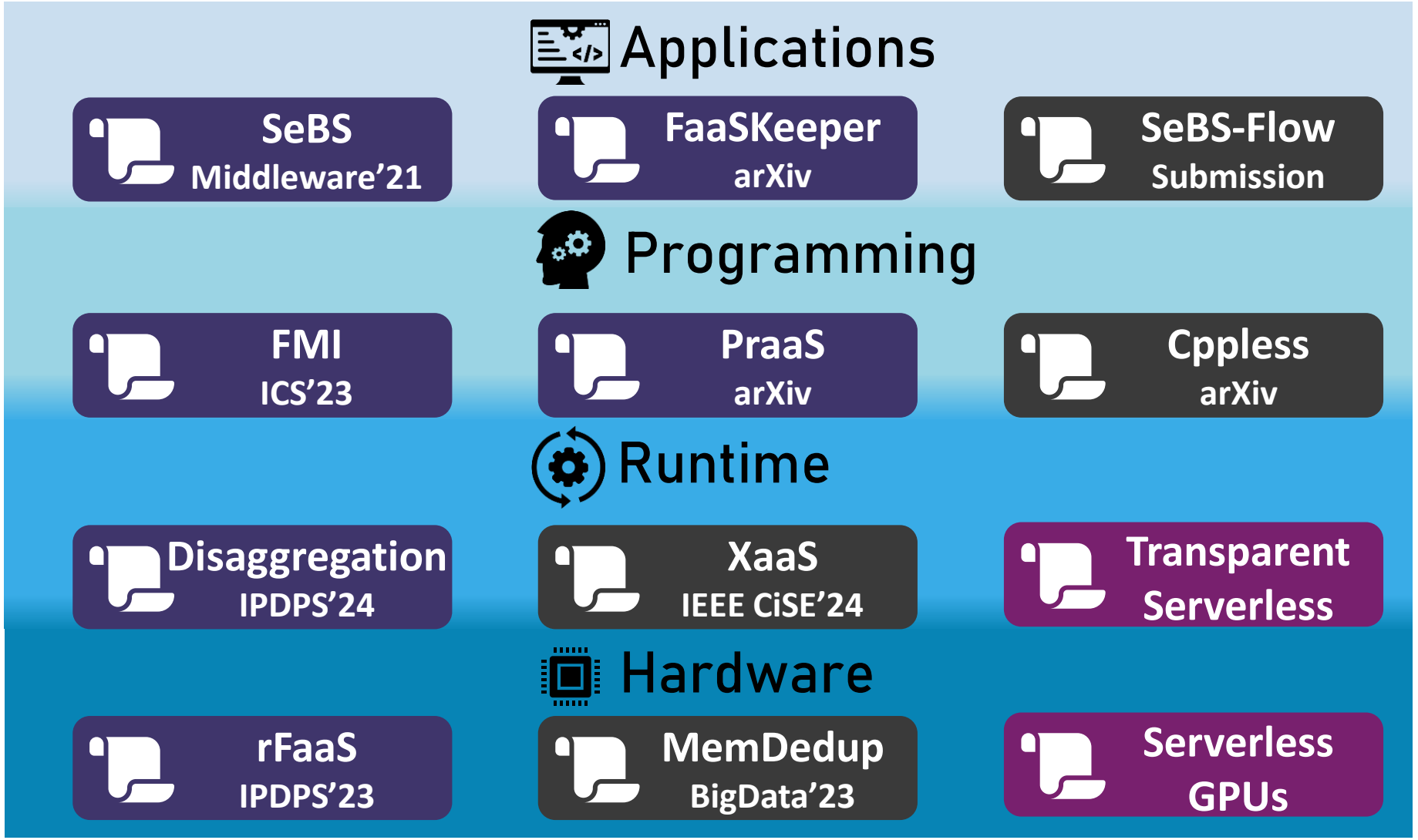




ZooKeeper – constant cost for VMs.
FaaSKeeper – pay per each request.

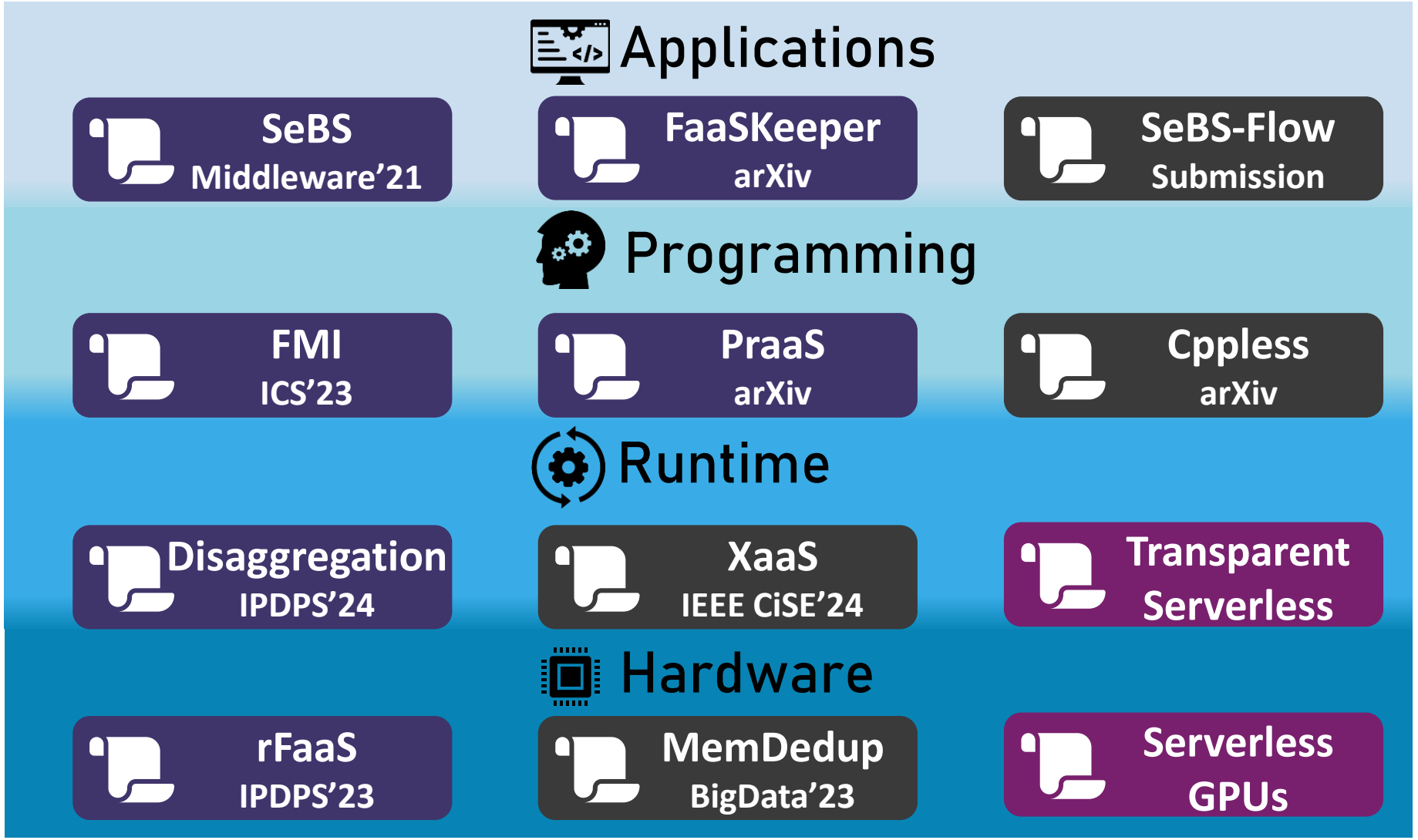
High-Performance Serverless Stack

- Multi-platform benchmarking suite.
- Fast invocations with RDMA acceleration.
- Improved utilization with software disaggregation.
- Fast communication with hole punching.
- How to port existing and complex systems?



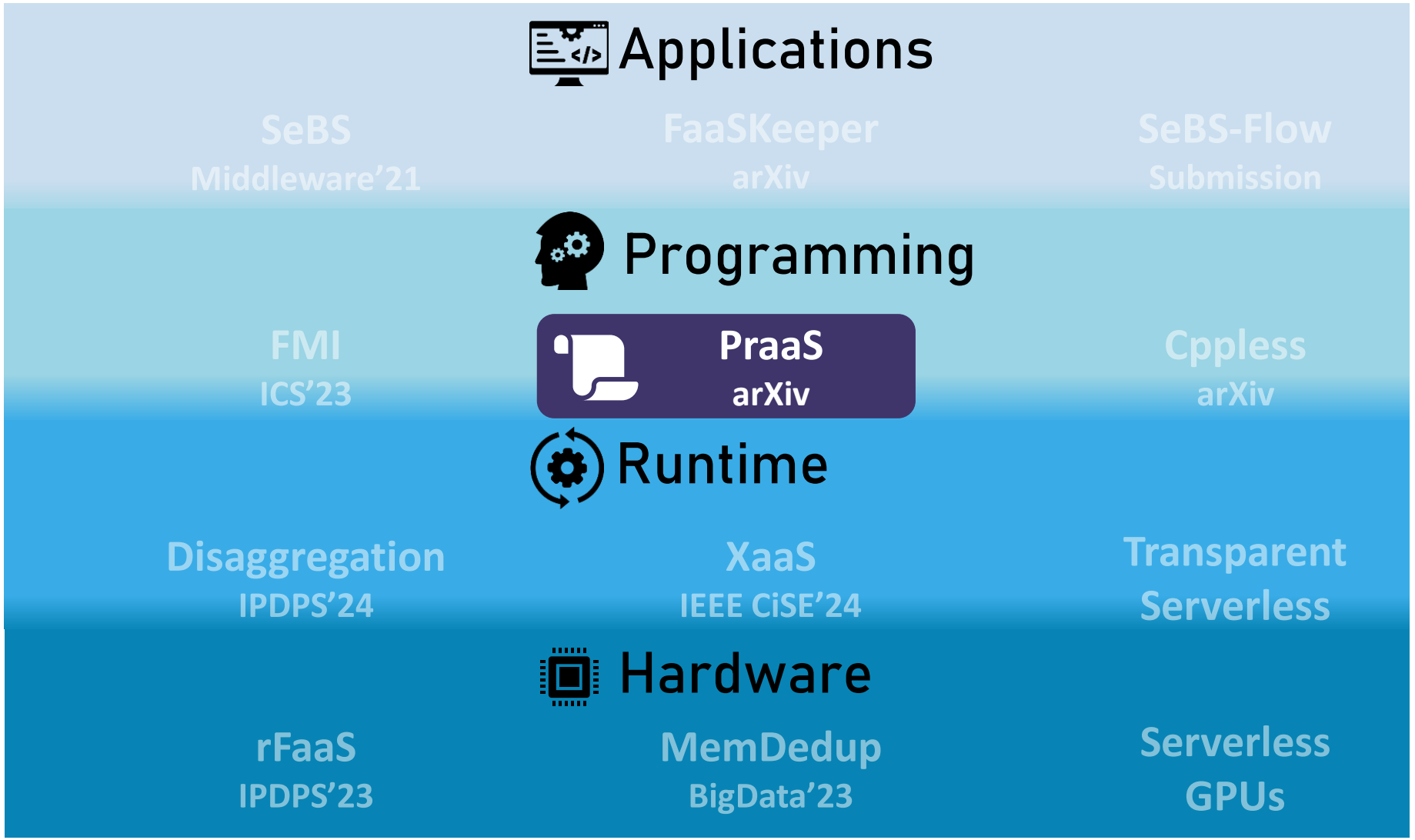
High-Performance Serverless Stack

- Multi-platform benchmarking suite.
- Fast invocations with RDMA acceleration.
- Improved utilization with software disaggregation.
- Fast communication with hole punching.
- Blueprint for serverless services.



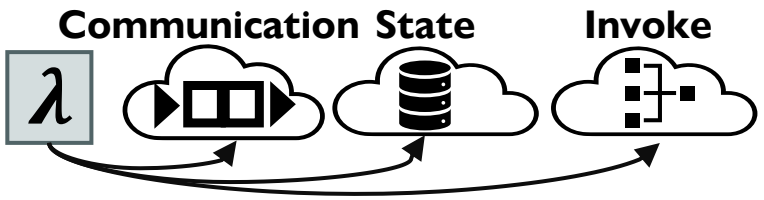
High-Performance Serverless Stack

- Multi-platform benchmarking suite.
- Fast invocations with RDMA acceleration.
- Improved utilization with software disaggregation.
- Fast communication with hole punching.
- Blueprint for serverless services.
- Serverless is hard to program.



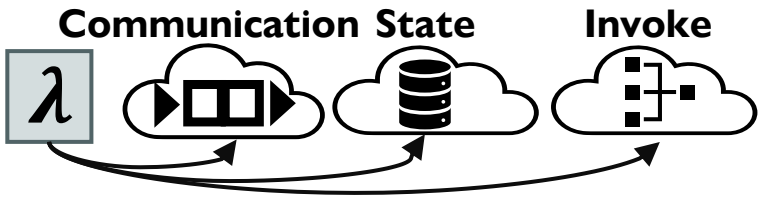
Serverless Process

Serverless Process

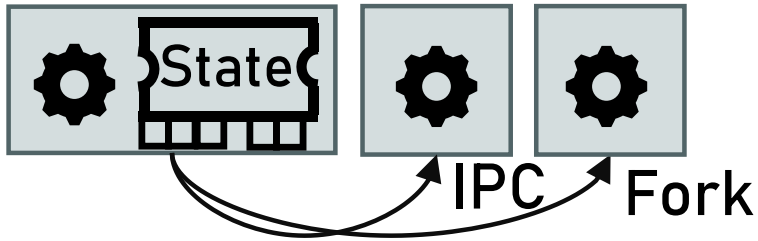


Serverless Function
Millisecond latency of cloud proxies.

Serverless Process

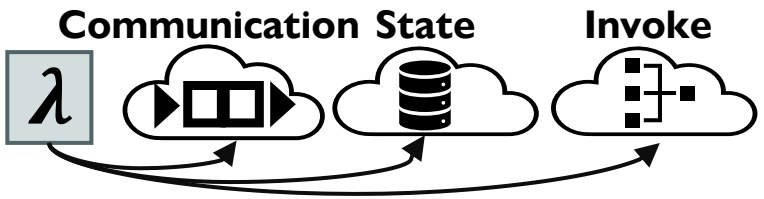


Serverless Function
Millisecond latency of cloud proxies.

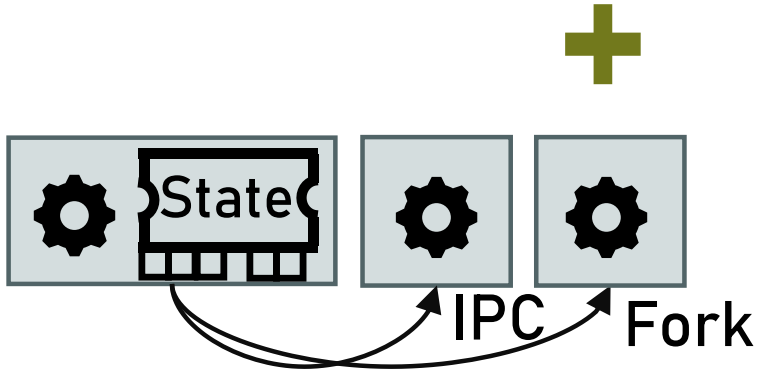


OS Process
Nano- and micro-second latency of OS primitives.

Serverless Process

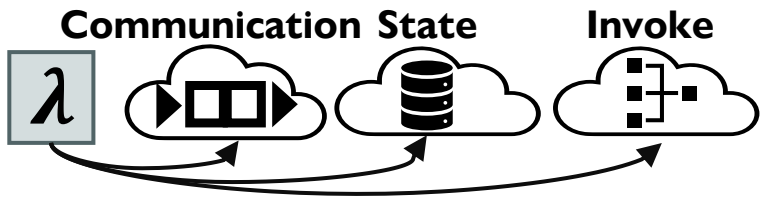


Serverless Function
Millisecond latency of cloud proxies.

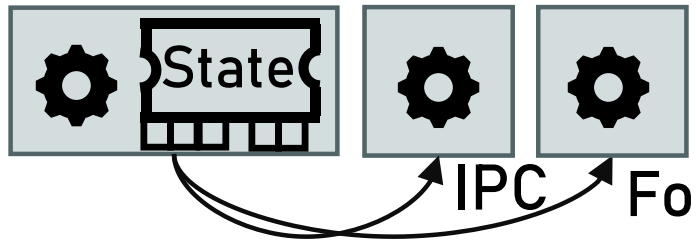


OS Process
Nano- and micro-second latency of OS primitives.

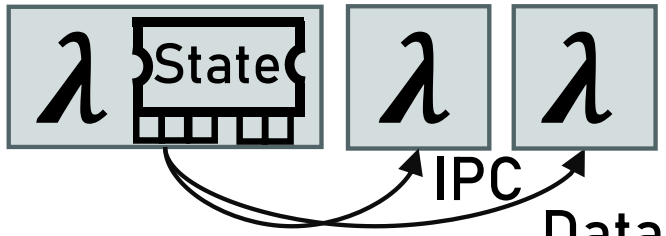
Serverless Process



Serverless Function
Millisecond latency of cloud proxies.



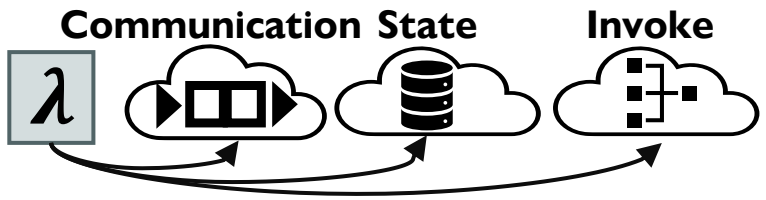
OS Process
Nano- and micro-second latency of OS primitives.



Serverless Process
Microsecond latency of PraaS backend.

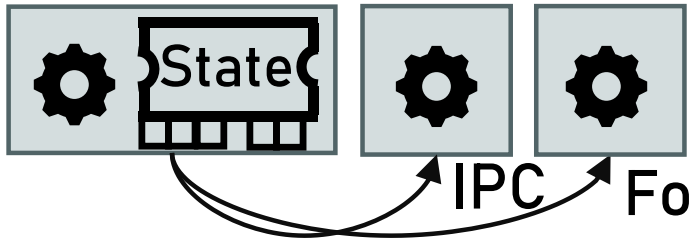
Data Plane

Serverless Process



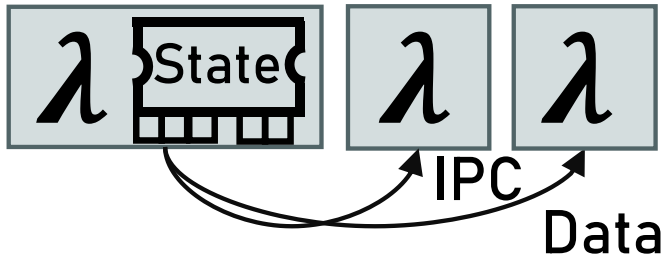
Serverless Function
Millisecond latency of cloud proxies.

+



OS Process
Nano- and micro-second latency of OS primitives.

=

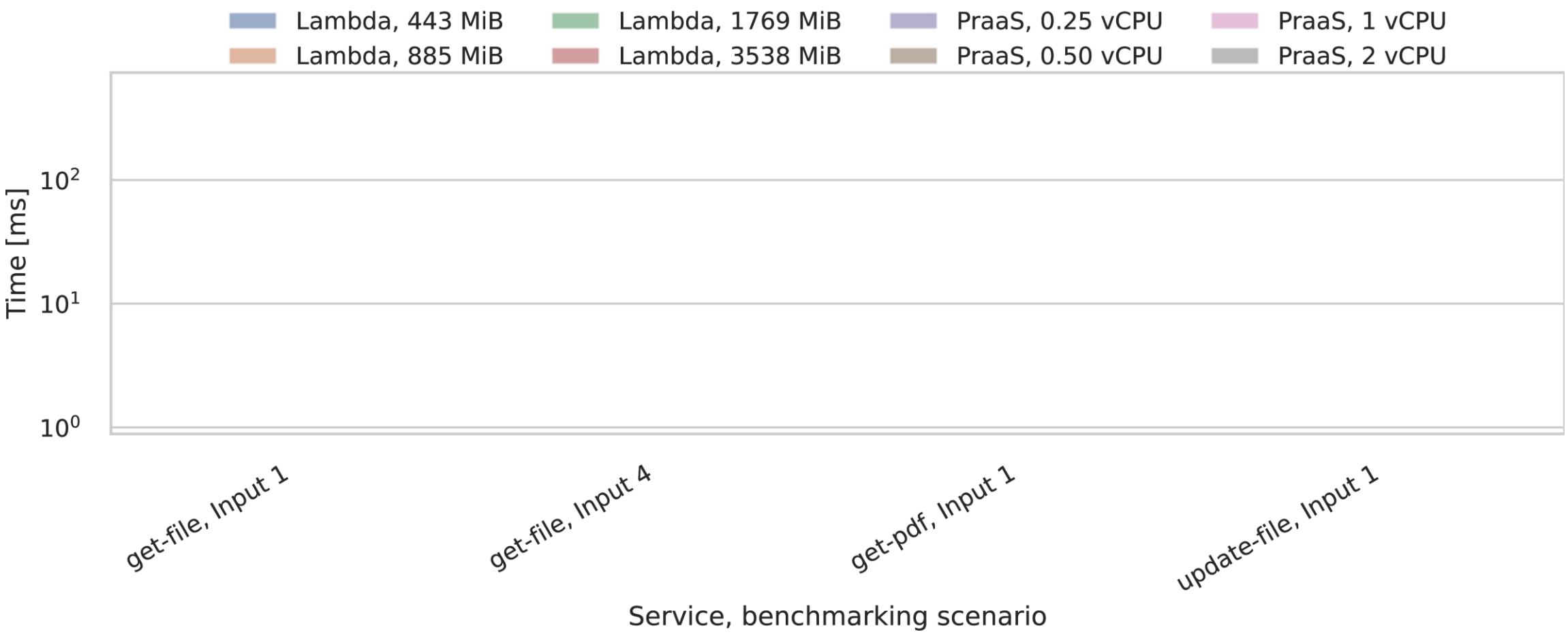


Serverless Process
Microsecond latency of PraaS backend.

Works on AWS Fargate, Knative, Kubernetes.

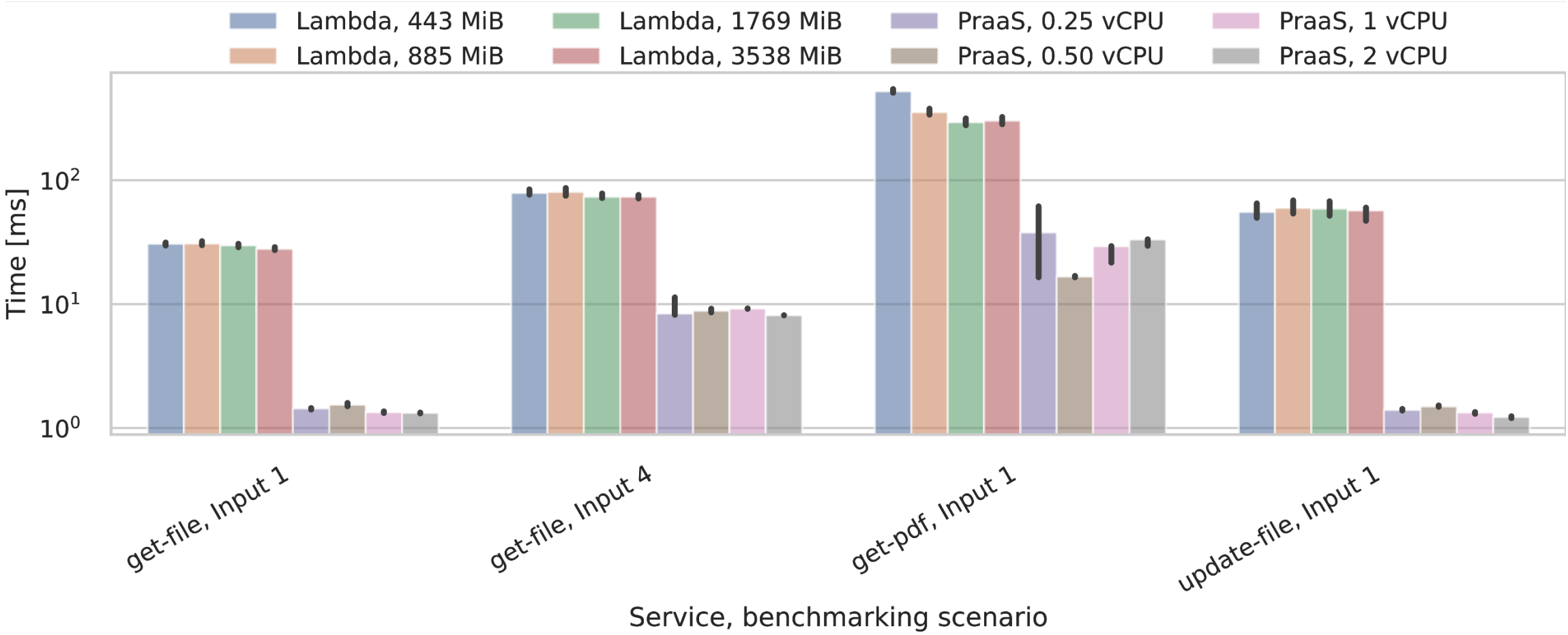
Benchmark: LaTeX Microservice

Benchmark: LaTeX Microservice



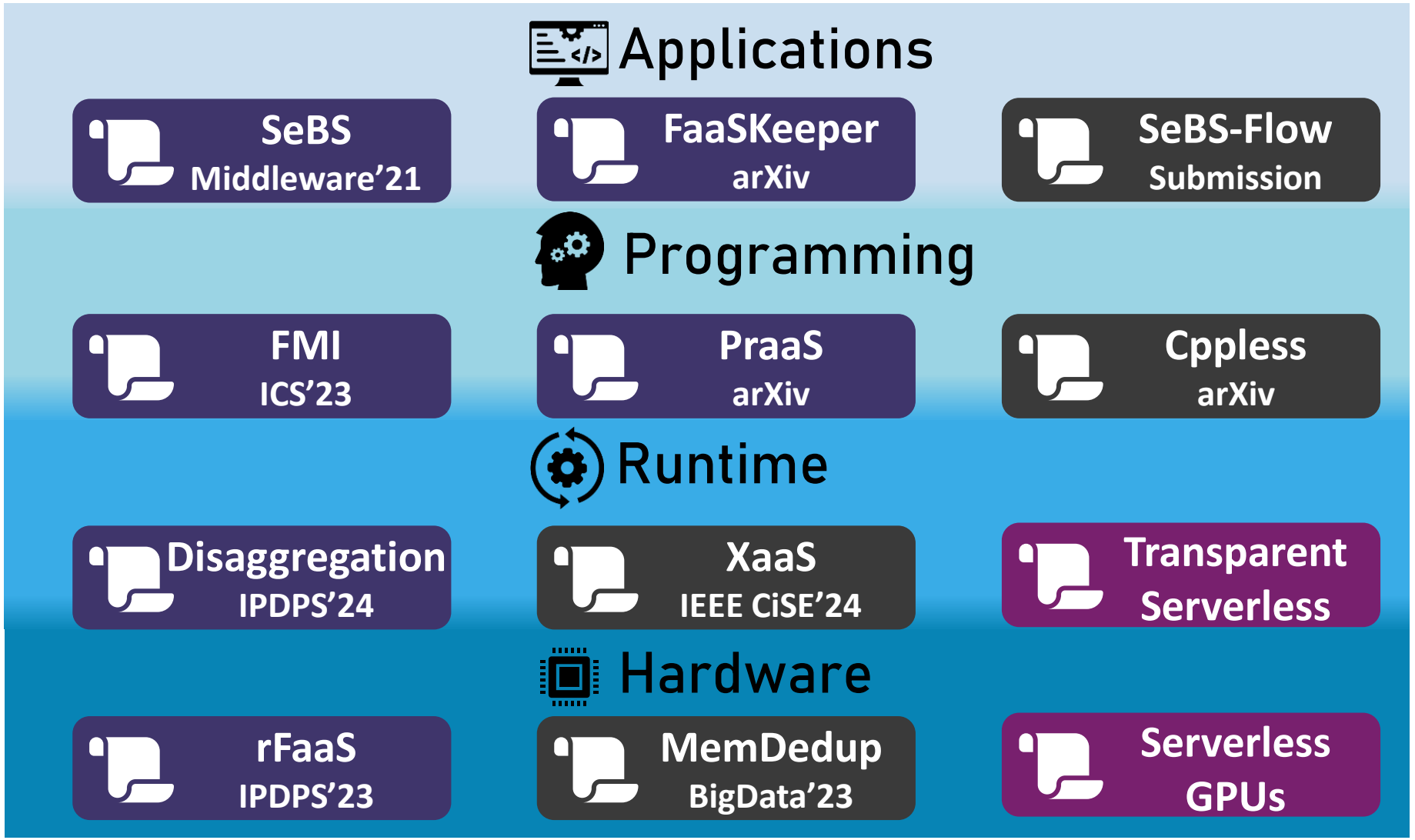
Benchmark: LaTeX Microservice

Benchmark: LaTeX Microservice



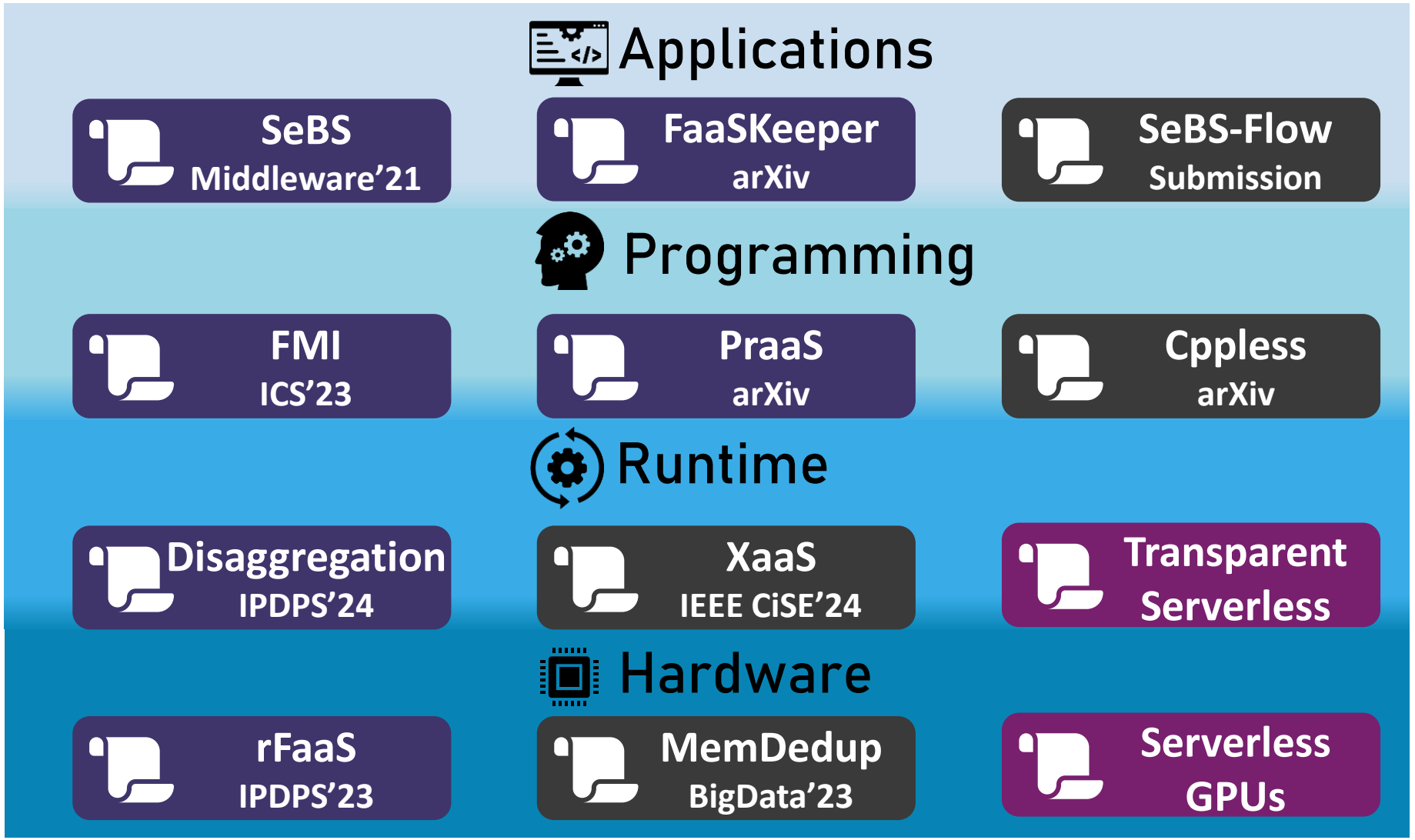
High-Performance Serverless Stack

- Multi-platform benchmarking suite.
- Fast invocations with RDMA acceleration.
- Improved utilization with software disaggregation.
- Fast communication with hole punching.
- Blueprint for serverless services.
- Serverless is hard to program.



High-Performance Serverless Stack

- Multi-platform benchmarking suite.
- Fast invocations with RDMA acceleration.
- Improved utilization with software disaggregation.
- Fast communication with hole punching.
- Blueprint for serverless services.
- Enhanced programming model with processes.



Availability and Acknowledgments

Availability and Acknowledgments

spcl/serverless-benchmarks

spcl/rFaaS

spcl/PraaS

spcl/fmi

spcl/FaaSKeeper

Availability and Acknowledgments

spcl/serverless-benchmarks

spcl/rFaaS

spcl/PraaS

spcl/fmi

spcl/FaaSKeeper

FNSNF Fonds national suisse Schweizerischer Nationalfonds Fondo nazionale svizzero Swiss National Science Foundation

erc

CSCS Centro Svizzero di Calcolo Scientifico Swiss National Supercomputing Centre

aws

Google Cloud

Google Summer of Code

Marcin Copik, Torsten Hoefler, Alexandru Calotoiu, Maciej Besta, Roman Böhringer, Rodrigo Bruno, Marcin Chrapek, Tobias Grosser, Grzegorz Kwaśniewski, Michał Podstawski, Wei Qiu, Gyorgy Rethy, Larissa Schmid, Konstantin Taranov, Nicolas Wicki, Felix Wolf, and Pengyu Zhou

High Performance Serverless for HPC and Clouds