Funky: Cloud-native FPGA Virtualization and Orchestration

Atsushi Koshiba, Charalampos Mainas, Pramod Bhatotia

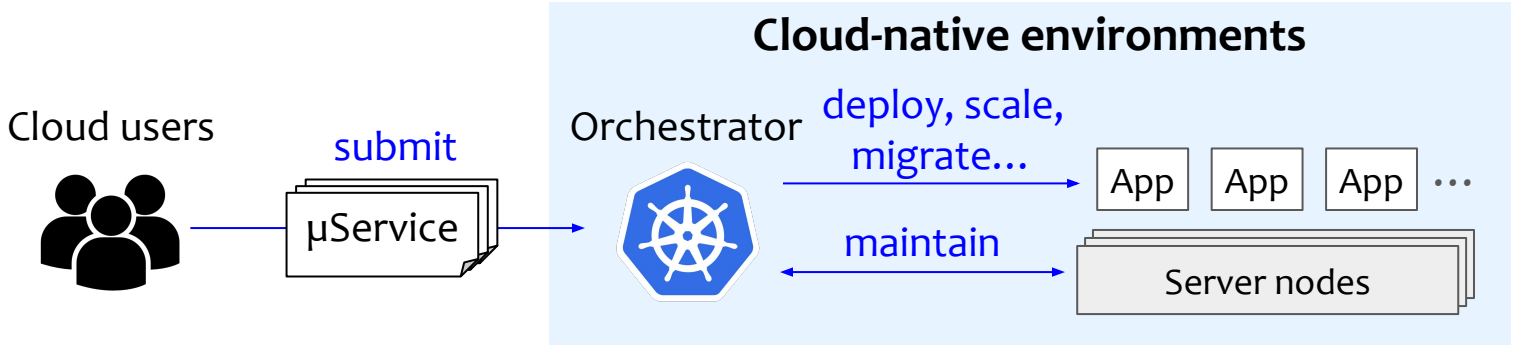
Systems Research Group at TU Munich
https://dse.in.tum.de/

Technische Universität München

ACM SoCC 2025

Cloud-native environments

Cloud-native environments are a de-facto standard in a modern cloud

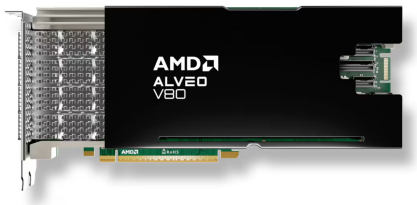


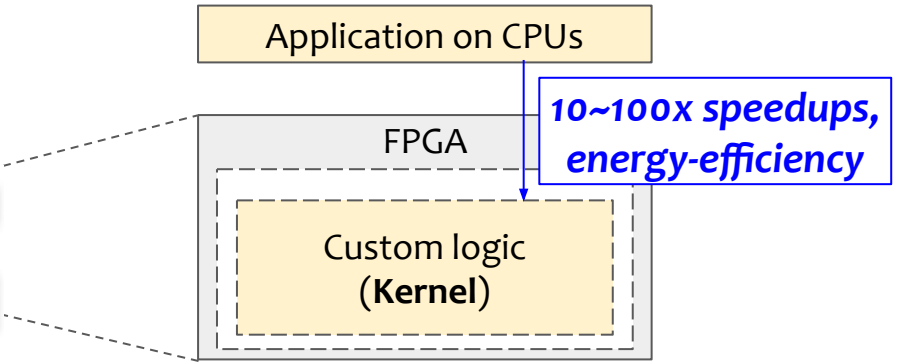
Cloud orchestrators facilitate application deployment and server management

Accelerators (***FPGAs***) for cloud workloads

Field Programmable Gate Arrays (**FPGAs**) promise accelerating cloud workloads

Cloud FPGA cards

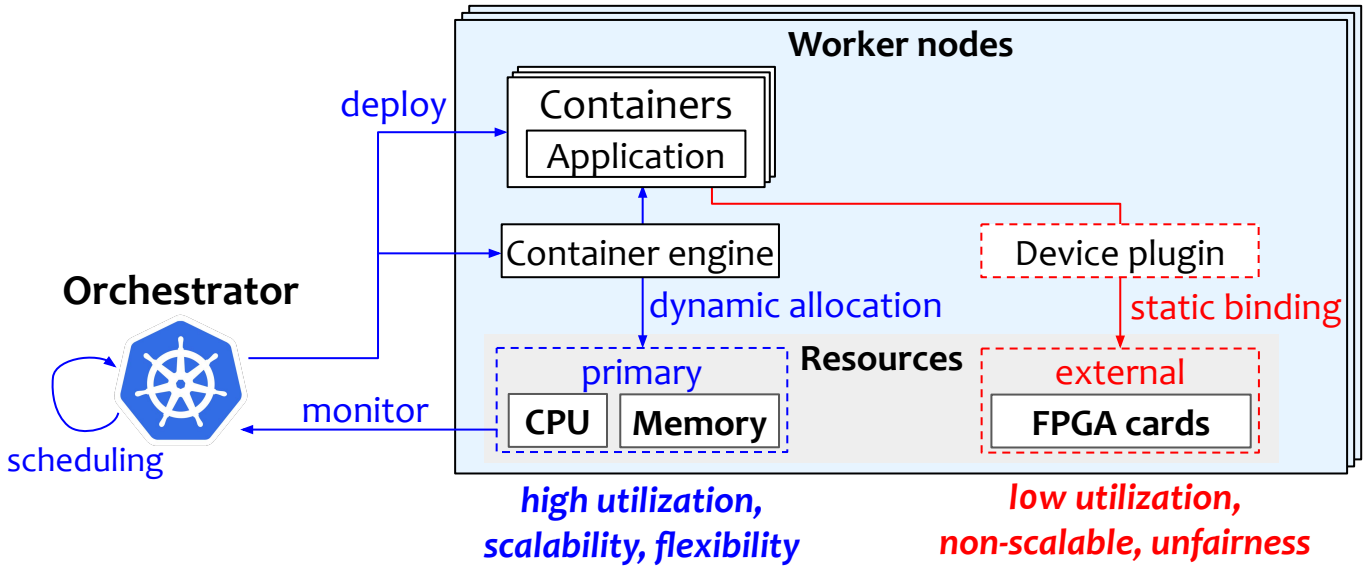




FPGA promises accelerating cloud workloads

Limited FPGA support in cloud-native environments

Cloud orchestrators do ***not*** natively support FPGAs

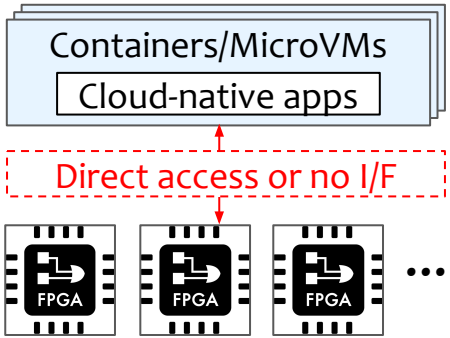


Orchestrators need to adapt FPGAs as primary hardware resources

Challenges

FPGA integration into cloud-native environments is hindered by:

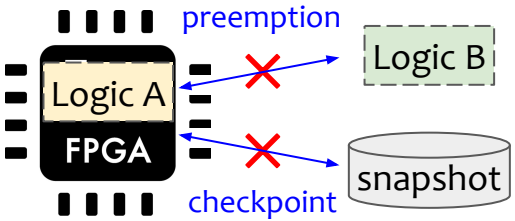
No FPGA virtualization



FPGAs unfit to lightweight sandboxes for cloud-native apps

→ ***Low FPGA utilization***

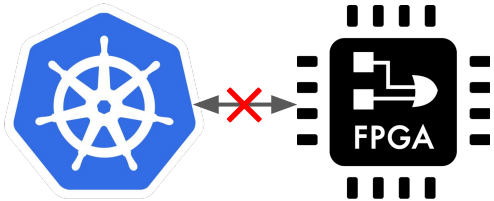
No FPGA preemption



Long-running apps can unfairly occupy FPGAs

→ ***Unfairness/data loss***

Limited FPGA orchestration



Open specifications (OCI, CRI) are only for CPU/memory

→ ***Impractical***

Research question

How do we **adapt FPGAs into cloud-native environments** to realize FPGA orchestration services, including task preemption, migration, and checkpointing?

Our proposal

Funky: Cloud-native FPGA virtualization and orchestration
an end-to-end FPGA orchestration engine for cloud-native applications

Contributions:

- Lightweight FPGA virtualization
  - A lightweight OS (*unikernel*) designed for FPGA applications
- FPGA state management
  - Hypervisor-driven task preemption, migration, and checkpointing
- FPGA-aware orchestration
  - CRI/OCI-compliant orchestrator extension

Outline

- ~~Motivation~~
- Overview
- Implementation
- Evaluation

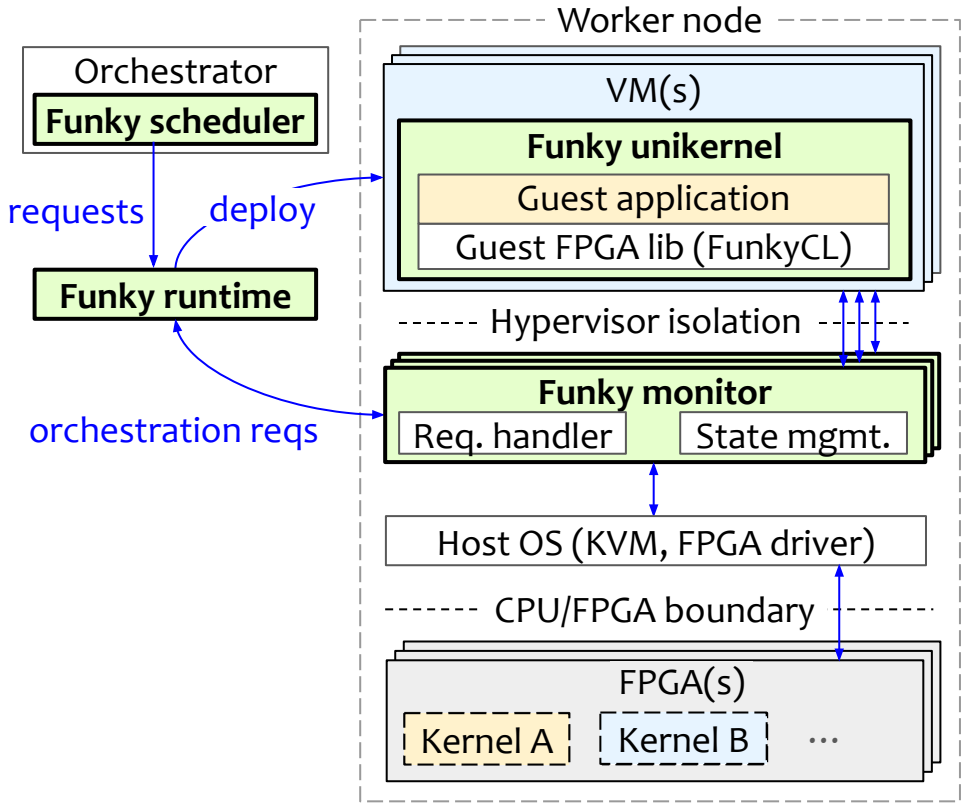
Funky overview

Key system components:

#1 Funky unikernel
→ FPGA virtualization

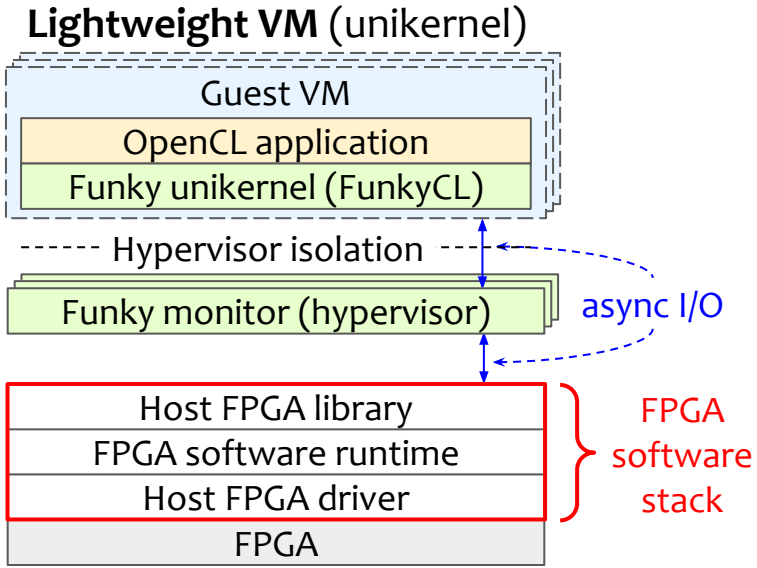
#2 Funky monitor
→ FPGA state management

#3 Funky orchestrator
→ FPGA-aware orchestration



#1: Funky unikernel

Preserve advantages of lightweight sandboxes (***unikernel***)



- Thin FPGA virtualization stack
  - *Lightweight context (< 10MB)*
- Asynchronous, exit-less FPGA I/Os
  - *Low runtime overhead (≈ 0%)*
- Hypervisor-driven isolation
  - *Strong isolation*
- OpenCL wrapper library (***FunkyCL***)
  - *Application portability*

Funky unikernel enables FPGA virtualization suited for cloud-native applications

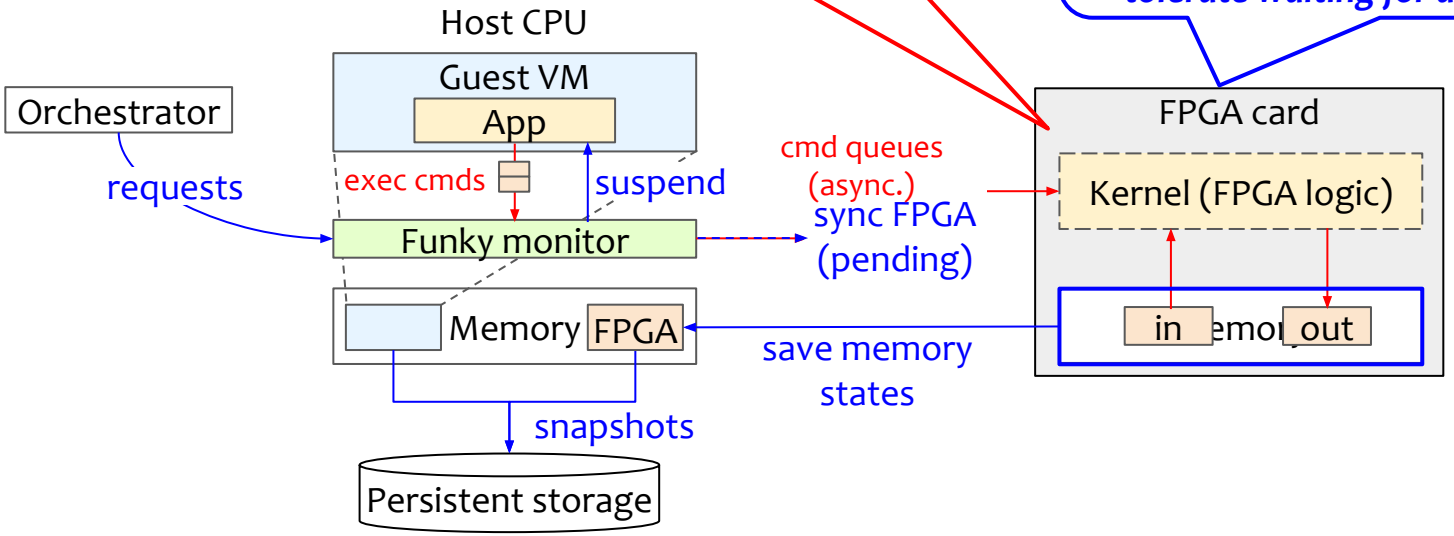
#2: Funky monitor

A hypervisor process e ing

Challenges
1. Kernel can't be suspended
2. FPGA states are unreadable

Observations
1. Funky monitor can track enqueued cmds
2. FPGA kernels are stateless (e.g., OpenCL)

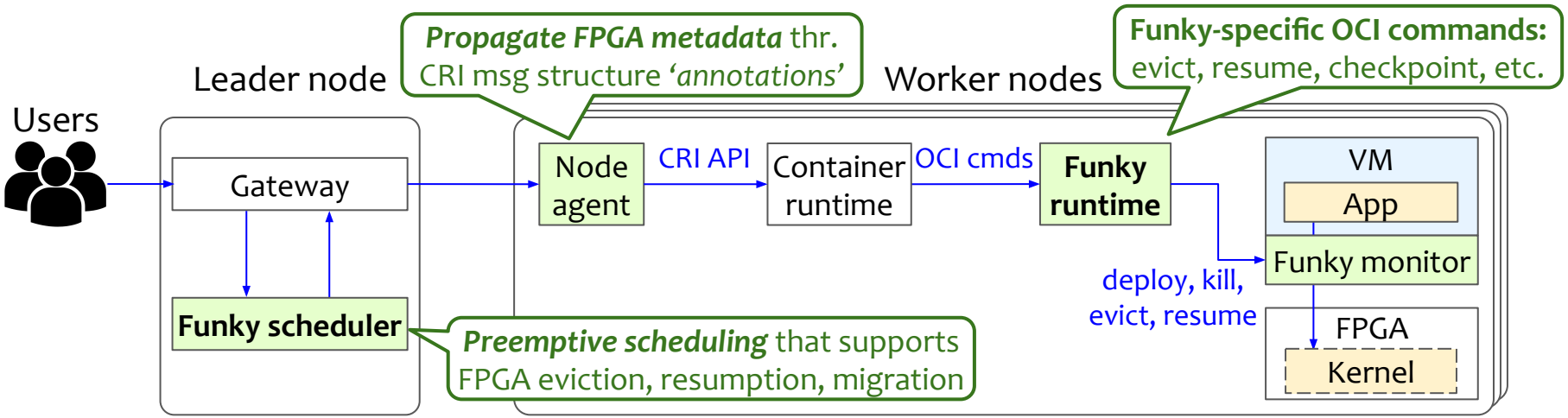
→ ***tolerate waiting for all on-the-fly commands***



Funky monitor can transparently/securely capture VM and FPGA contexts

#3: Funky orchestrator

CRI/OCI-compliant extension for Funky applications (**green boxes**)



Funky orchestrator transpantly enables orchestration services for FPGAs

Outline

- ~~Motivation~~
- ~~Overview~~
- Implementation
- Evaluation

Implementation

An end-to-end prototype for AMD FPGAs

- Funky orchestrator : Full-scratch implementation
- Funky unikernel : IncludeOS v0.13.1
- Funky monitor : Solo5 v0.3.1
- FunkyCL : Ported & tested 34 OpenCL APIs

FPGA backends

- FPGA runtime : Xilinx Runtime (XRT) 2021.2
- FPGA card : AMD Alveo U50

Outline

- ~~Motivation~~
- ~~Overview~~
- ~~Implementation~~
- Evaluation

Evaluation

Highlighted evaluations:

- End-to-end performance (virtualization overheads)
- Fault tolerance (checkpointing)

See our paper for more results!

Experimental setup:

- Cluster : 1x leader (Xeon G5317@3.0GHz), 3x workers (Xeon G6238R@2.2GHz)
- FPGA : 3x FPGAs (each worker node equipped with 1x Alveo U50)

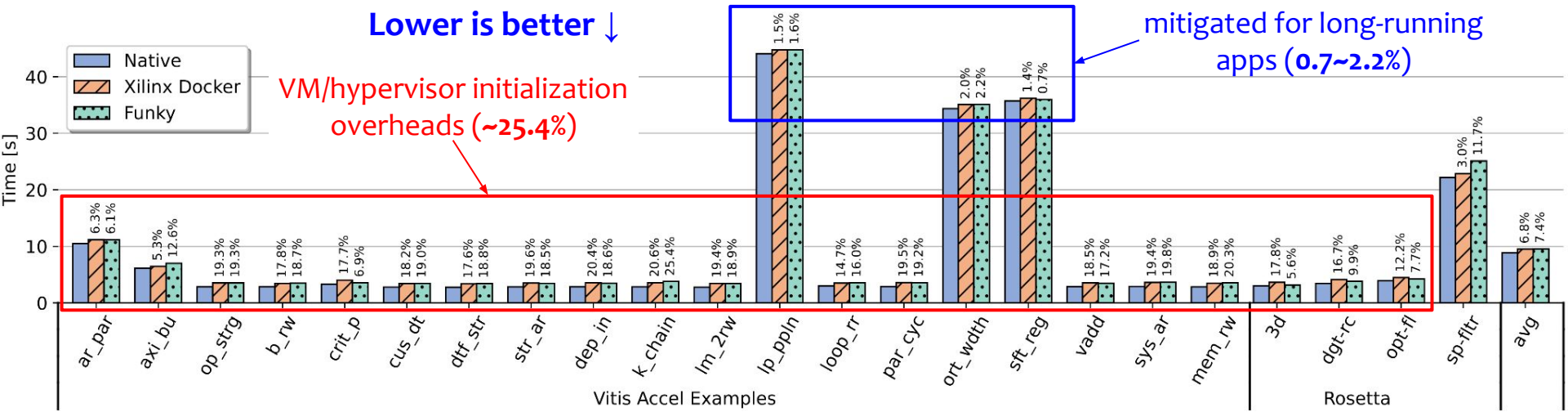
Applications:

- 23 OpenCL applications ported from Vitis Accel Examples and Rosetta[1]
- Google production traces of Borg clusters[2] for large-scale simulation

[1]Rosetta: A Realistic High-Level Synthesis Benchmark Suite for Software Programmable FPGAs, FPGA'18.
[2]Borg: The Next Generation, EuroSys'20

End-to-end performance (virtualization overheads)

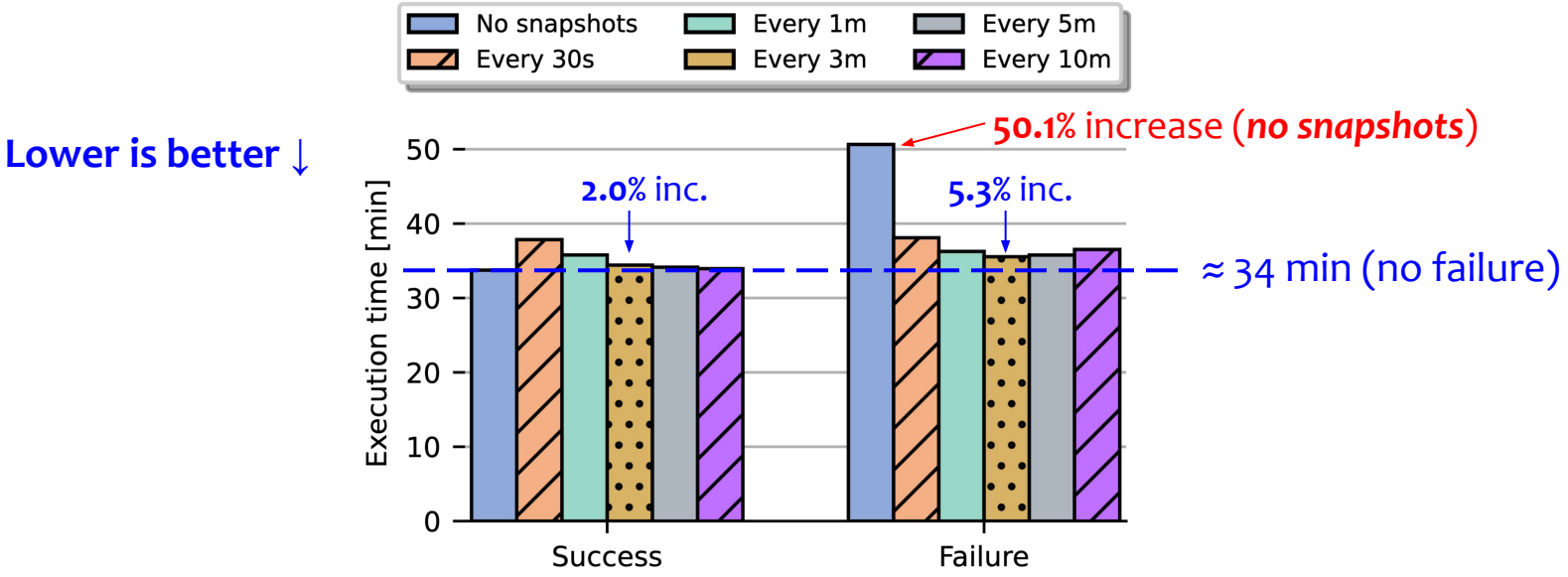
Execution time ***against baselines w/o FPGA virtualization*** (native, AMD's container)



Funky enables ***FPGA virtualization*** with **0.6% higher overheads** than containers

Fault tolerance (checkpointing overheads)

Execution time including recovery time in failure, simulated by Google job traces



Funky mitigates **44.7%** of recovery time at the cost of **2.0% overheads** in success

Summary

Lack of FPGA virtualization and orchestration support for cloud-native applications

- ***Lack of FPGA virtualization*** suitable for cloud-native applications
- ***Lack of FPGA state management*** for orchestration services
- ***Limited orchestration support*** for FPGAs

Funky: an end-to-end FPGA orchestration engine

- ***Lightweight FPGA virtualization*** for unikernels
- ***FPGA state management*** driven by a hypervisor
- ***Orchestrator extension*** that is OCI/CRI-compatible

Paper

Code

https://github.com/TUM-DSE/Funky

Contact : atsushi.koshiba@tum.de
Website : https://atsushikoshiba.github.io