Investigating the Performance and Scalability of Kubernetes on Distributed Cluster of Resource-Constrained Edge Devices

Vahid Daneshmand*, Renato Figueiredo*, Kohei Ichikawa**,

Keichi Takahashi**, Kundjanasith Thonglek**, Kensworth Subratie*

* ACIS Lab, University of Florida, USA

** Nara Institute of Science and Technology (NAIST), Japan

Background and Motivation

Edge Computing: From Centralized Nodes → To Logical Extremes of Network

Application, Data, Service Deployment: Containers - Docker

Problem: Deploy and Manage Containers in Scale

Solution: Container Orchestration - Kubernetes

Question: Performance of Kubernetes in Widely Distributed Edge Nodes?

Distributed Testbed for Edge Computing

24 Raspberry Pi 3B+ nodes on two sites:

- Master Node + 16 Nodes at University of Florida, USA
- 8 Nodes at Nara Institute of Science and Technology, Japan

Virtualization and Orchestration: Docker and Kubernetes

Virtual Network: IPOP VPN

You're welcome to join/use the cluster!

Experiments Methodology

First Scenario: 8 local nodes at UF (UF-8) vs 8 non-local nodes at NAIST (NAIST-8)

four different-size Docker images:

- Tiny Image: hello-world (1.64kB)
- Small Image: eclipse-mosquitto (4.6MB)
- Medium Image: ubuntu (46.7MB), nginx (97.6MB)
- Large Image: python (708MB)

Docker images pre-pulled on the pods

Repeat experiments for 3 times

Second Scenario: 16 local nodes at UF (UF-16) versus a hybrid setup of 8 local nodes at UF plus 8 non-local nodes at NAIST (UF-NAIST-16)

Kubernetes Cluster Benchmarking

kbench*, a Kubernetes benchmarking test suite:

pod-throughput Test (*kbench pod-throughput -n <number> -i <image>*)
Launches *n* pods in parallel and measures their start-up and clean-up time.

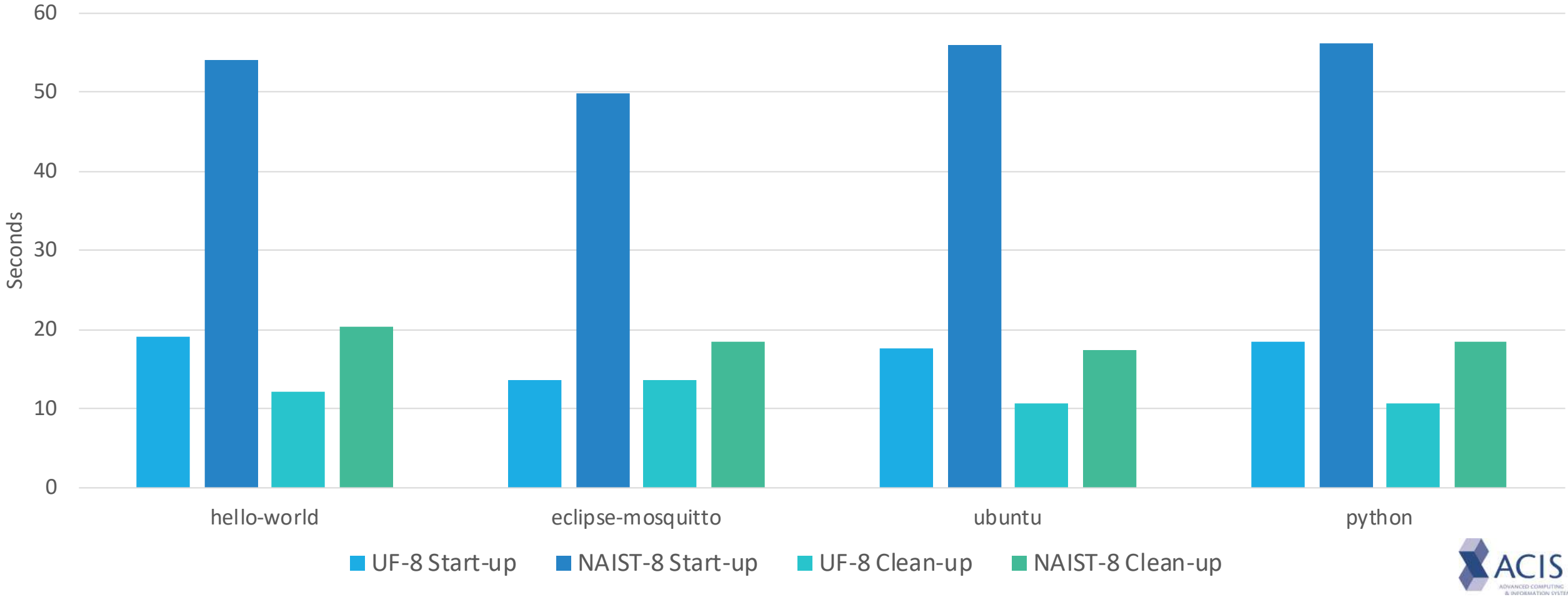
pod-latency Test (*kbench pod-latency -n <number> -i <image>*)
Launches *n* pods sequentially and measures their start-up and clean-up time.

deployment-scaling Test (*deployment-scaling –m <number> -n <number> -i <image>*)
Creates a deployment and measures scale-up/down latency. First, a deployment with *m* pods is created. Then, the deployment is scaled-up to *n* pods. Once the scale-up is completed, the deployment is scaled-down to *m* pods, again.

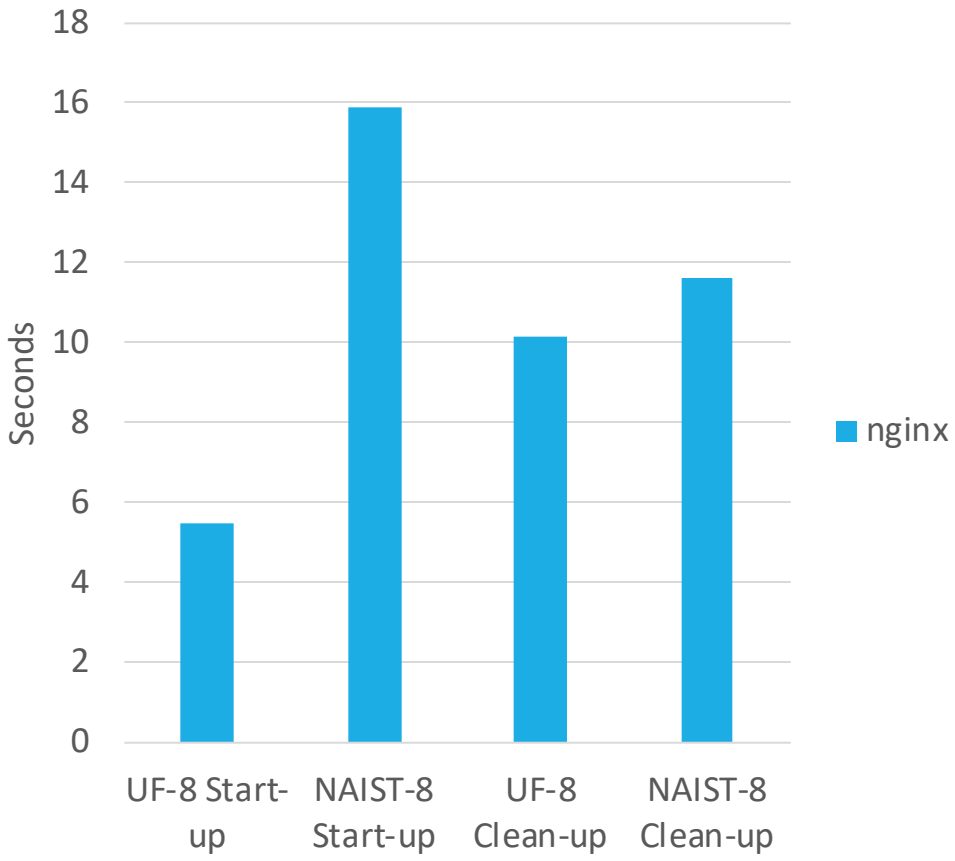
*https://github.com/keichi/kbench

ACIS
ADVANCED COMPUTING & INFORMATION SYSTEMS

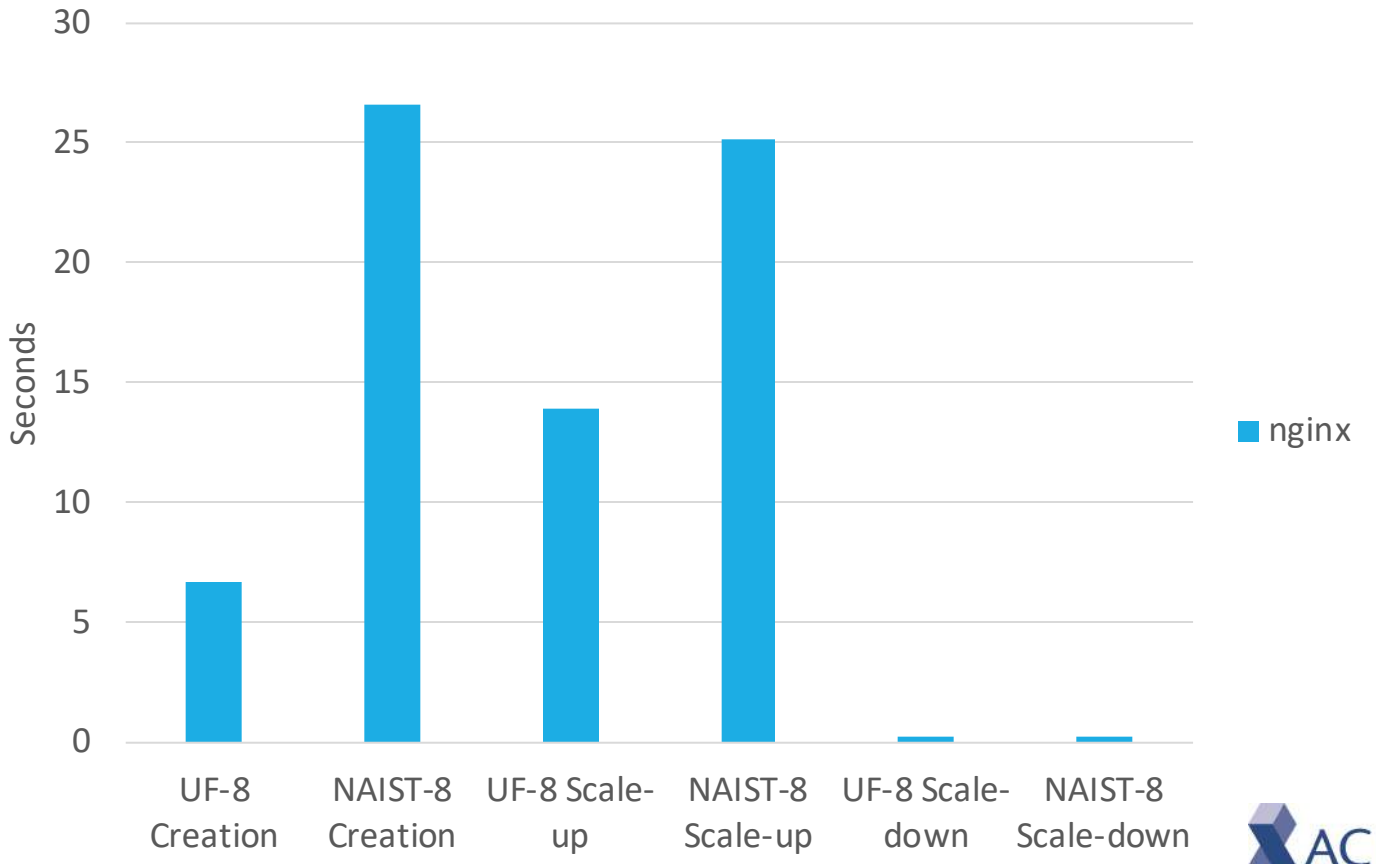
Pod-Throughput Test: UF-8 vs NAIST-8
Average Start-up, Clean-up Time for 48 Pods



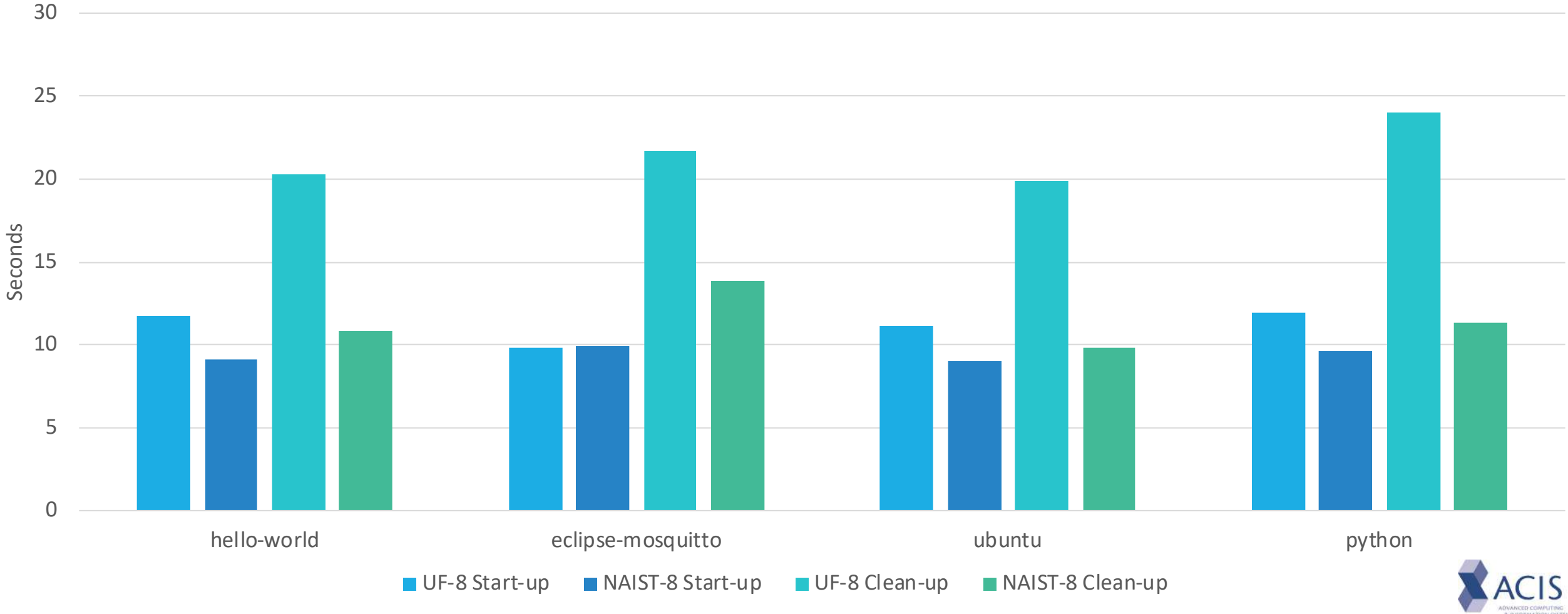
Pod-Latency Test: UF-8 vs NAIST-8 Average Start-up, Clean-up Time for 8 Pods



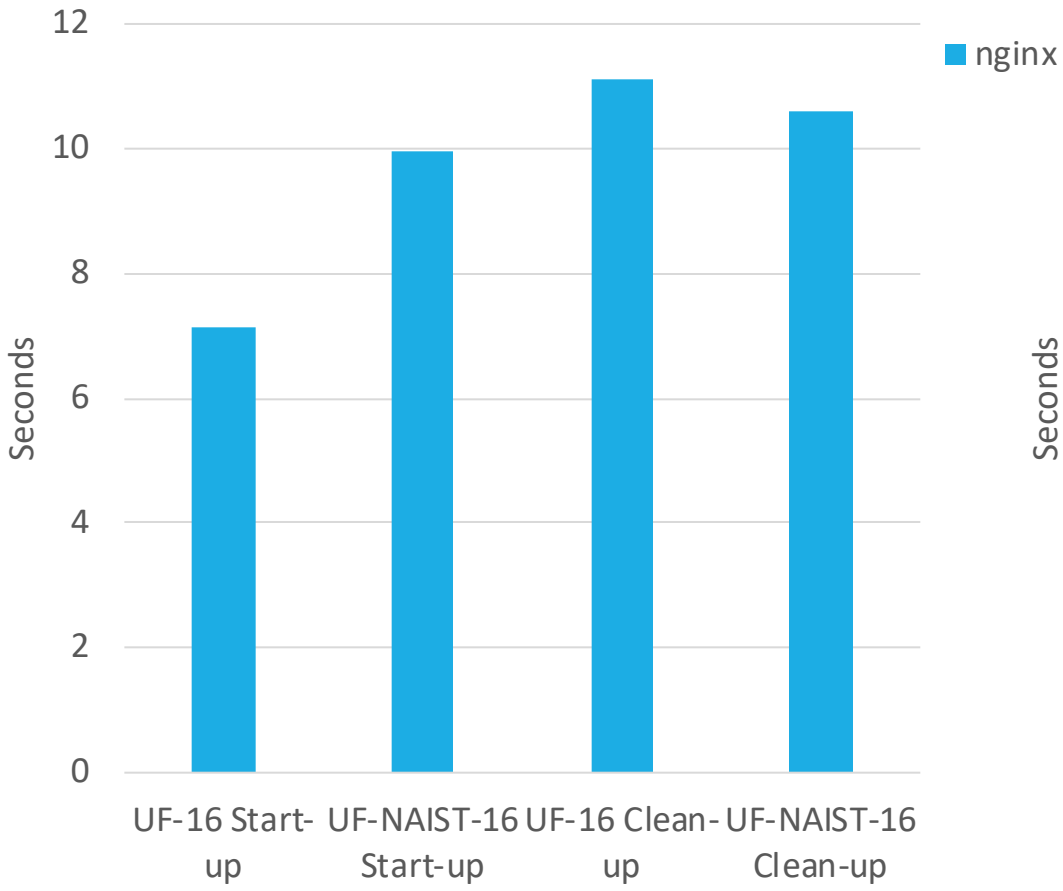
Deployment-Scaling Test: UF-8 vs NAIST-8 Average Creation, Scale-up, Scale-down Time for 4 to 8 Pods



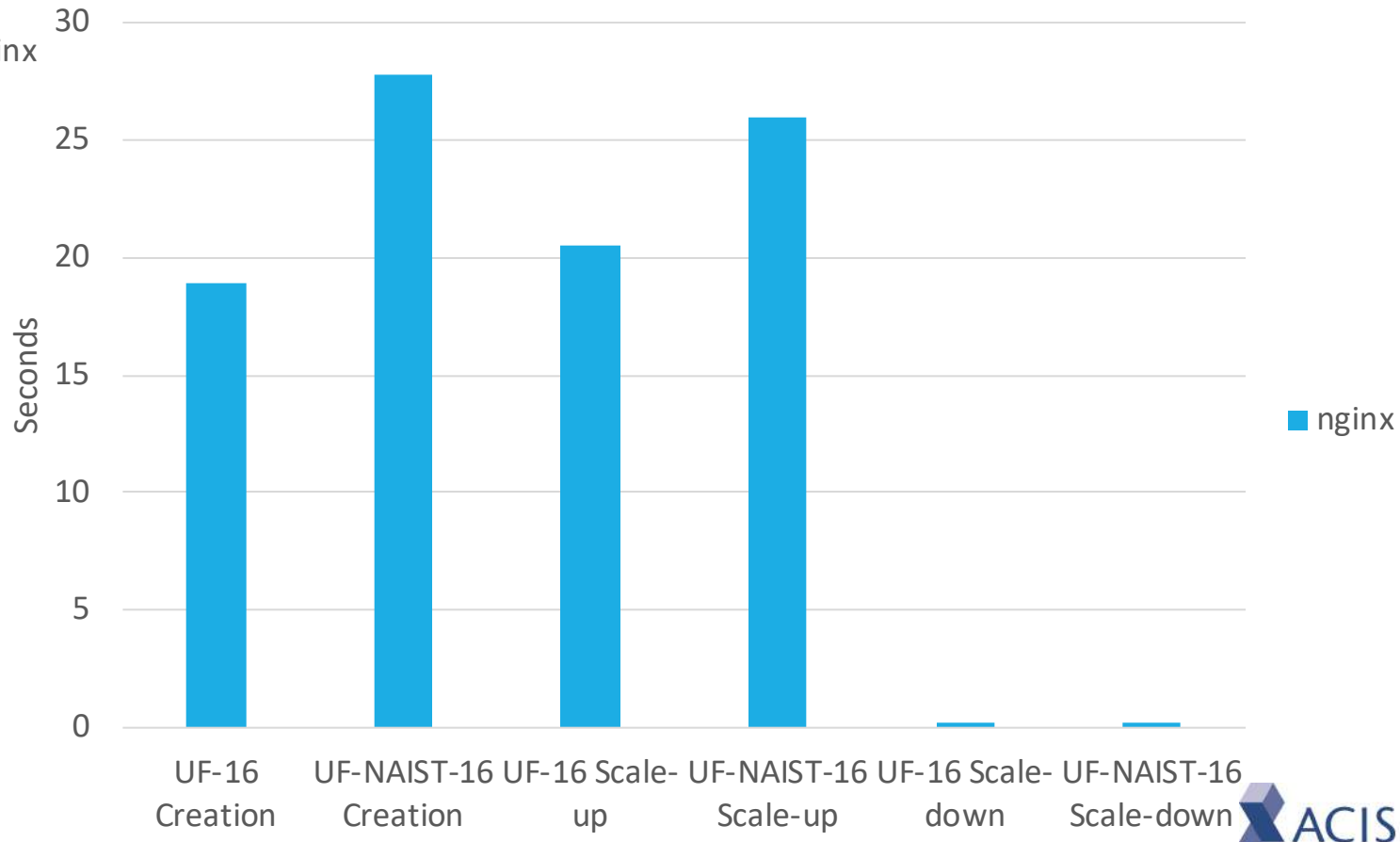
Pod-Throughput Test: UF-16 vs UF-NAIST-16
Average Start-up, Clean-up Time for 48 Pods



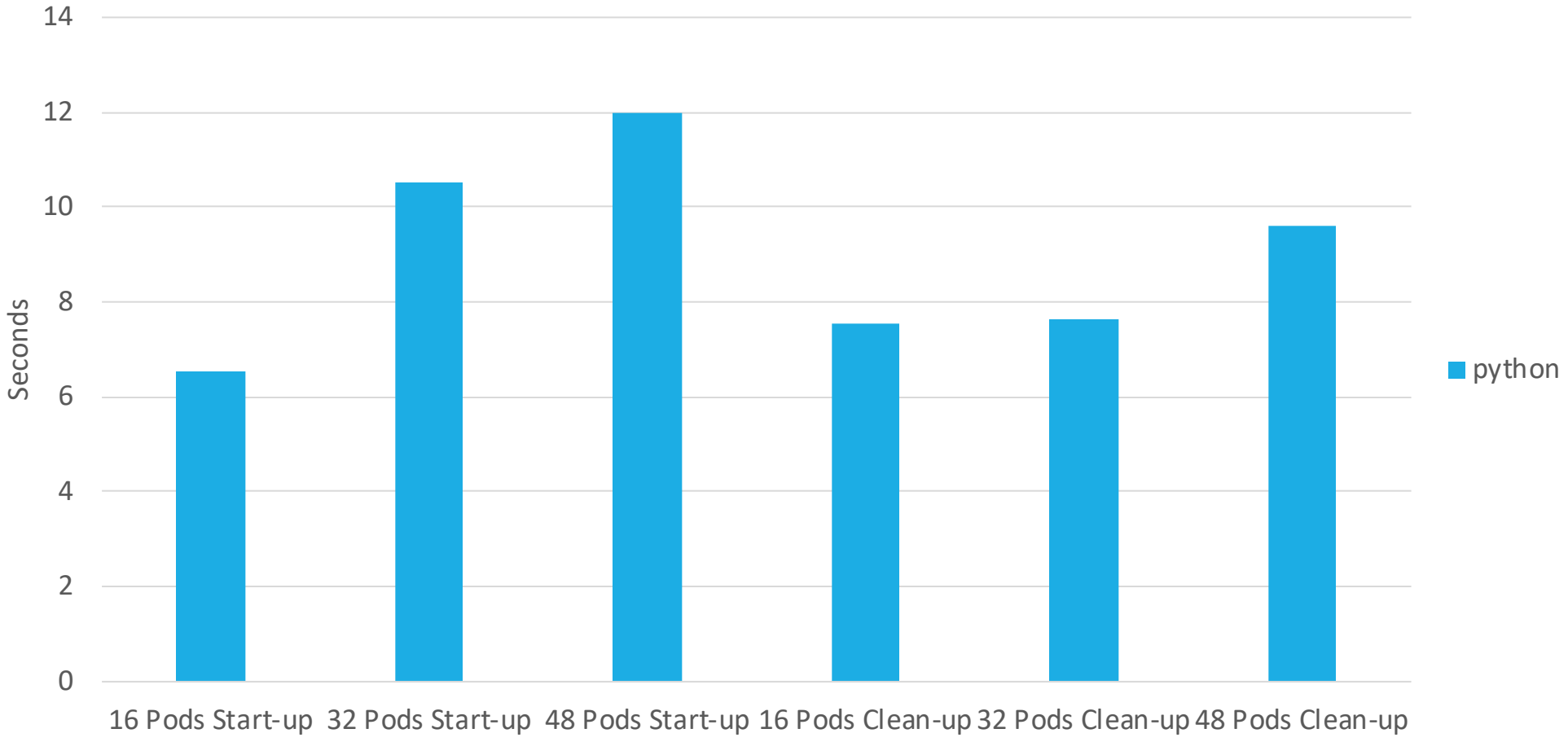
Pod-Latency Test: UF-16 vs UF-NAIST-16 Average Start-up, Clean-up Time for 16 Pods



Deployment-Scaling Test: UF-16 vs UF-NAIST-16 Average Creation, Scale-up, Scale-down Time for 8 to 16 Pods



Pod-Throughput Test: UF-16
Average Start-up, Clean-up Time for 16, 32, 48 Pods



Conclusion

Pod start-up time vastly lower in Local vs non-local nodes

Pod clean-up time, difference much less notable

No correlation between the size of the Docker image and the pod start-up and clean-up time

Start-up and clean-up time increases with the rise of total number of pods. More notable for start-up than clean-up time.