DIPDCE: OFFLOADING-AWARE VISION INFERENCE IN EDGE WITH CONCURRENT EXECUTION

ABHINABA CHAKRABORTY, WOUTER TAVERNIER, MARIO PICKAVET, DIDIER COLLE

OVERVIEW

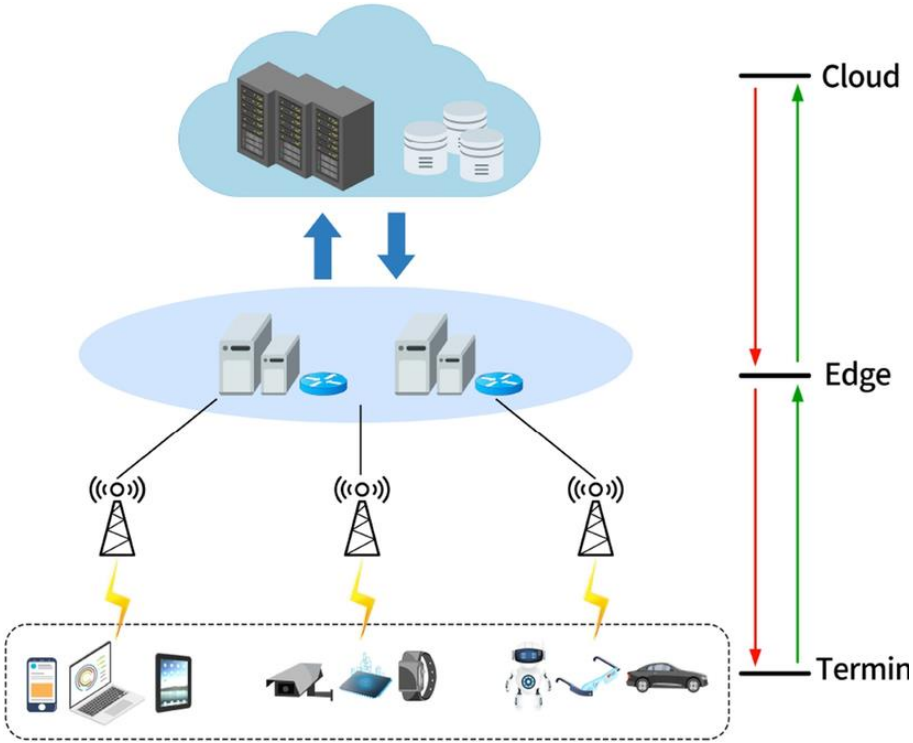
- Introduction
- Related Works
- Motivation and Use cases
- System Architecture
- System Modelling
- Experimental Setup
- Results
- Conclusion

INTRODUCTION

- Distributed Computer vision use hierarchical tiers

$$Sensor \leftrightarrow edge \leftrightarrow cloud$$

- Edge tier reduces latency and backbone traffic
- Challenges
  - Heterogeneous hardware, limited battery, unreliable networks
  - Imbalanced workloads →QoS violations
- Prior Approaches
  - Optimisation-based placement (ILP, Heuristics)
  - Adaptive scheduling
- Proposed DipDCE – benchmark-driven optimization for predictable workloads



BACKGROUND

FRAMEWORKS FOR STREAMING IMAGE DATA

- Monitoring for streaming workloads
- AI power virtual streamer framework
- ABUV combines adaptive bitrate selection with super resolution
- Sampling based dual-stream hybrid network
- Multi-radio 5G connections

BACKGROUND

HYBRID EDGE-CLOUD DECISION MODELS

DeepDecision

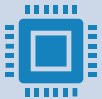
VideoPipe, VideoStorm

Several researches exists which optimizes latency, energy consumption, per-frame delay etc

BACKGROUND

CONCURRENT PROCESSING OF IMAGE STREAMS

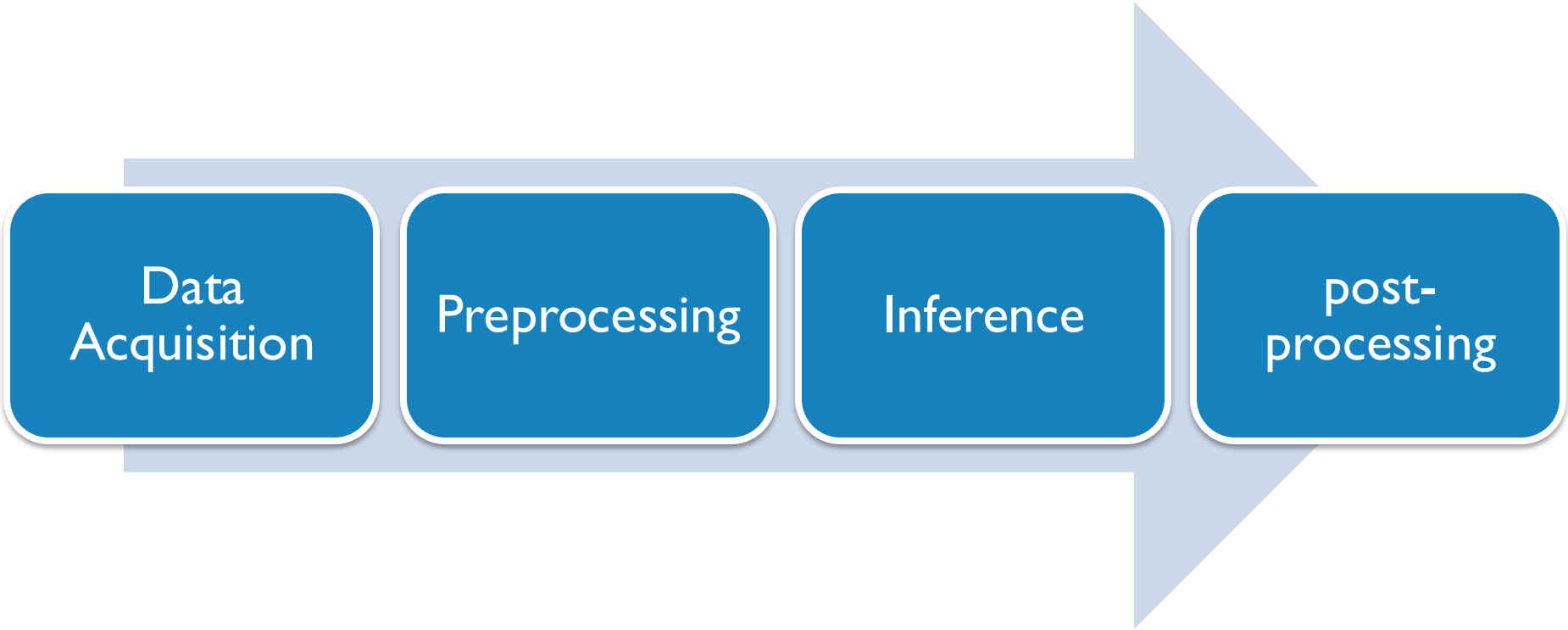
Concurrency exists for to increase the overall throughput but achieving GPU level concurrency is still hard and requires careful configuration.



Time multiplexing, space multiplexing, MPS exists and having multiple process running on the same hardware had adverse effects

There are some study which explores that study but not a structured study exists for this domain.

BACKGROUND
VISION INFERENCE PIPELINES

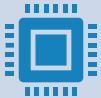


BACKGROUND

VISION INFERENCE PIPELINES… (CONTINUED)

For inference, the DL models needs to be optimized for the specific platforms

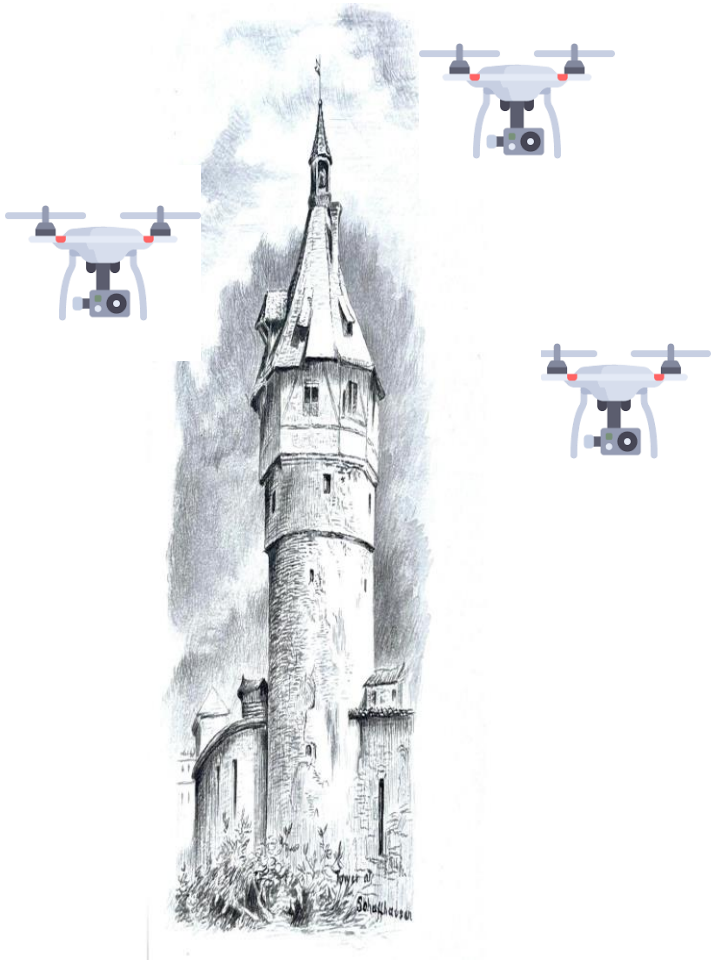
Precision reduction (fp32 to fp16, int8 etc), dead nodes removal, layer fusion etc.



NVIDIA TensorRT, TVM, Intel OpenVINO does these optimization for specific platforms

MOTIVATION
USE CASE

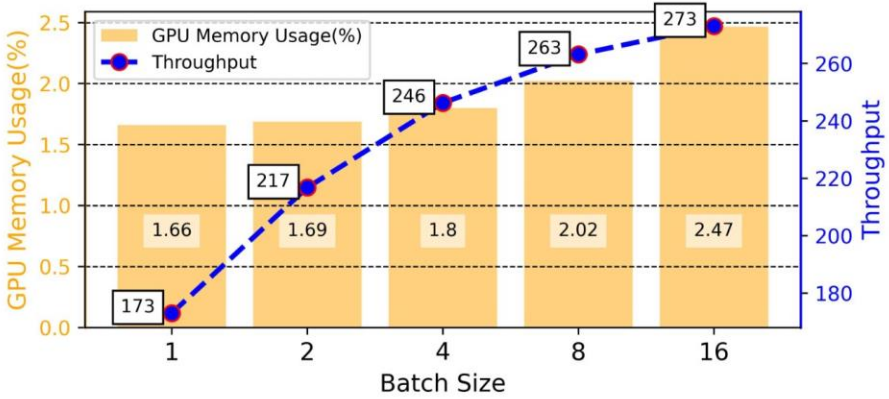
1. Multiple Drone/agents captures images
2. How can you process those captured imaged?
    1. Fit every drone with AI accelerators
    2. Maybe offloading can be an option



Images are taken from Google

MOTIVATION
WHY BATCHING?

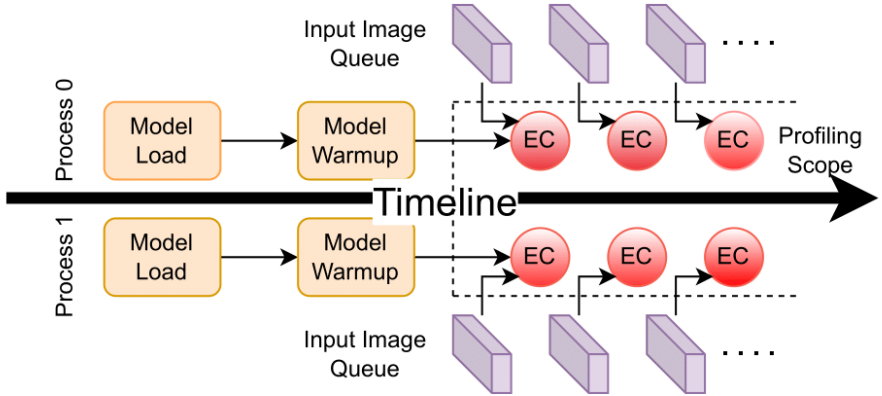
1. Serial Processing
   - What if $arrival\ rate > service\ rate$?
2. Solution:
   - Wait for some time, put some incoming images together and process (batching increases throughput)
   - But batching also increases average delay which may violate QoS
   - Increasing the batch size is also not a solution → throughput saturates



A. Chakraborty et.al, Profiling Concurrent Vision Inference Workloads on NVIDIA Jetson (ISPASS, 2025)

MOTIVATION
WHY CONCURRENT PROCESSING?

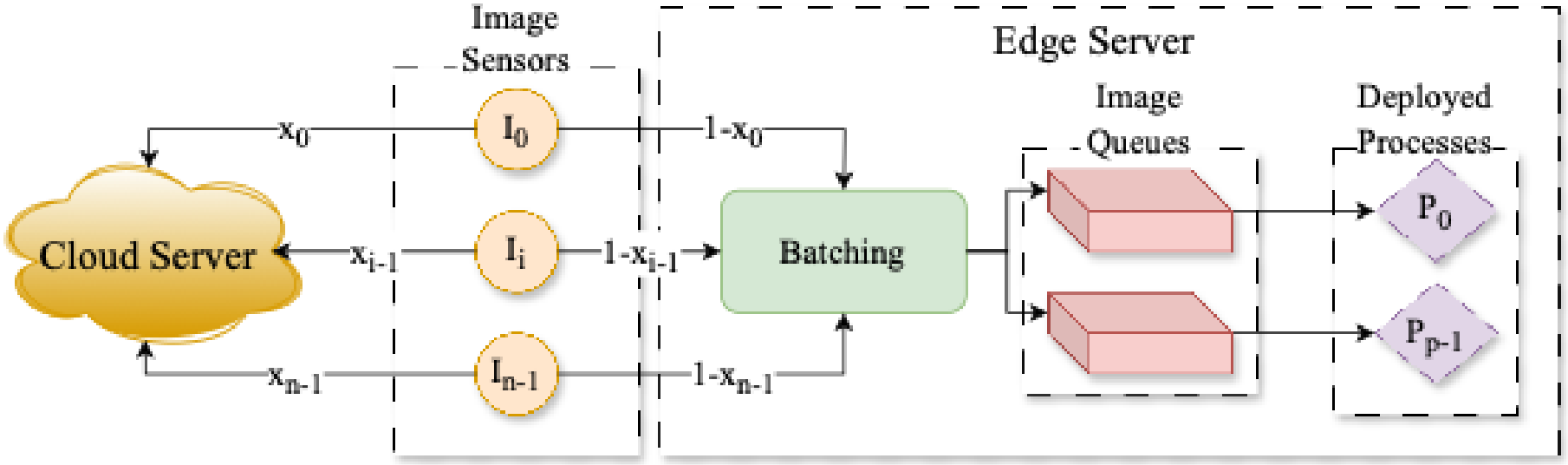
1. Infinite batching is not feasible
   - What if we deploy multiple processes
2. May/mayn't be good:
   - Multiple process always have an adverse effect on processing speed but how much ?
     - Any quantification ? Not yet
   - Space multiplexing/ time multiplexing/ MPS ?



A. Chakraborty et.al, Profiling Concurrent Vision Inference Workloads on NVIDIA Jetson-Extended (Arxiv)

PROPOSED FORMULATION

PROPOSED DEPLOYMENT STRATEGY – EDGE-CLOUD CONTINUUM



PROBLEM FORMULATION

PROPOSED DEPLOYMENT STRATEGY – EDGE-CLOUD CONTINUUM

$$min \frac{1}{n} \sum_i x_i + \lambda/2 \sum_i (x_i - E[x])^2$$

such that, $M_{avail} \geq p\{M_{cv} + bM_o\}$

$$max\{1 + g(b,p), f_i x_i (t_{cloud}^{prop} + t_{cloud}^{proc})\} \leq 1 + \chi_i$$

$$\frac{bp}{g(b,p)} \geq \sum_i f_i (1 - x_i)$$

Metric	Description
p	Number of deployed processes
b	Batch size
f_i	Frame rate of each sensor , i
x_i	Total % of image frame offloaded
$g(b,p)$	Latency of one inference
M_{avail}	Main memory available for edge
M_{cv}	Memory occupied by vision model
M_o	Memory occupied by one image
$t_{cloud}^{proc}, t_{cloud}^{prop}$	Cloud network, processing delay
χ_i	Affordable delay (provided by user)

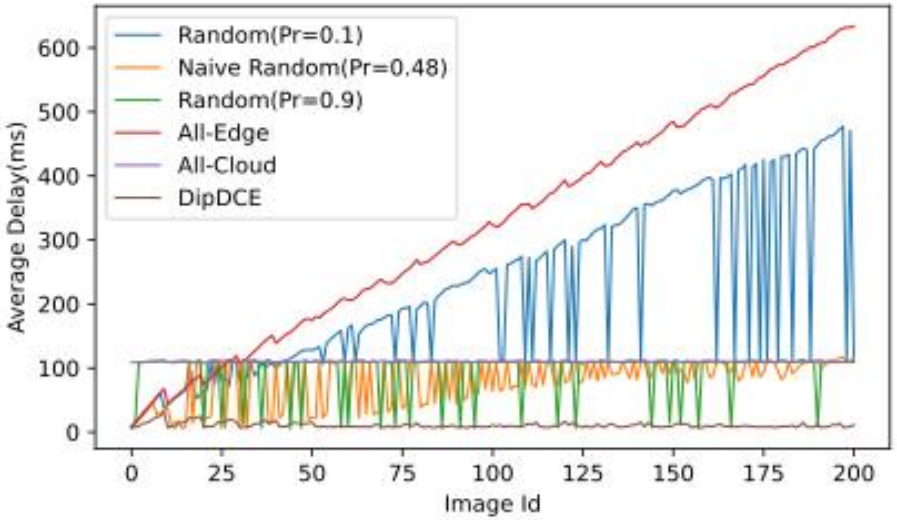
EXPERIMENTS

TESTBEDS

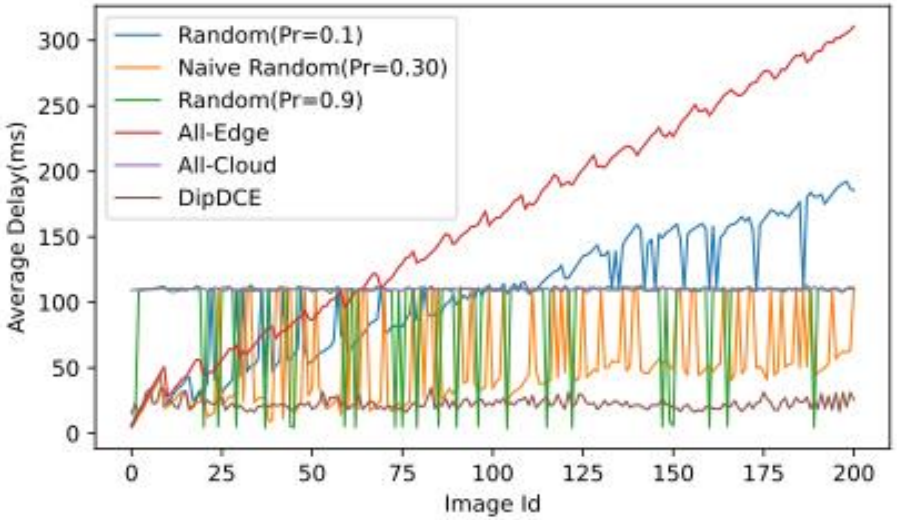
Metric	Jetson Orin Nano	GTX 1080Ti
CPU	6-Core Arm Cortex	6 Core Intel i5
GPU	1024 Core Ampere	3584-core Pascal
Tensor Cores	32	-
RAM	8GB (Unified)	11 GB(Discrete)
Power	7-15W	250W

Images are taken from Google

RESULTS
DELAY ANALYSIS



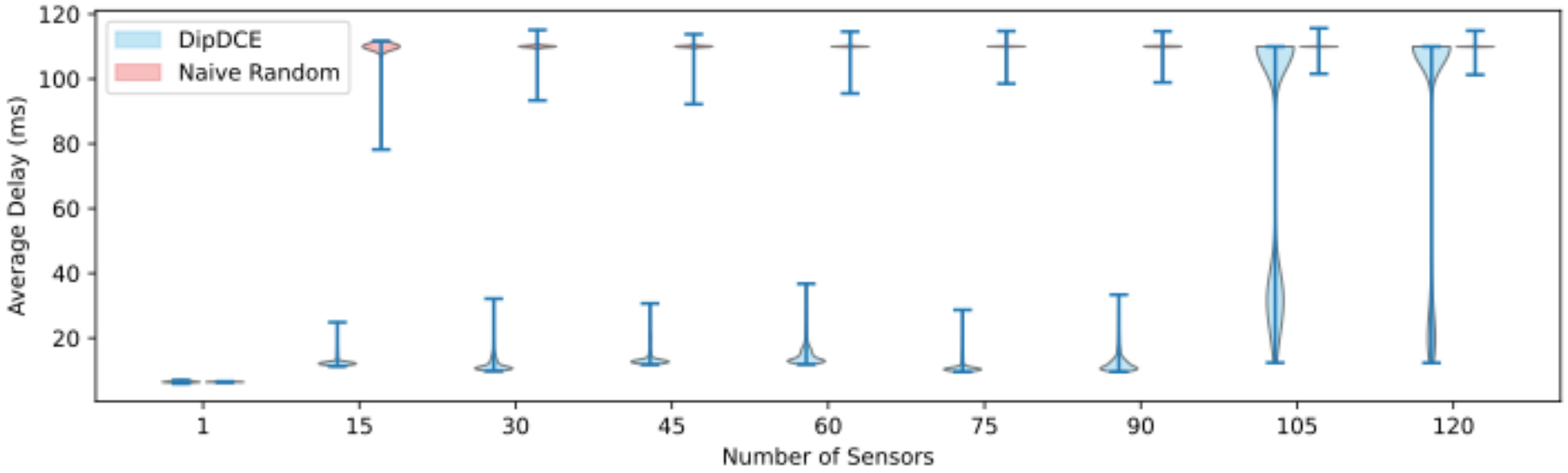
1080Ti



Jetson Orin Nano

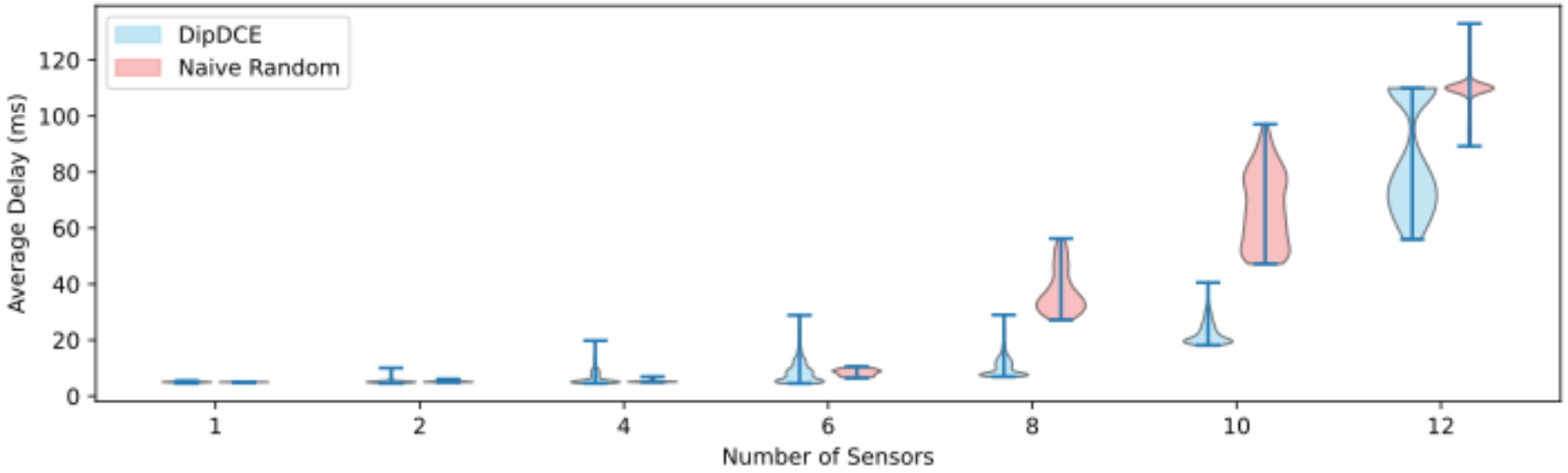
RESULTS

DELAY ANALYSIS – 1080TI



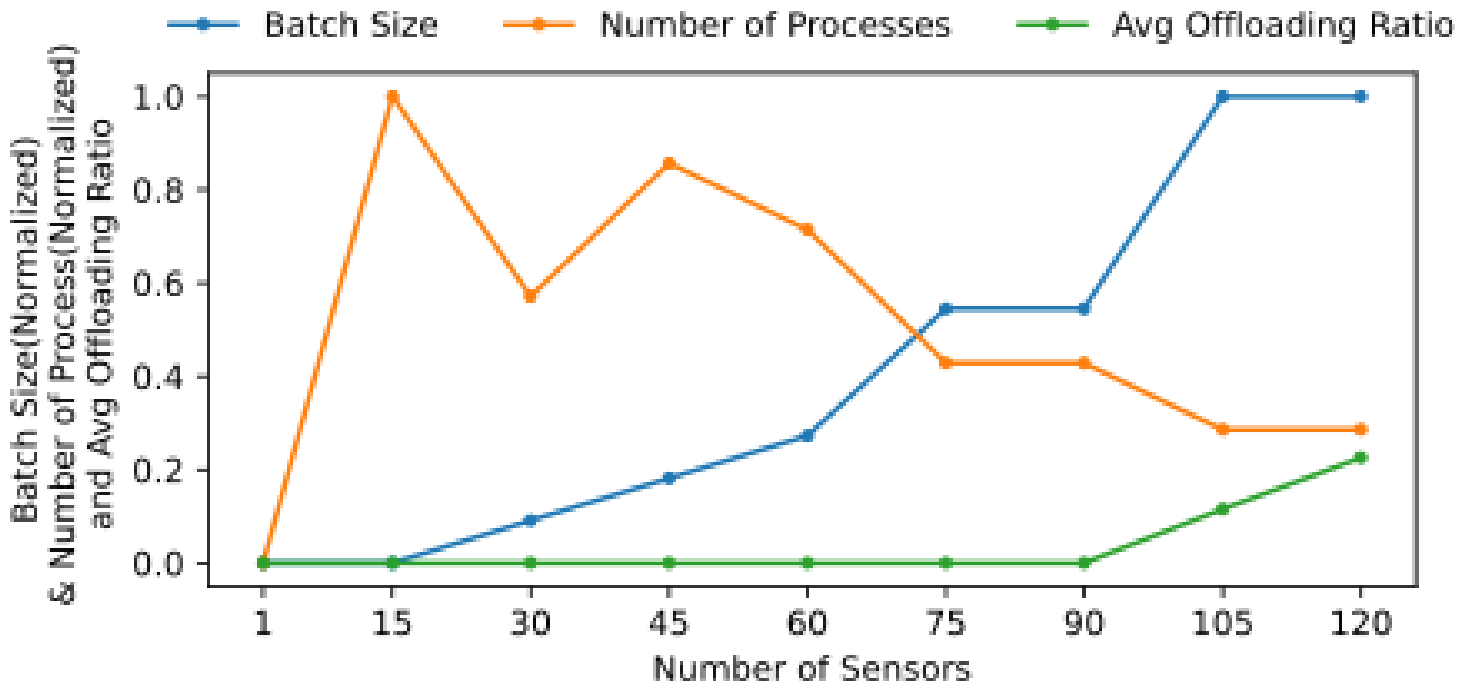
RESULTS

DELAY ANALYSIS – JETSON ORIN NANO



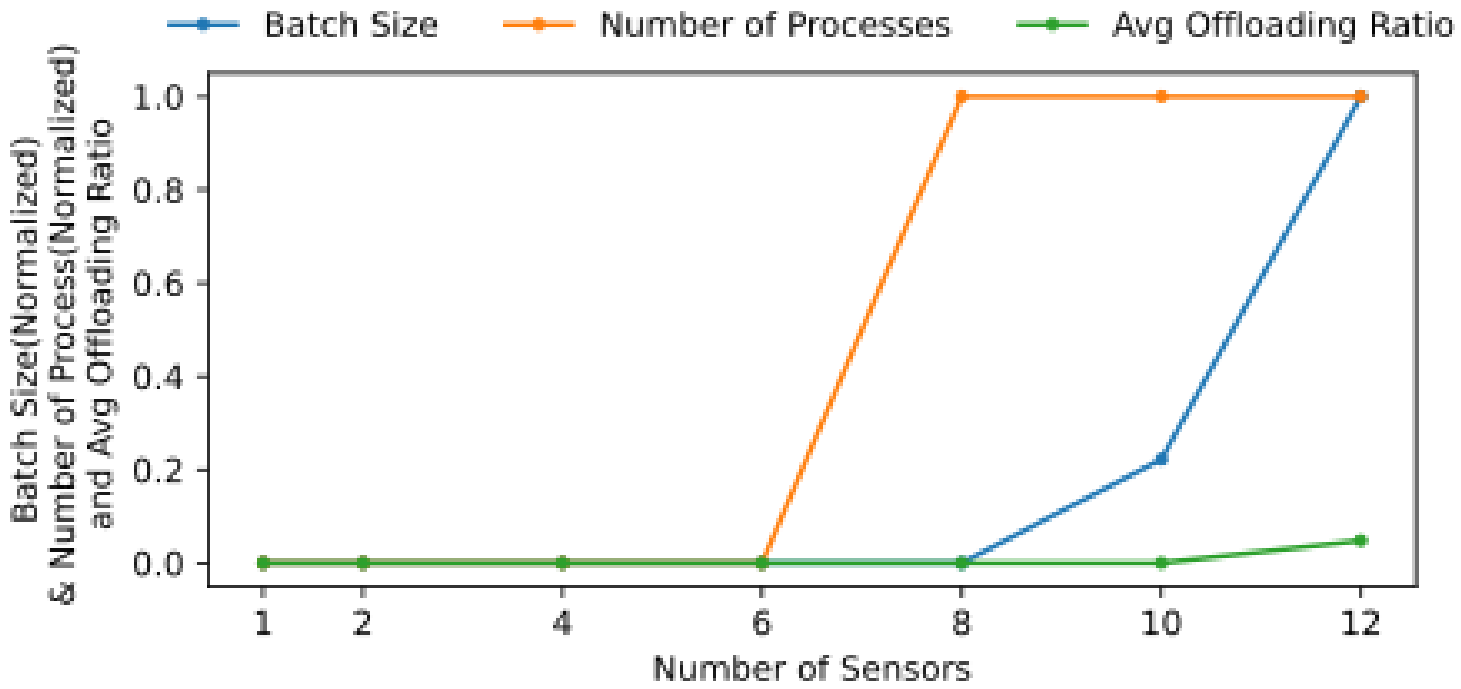
RESULTS

OFFLOADING PERCENTAGE ANALYSIS – 1080TI



RESULTS

OFFLOADING PERCENTAGE ANALYSIS – JETSON ORIN NANO



CONCLUSION

Edge Concurrent Processing to minimize delay and improve utilization

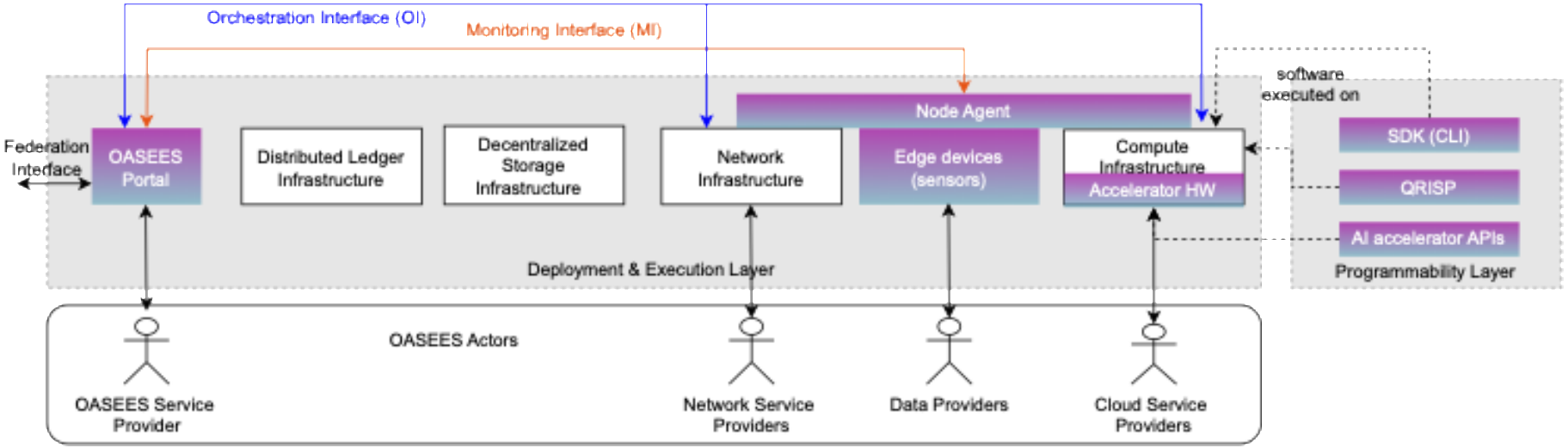
Our results demonstrate that QoS requirements can be met predominantly by concurrent processing, batching and optional cloud offloading

For future work will be focused on, dynamic workloads, handling failures etc.

embracing a better life

SYSTEM ARCHITECTURE

OASEES – DEPLOYMENT LAYER



Chochliouros, Ioannis P., et al. "OASEES: An Innovative Scope for a DAO-Based Programmable Swarm Solution, for Decentralizing AI Applications Close to Data Generation Locations." *ARTIFICIAL INTELLIGENCE APPLICATIONS AND INNOVATIONS. AIAI 2023 IFIP WG 12.5 INTERNATIONAL WORKSHOPS*, edited by I Maglogiannis et al., vol. 677, Springer International Publishing AG, 2023, pp. 91–105