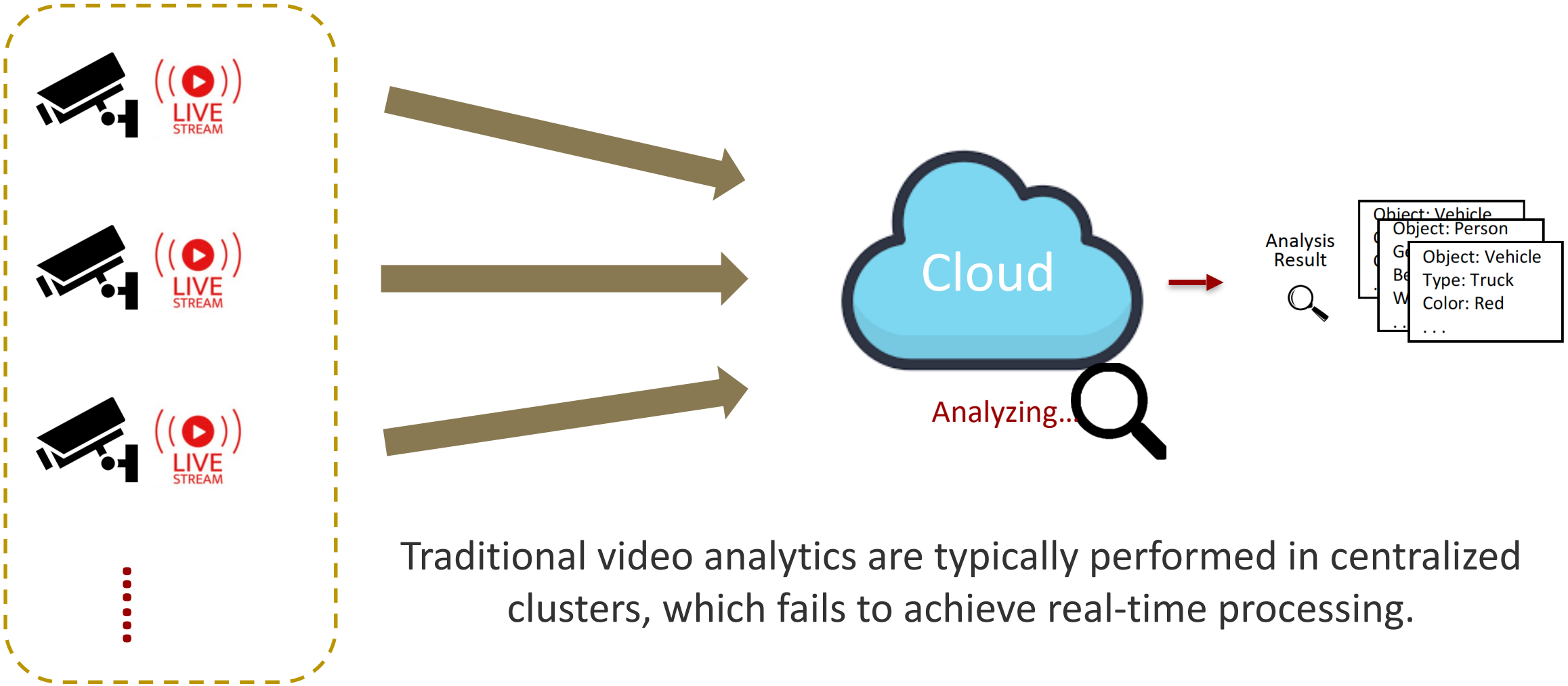
MICHIGAN STATE UNIVERSITY

Distream: Scaling Live Video Analytics with Workload-Adaptive Distributed Edge Intelligence

Xiao Zeng†, Biyi Fang†, Haichen Shen*, Mi Zhang†

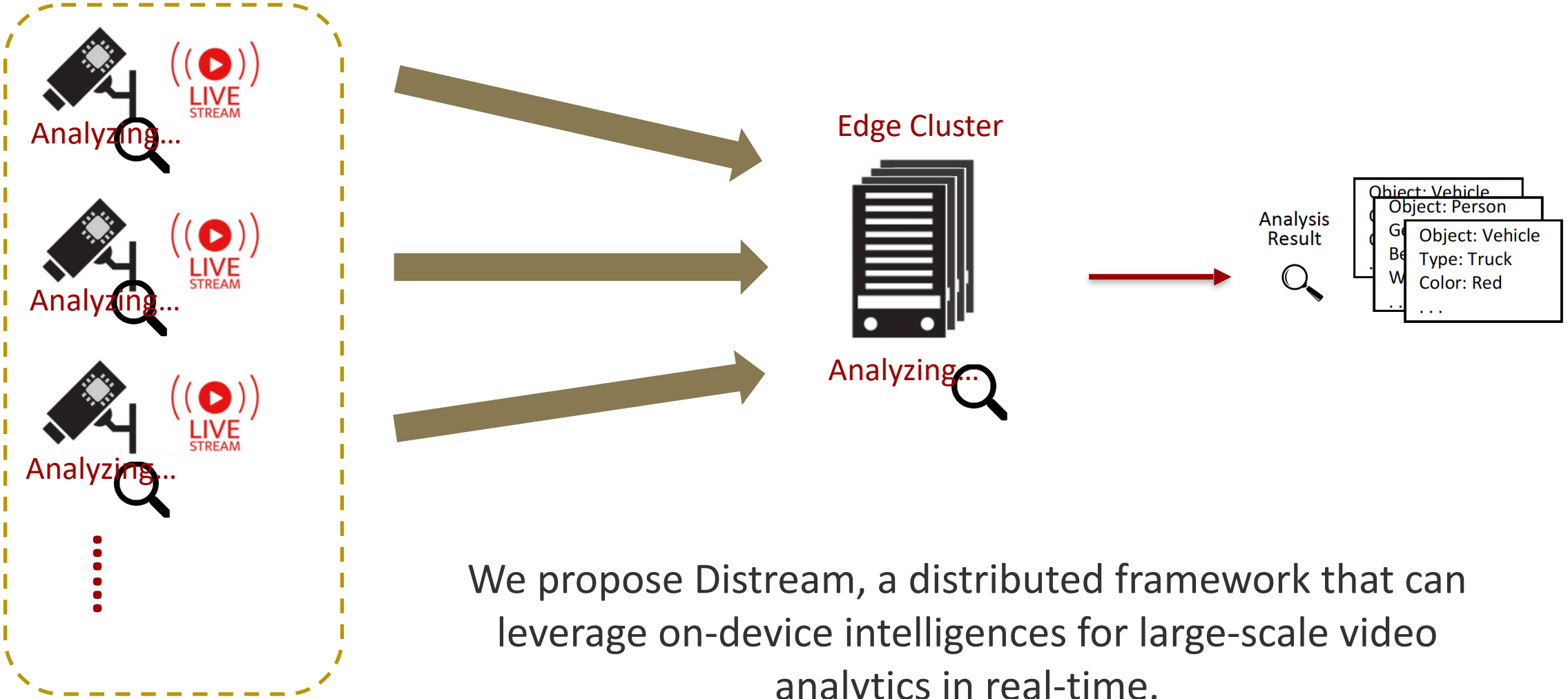
†Michigan State University, *Amazon Web Services

Introduction



Traditional video analytics are typically performed in centralized clusters, which fails to achieve real-time processing.

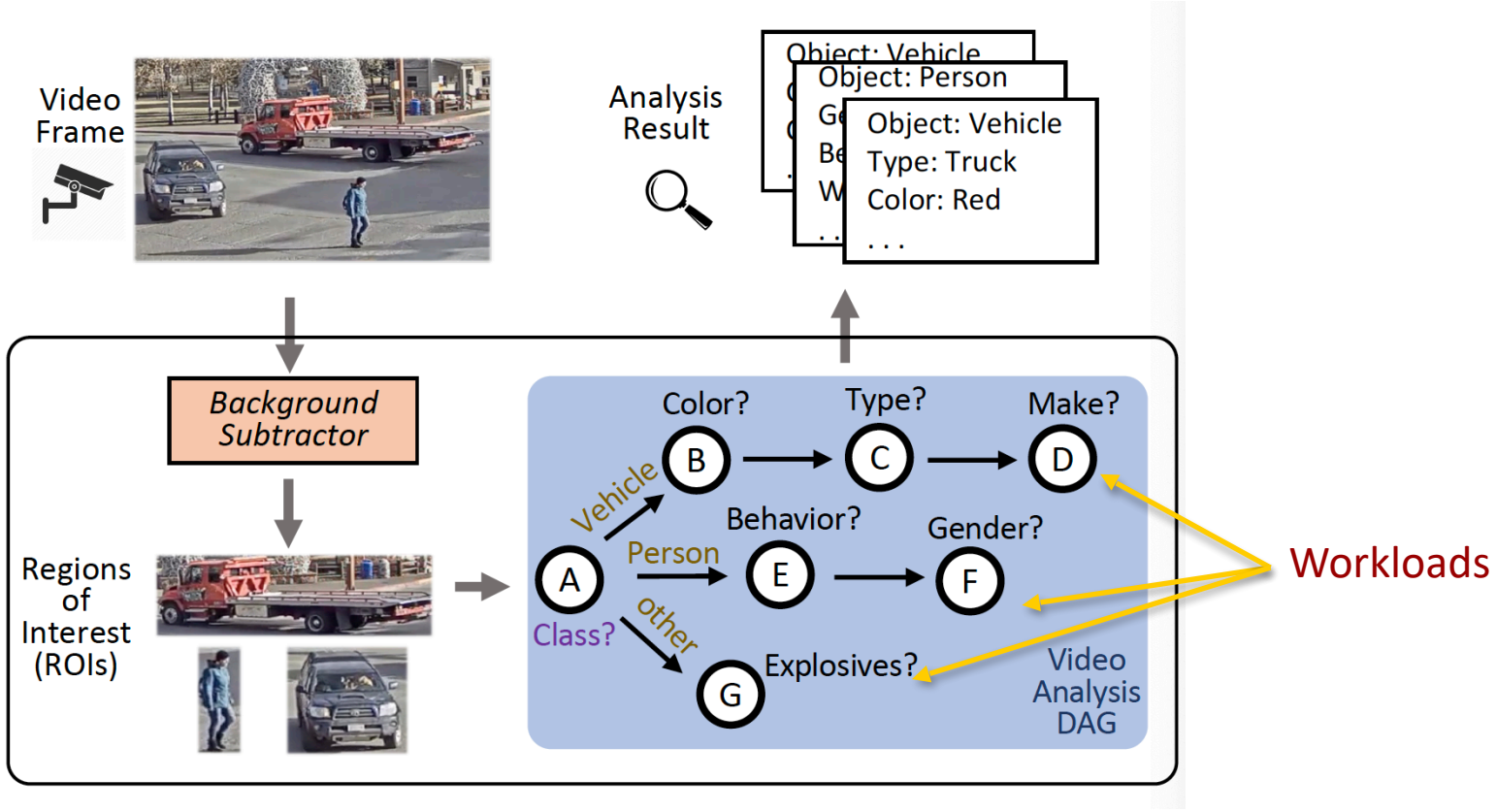
Introduction



We propose Distream, a distributed framework that can leverage on-device intelligences for large-scale video analytics in real-time.

Introduction

Live Video Analytics Pipeline

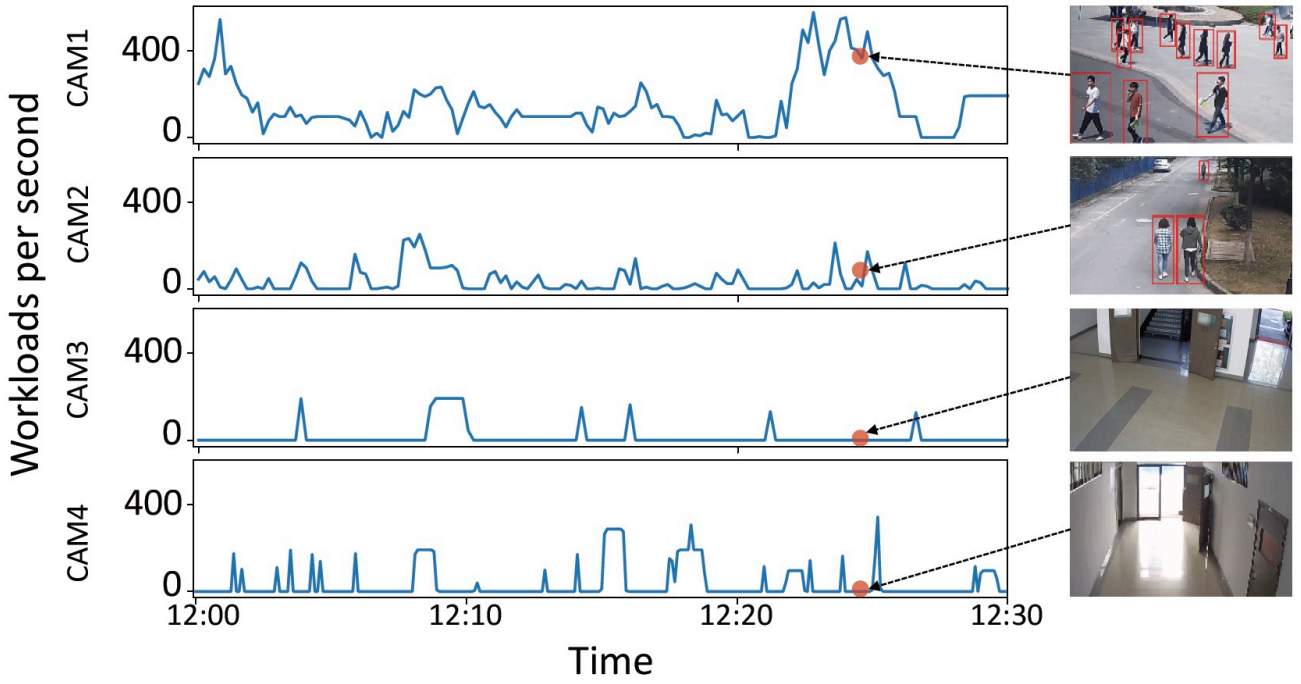


Motivation

Workloads are dynamically changing over time:

I. The workloads are different across cameras

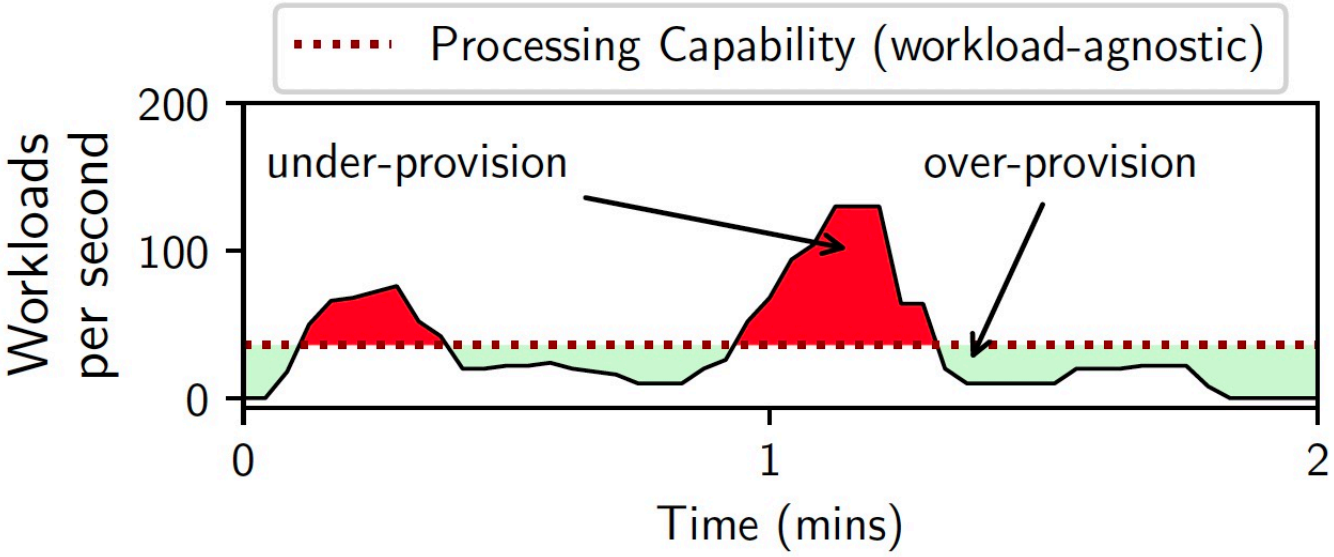
II. The workload generated at each camera is dynamic over time and can vary significantly



What if we ignore these workload dynamics when performing live video analytics?

Motivation

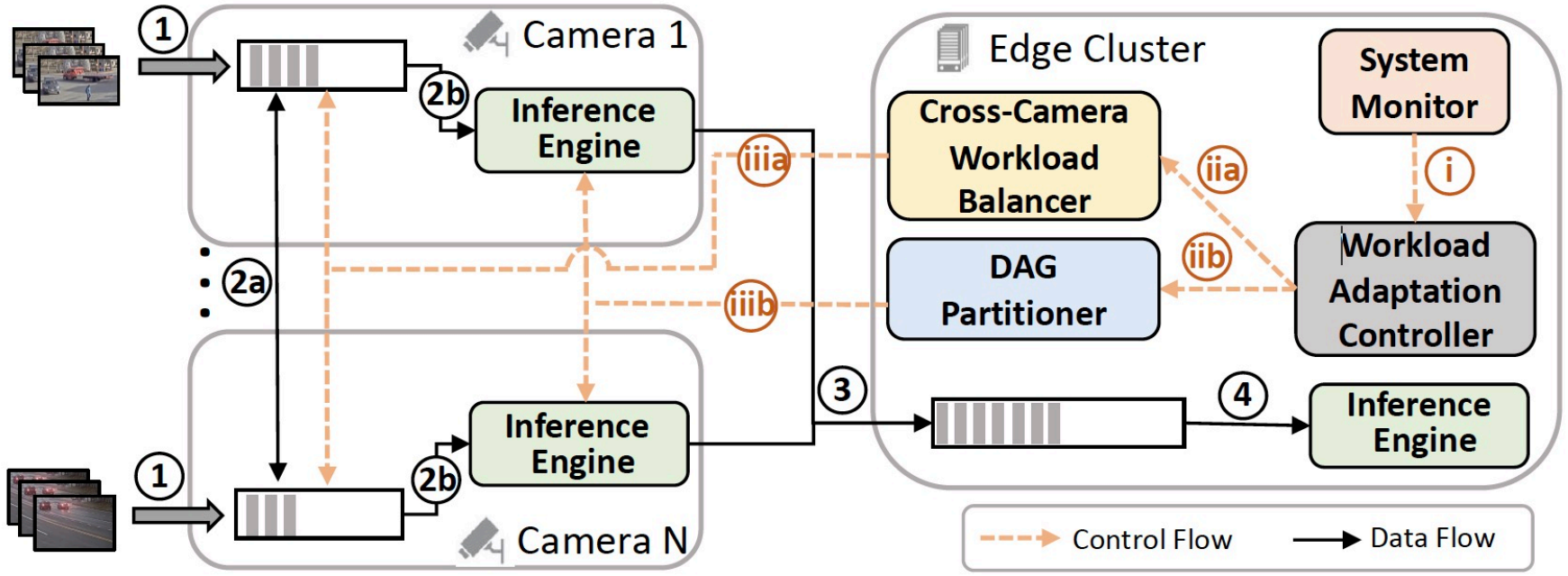
Zoom-in 2-min camera workload video clip



We need a workload-adaptive solution to avoid under-provisioning and over-provisioning

Distream Overview

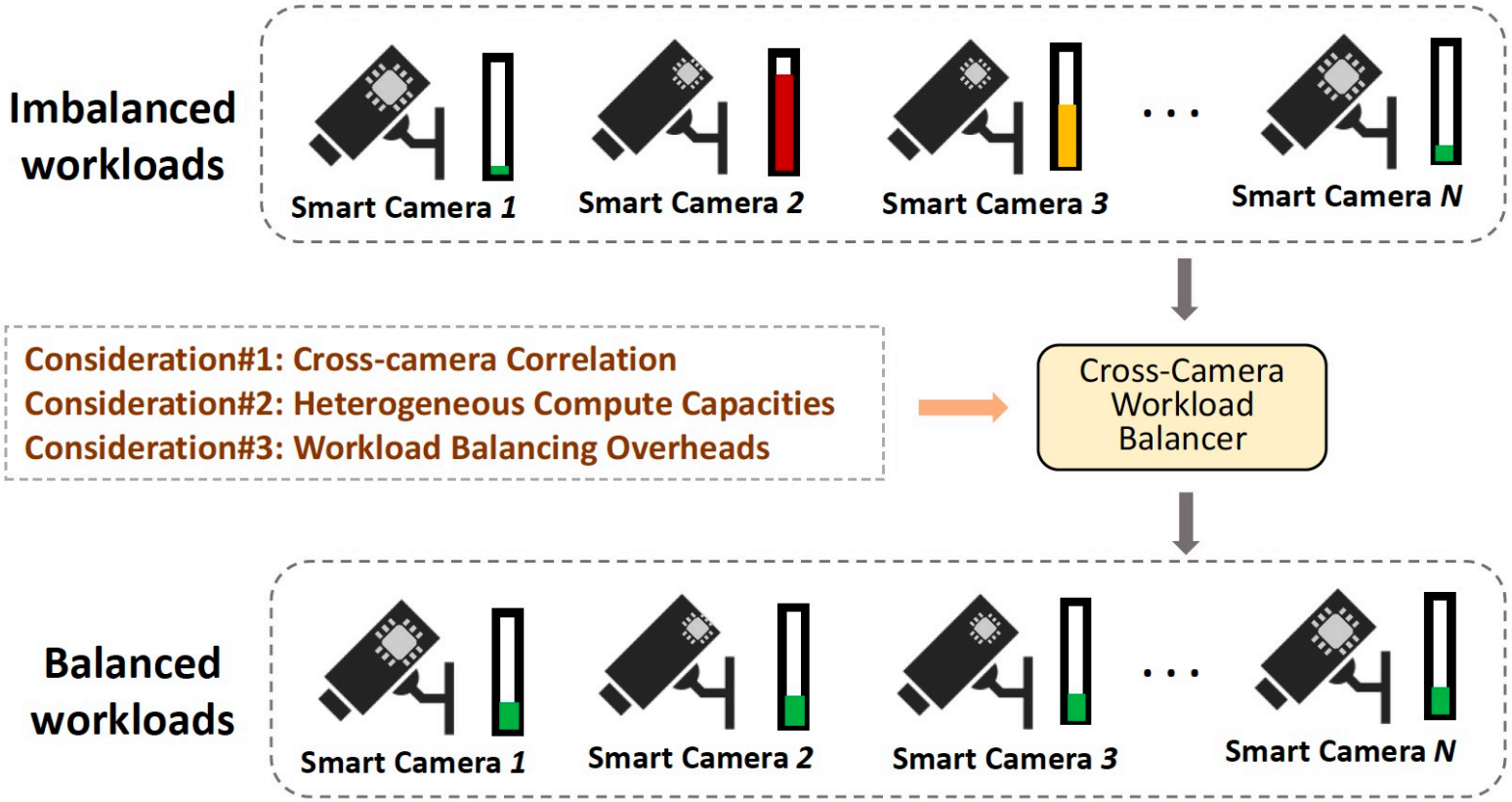
Distream System Architecture



- Cross-Camera Workload Balancer
- Camer-Edge DAG Partitioner
- Workload Adaptation Controller

Distream - Approach

First Component : Cross-Camera Workload Balancer



Distream - Approach

Let x_{ij} represents the workload migrated from camera i to j

We want to know 1) Workloads migrated to camera i $\sum_j x_{ji}$ 2) Workloads migrated from camera j $\sum_j x_{ij}$

Workload imbalance index

#3 Trigger threshold

$$\min_{x_{ij} \geq 0} \quad \max(v' - \beta, 0)$$

$$s.t. \quad u_i = w_i + \sum_j x_{ji} - \sum_j x_{ij} + \hat{w}_i$$

$$a_i = C_i / \bar{C}$$

$$u'_i = u_i * a_i$$

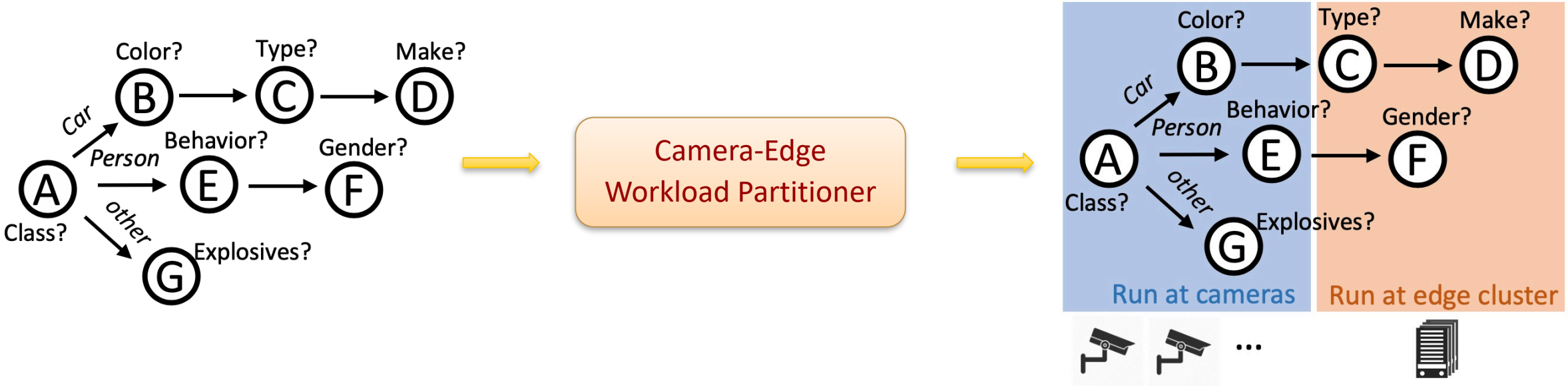
$$v' = \left(\frac{u'_{max}}{\bar{u'}} - 1\right)$$

#1 Predict incoming workload using LSTM

#2 Normalize camera workloads by compute capacities

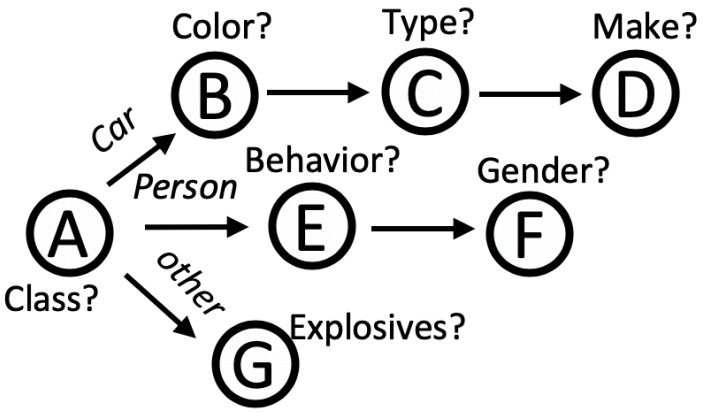
Distream - Approach

Second Component : Camera-Edge Workload Partitioner



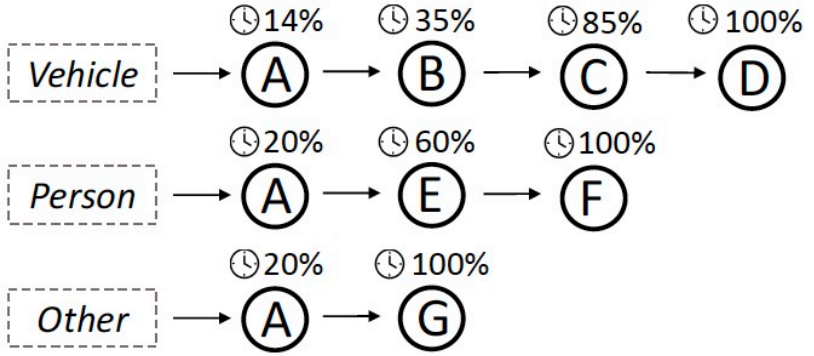
Distream - Approach

Camera-Cluster Workload Partitioner
Step 1/2



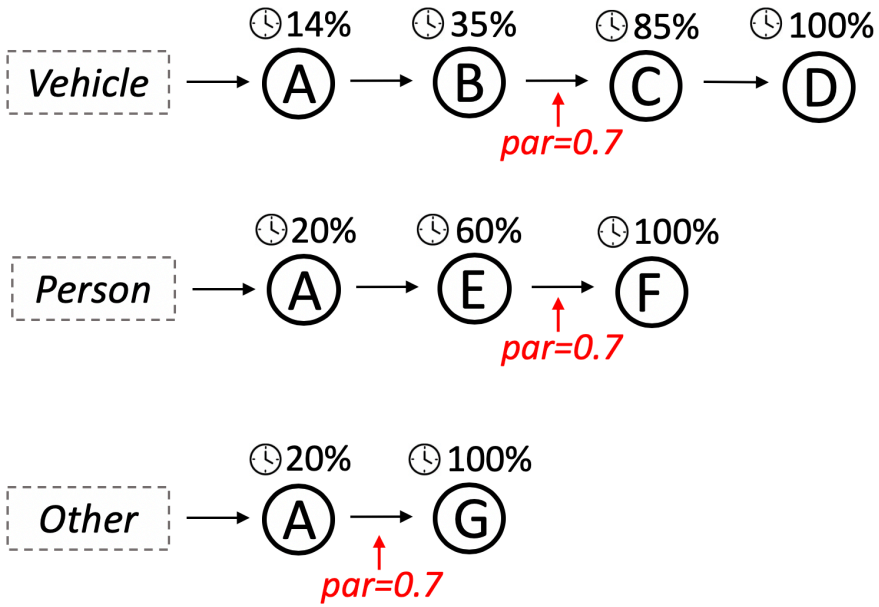
- Extract each possible path
- Profile the accumulated overhead

at each partition point



Distream - Approach

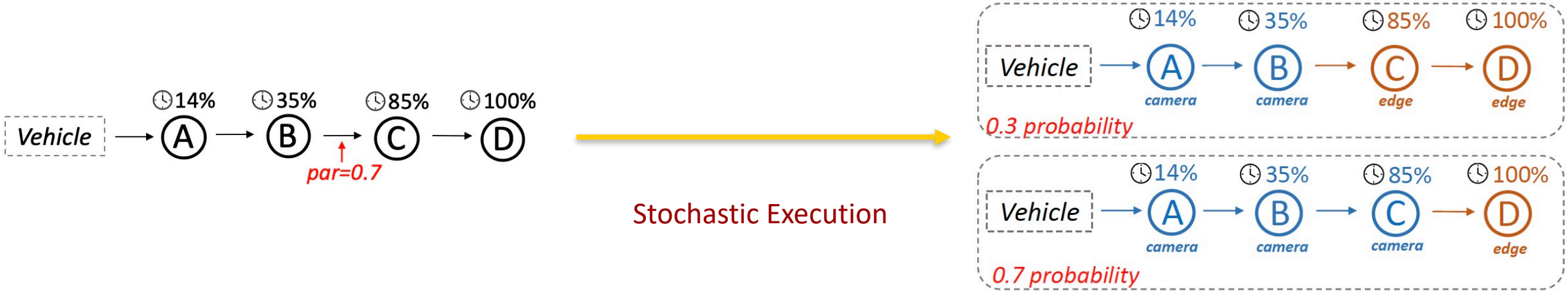
Camera-Cluster Workload Partitioner
Step 1/2



What if we want to achieve par=0.7, i.e.,
70% workload at camera 30% workload at edge, while the corresponding partition point does not exist?

Distream - Approach

Camera-Cluster Workload Partitioner
Step 2/2



Goal:
70% workload at camera
30% workload at edge

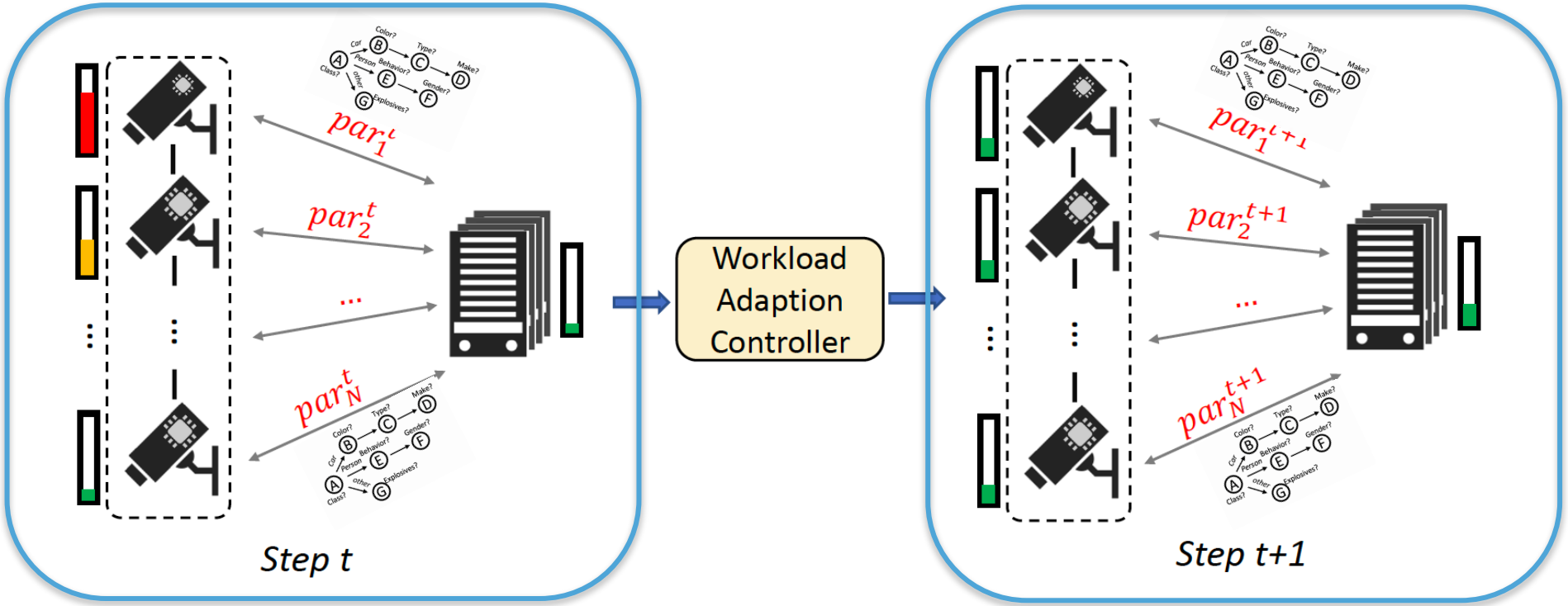
Expected workloads at camera $= 0.3 * 0.35 + 0.7 * 0.85 = 70\%$

Distream - Approach

Third Component : Workload Adaptation Controller

Consideration#1: the optimal DAG partition point is different for each camera

Consideration#2: the optimal DAG partition points of all cameras need to be determined jointly



Distream - Approach

Throughput

$$TP^{cams} = \sum_{i=1}^{N} TP^{cam_i}(par_i) \quad (6a)$$

$$TP^{edge} = TP^{edge}(par_1, par_2, \ldots, par_N) \quad (6b)$$

$$TP^{system} = TP^{cams} + TP^{edge} \quad (6c)$$

Latency

$$L^{cam_i} = \frac{w_i}{TP^{cam_i}} + \frac{1}{TP^{cam_i}} \quad (7a)$$

$$L^{edge} = \frac{w_{edge}}{TP^{edge}} + \frac{1}{TP^{edge}} \quad (7b)$$

Final Optimization

$$\max_{par_i \forall i} \quad TP^{system} \quad (8a)$$

$$s.t. \quad L^{cam_i} \leq L_{Max}, \forall i \quad (8b)$$

$$L^{edge} \leq L_{Max} \quad (8c)$$

Maximize system throughput

Under latency constraint

Distream - Approach

Approach Summary:

1) Distream uses cross-camera workload balancer to balance the workloads among cameras

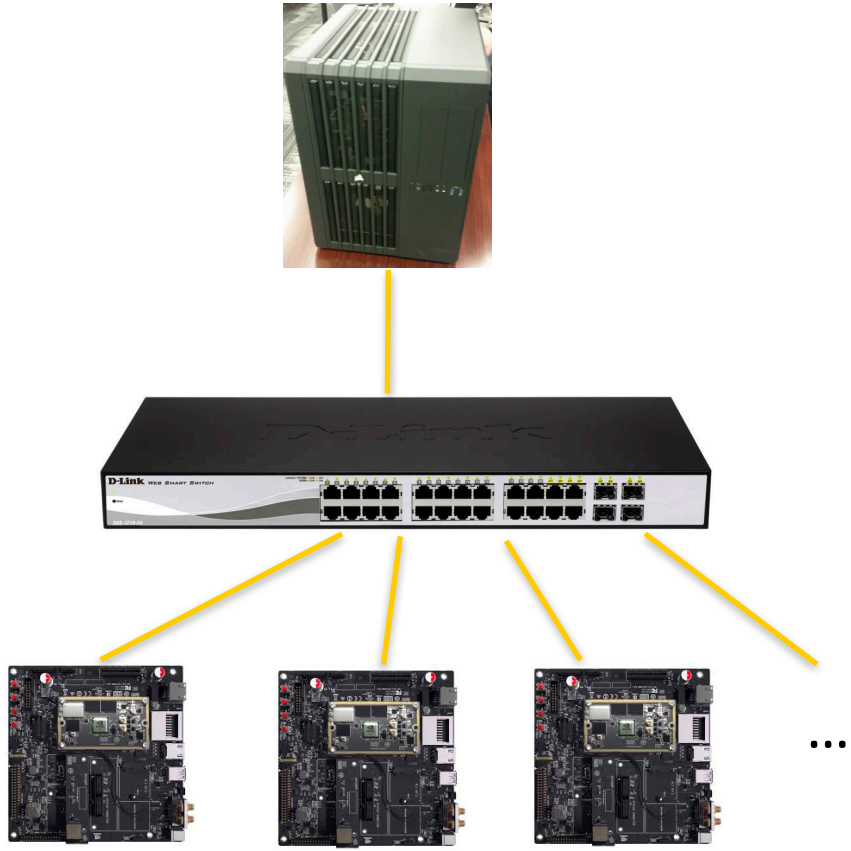
2) Distream uses camera-cluster workload partitioner to partition the workloads between cameras and edge cluster.

3) Distream uses workload adaptation controller to trigger workload balancing and dynamically determine the partition points between cameras and edge cluster.

Experiment

Testbed

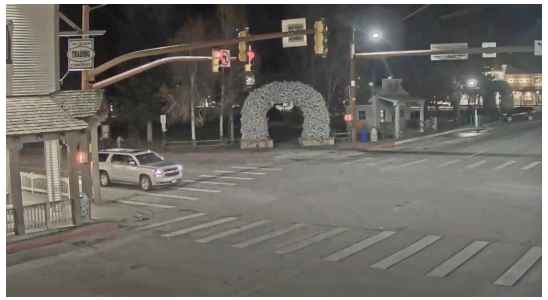
- The testbed consists of 24 smart cameras and a single edge cluster. Among the 24 cameras, 18 were prototyped using Nvidia Jetson TX1 while the other six were prototyped using Nvidia Jetson TX2. All smart cameras has a powerful onboard mobile GPU.
- For the edge cluster, we use a desktop server equipped with a cluster of 4 Nvidia Titan X GPUs.
- The 24 smart cameras and the edge cluster are connected via a single switch (D-Link DGS-1510-28X) to form a local network (50 Mbps)



Experiment

Datasets:

Two Applications using two large-scale two datasets.



1) Traffic Monitoring
   - 200 5-minute video clips across 48 hours of video
   - Video streams from each of the six cameras at a frame rate of 15 FPS
   - 1280x720P frame resolution

2) Campus Surveillance
   - 200 5-minute video clips across 48 hours of video
   - 24 cameras at a frame rate of 15 FPS
   - 1280x720P frame resolution

Experiment

Baselines:

- VideoEdge-L: A state-of-the-art distributed live video analytics system that is workload-agnostic.
- Centralized: All workloads are processed at edge cluster
- Camera-Only: All workload are processed at cameras.

Metric:

Throughput: Inference Per Second (IPS)

Latency: Time elapsed from the workload is produced to the workload is completed.

Latency Service Level Objective (SLO) Miss Rate: quantifies the percent of theworkloads that does not meet the latency requirement set by a live video analytics application

Experiment – Overall Performance

System Performance Comparison (speedup over Camera-Only)

Campus Surveillance Dataset:

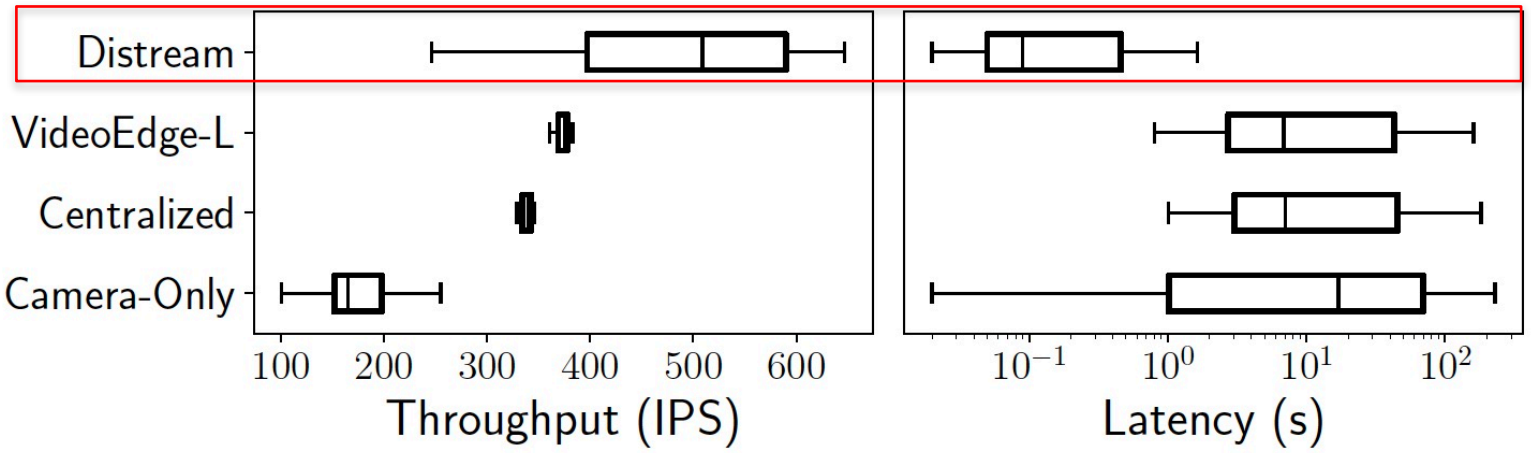
	Throughput				Latency			
	avg	med	75th	99th	avg	med	75th	99th
Distream	**2.9×**	**3.1×**	**2.9×**	**2.5×**	**128.2×**	**189.3×**	**147.9×**	**112×**
VideoEdge-L	2.1×	2.2×	1.9×	1.5×	1.5×	2.4×	1.6×	1.4×
Centralized	1.9×	2.0×	1.7×	1.3×	1.4×	2.2×	1.5×	1.2×

Traffic Monitoring Dataset:

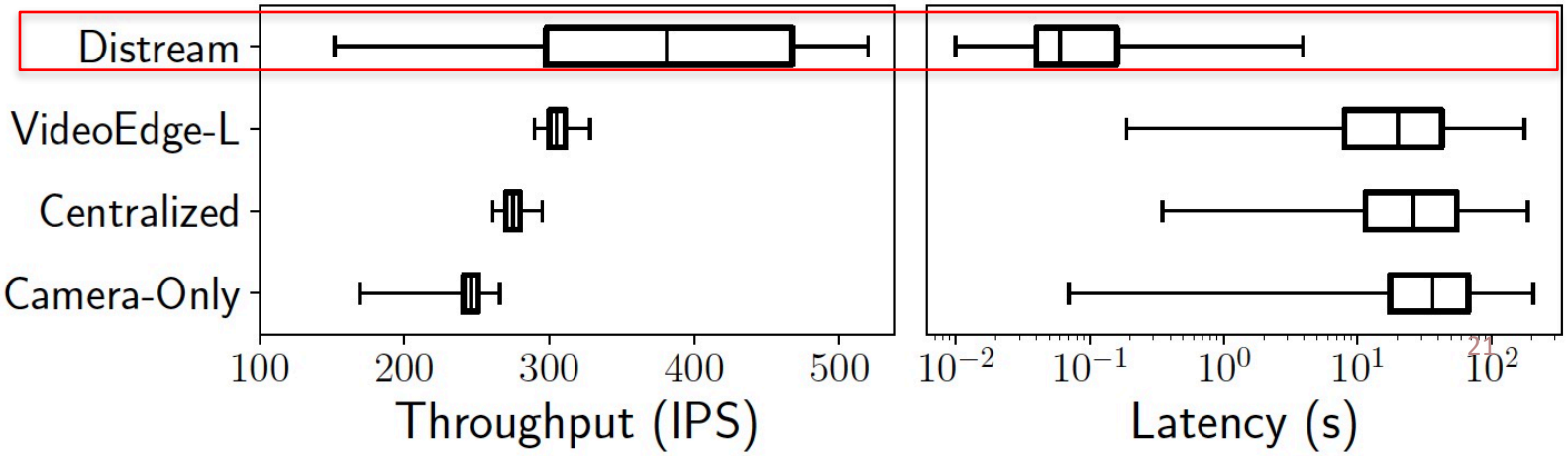
	Throughput				Latency			
	avg	med	75th	99th	avg	med	75th	99th
Distream	**1.6×**	**1.6×**	**1.9×**	**2.0×**	**184×**	**604.3×**	**446.8×**	**52.6×**
VideoEdge-L	1.2×	1.2×	1.2×	1.2×	1.5×	1.8×	1.5×	1.2×
Centralized	1.1×	1.1×	1.1×	1.1×	1.2×	1.4×	1.2×	1.1×

Experiment – Overall Performance

Campus Surveillance Dataset:

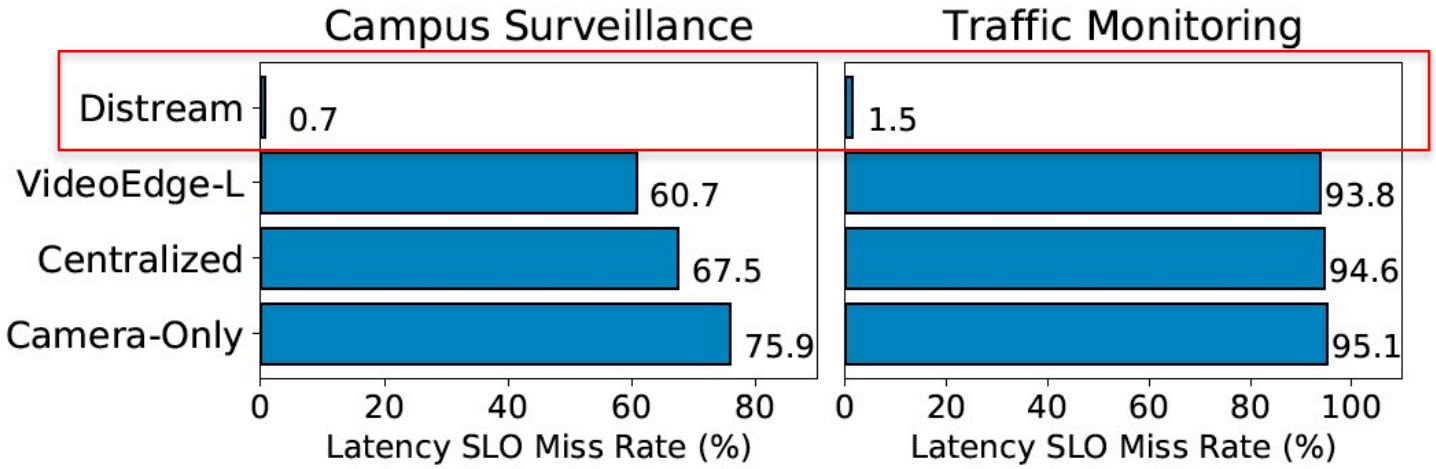


Traffic Monitoring Dataset:



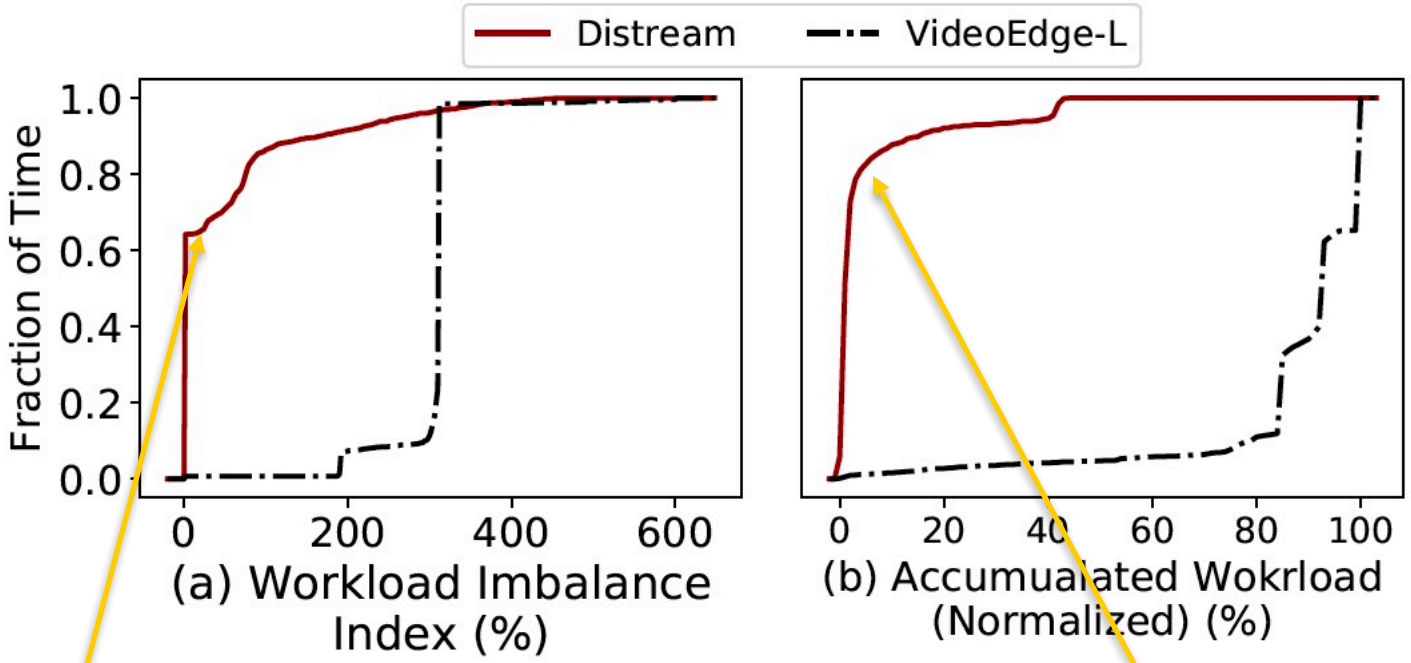
Experiment – Overall Performance

Latency SLO Miss Rate Comparison
(Latency requirement: 3s)



Experiment - Analysis

Why does Distream perform significantly better?



Distream has less workload imbalance, which indicates higher compute resource utilization.

Distream suffers less from workload overload, which indicates it can better handle workload burst.

Experiment – Sensitivity Analysis

How does Distream perform under low and high bandwidths?

	Throughput (IPS)				Latency (s)			
	avg	med	75th	99th	avg	med	75th	99th
Low Bandwidth (10Mbps)	489.1	509	590	647	0.34	0.09	0.47	2.05
High Bandwidth (50Mbps)	490.9	510	593	650	0.31	0.07	0.41	1.75

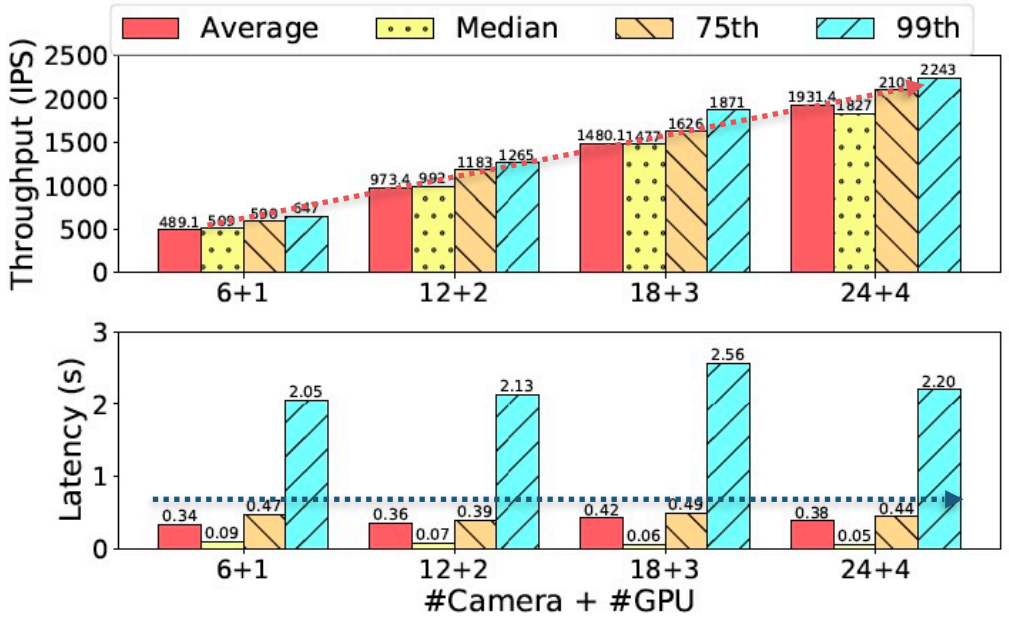
Distream achieves similar performance even under low bandwidth (10Mbps).

Experiment – Scalability

How does Distream perform when the system is scaled up to more cameras?

Throughput increases linearly.

Latency remains the same.



Conclusion

- We present Distream, a workload-adaptive live video analytics system based on the smart camera-edge cluster distributed architecture.
- Distream addresses the key challenges brought by workload dynamics in real-world deployments.
- Experiments on two real-world video datasets showed that Distream consistently outperforms the status quo in terms of throughput, latency, and latency SLO miss rate.

Thank you!

Q & A