INT Based Network-Aware Task Scheduling for Edge Computing

Bibek Shrestha
University of Nevada, Reno

Richard Cziva
Lawrence Berkeley National Laboratory

Engin Arslan
University of Nevada, Reno

Edge Computing

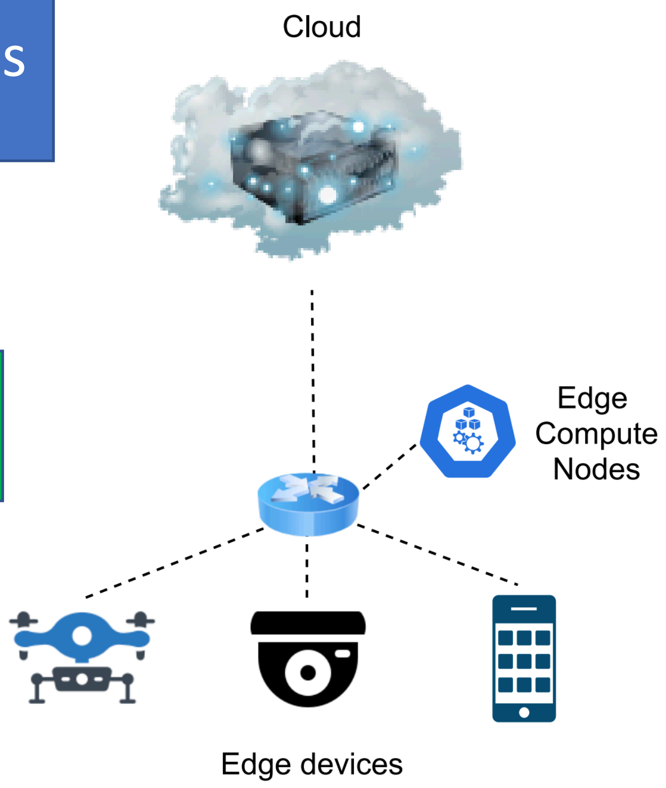
Azure reported max latency **~400ms** between different regions

*Azure network round-trip latency statistics, 2020

Latency in the closer regions < 50ms

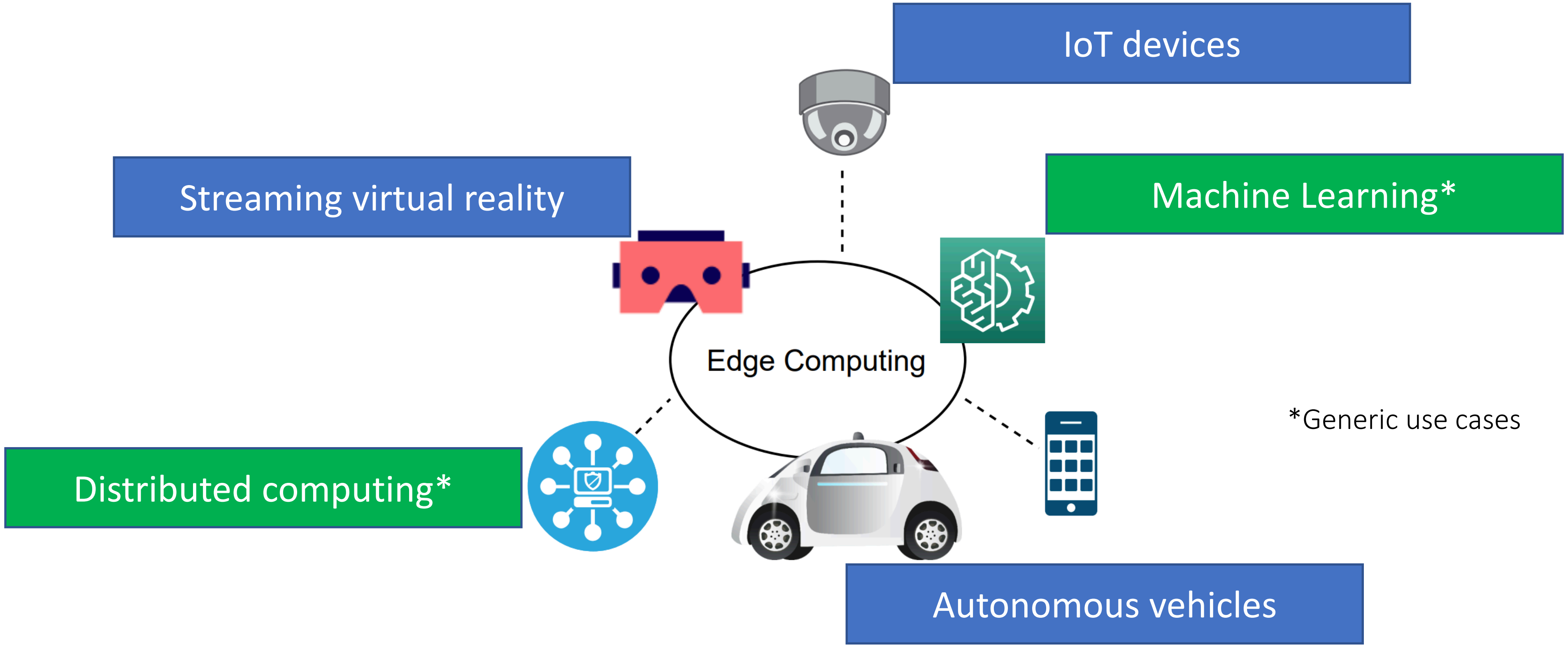
Latency largely reduced when regions are closer

Edge computation brings data and computation closer to the source → latency further reduced



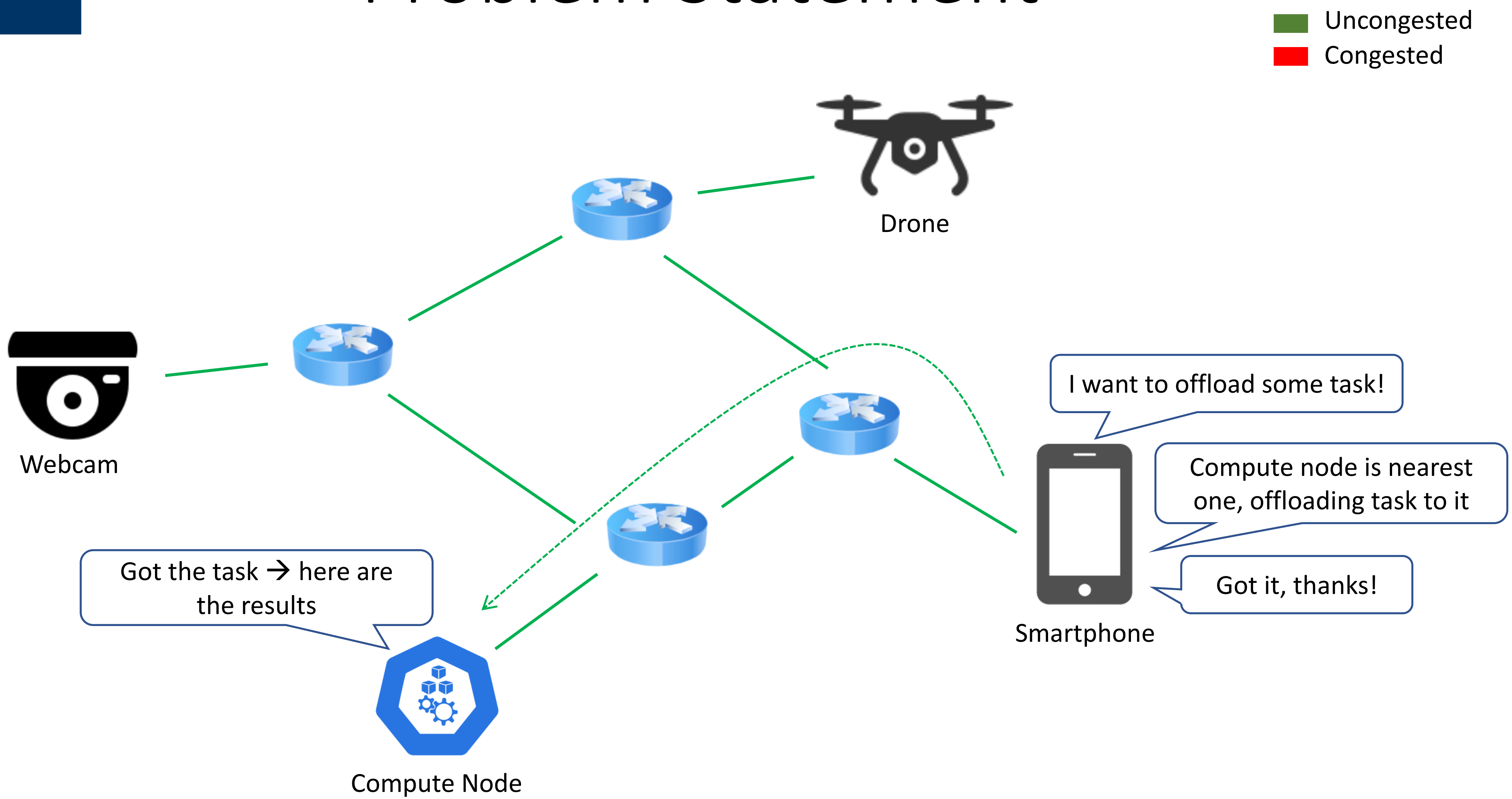
Edge Computing

- Applications with low latency requirements benefits from edge computing

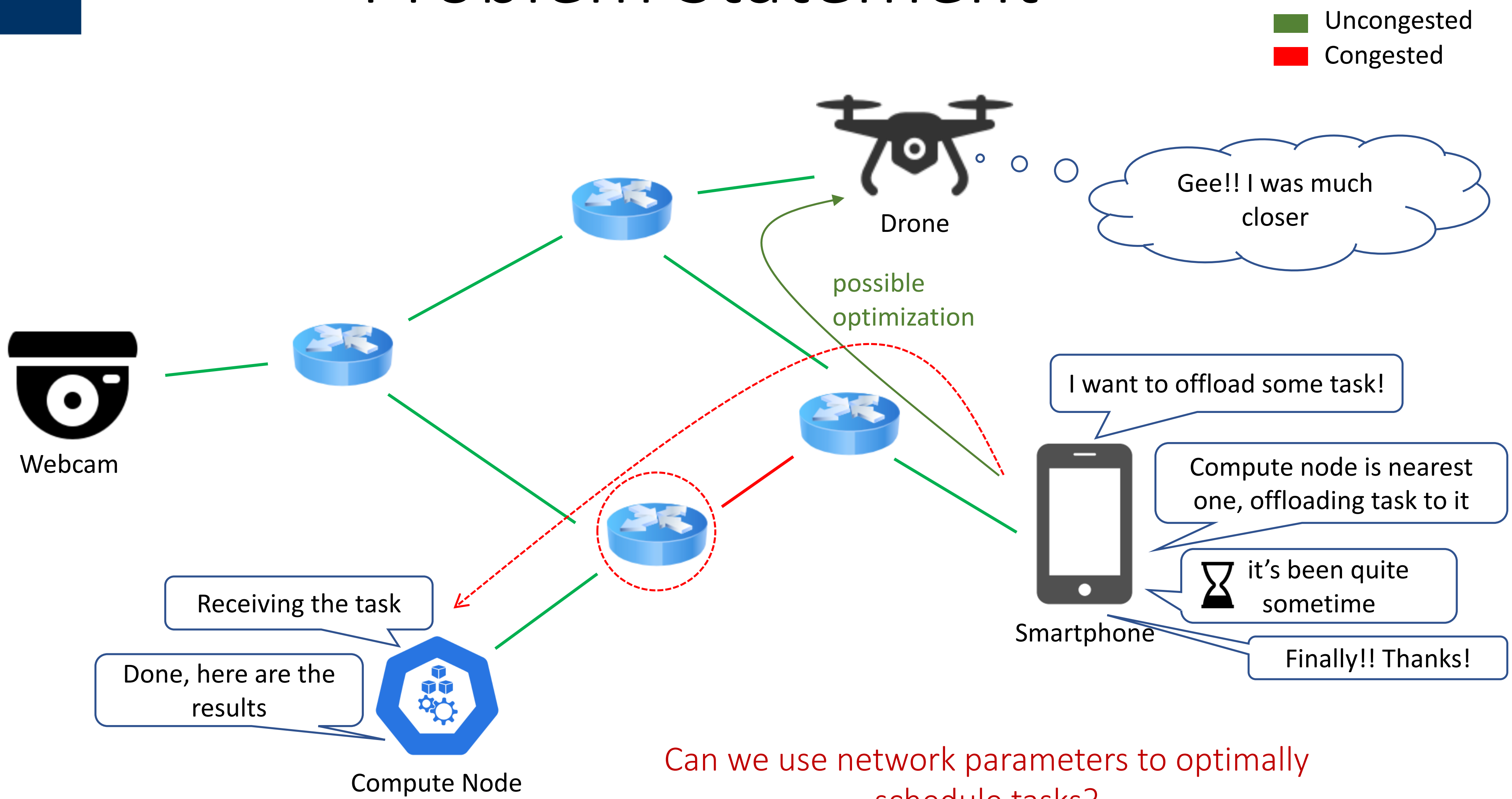


*Generic use cases

Problem Statement



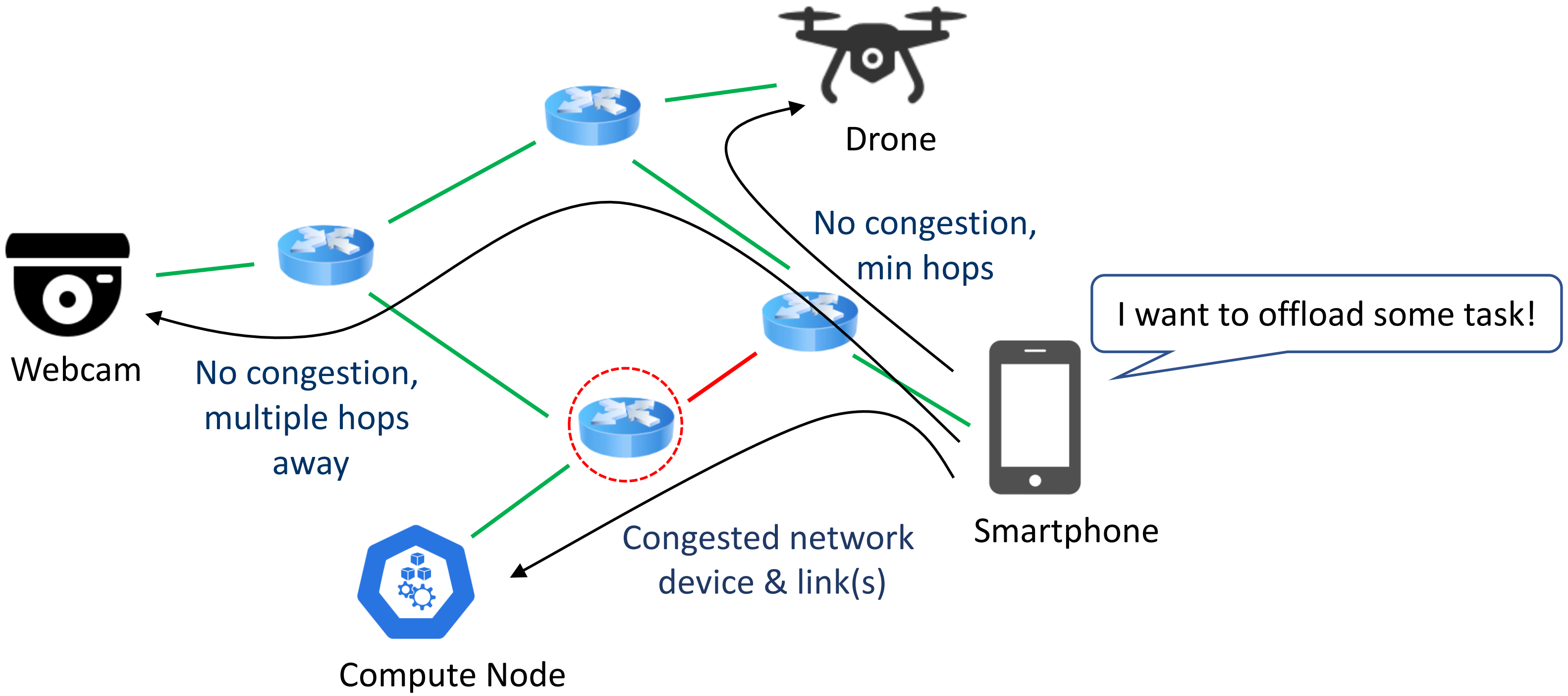
Problem Statement



Can we use network parameters to optimally schedule tasks?

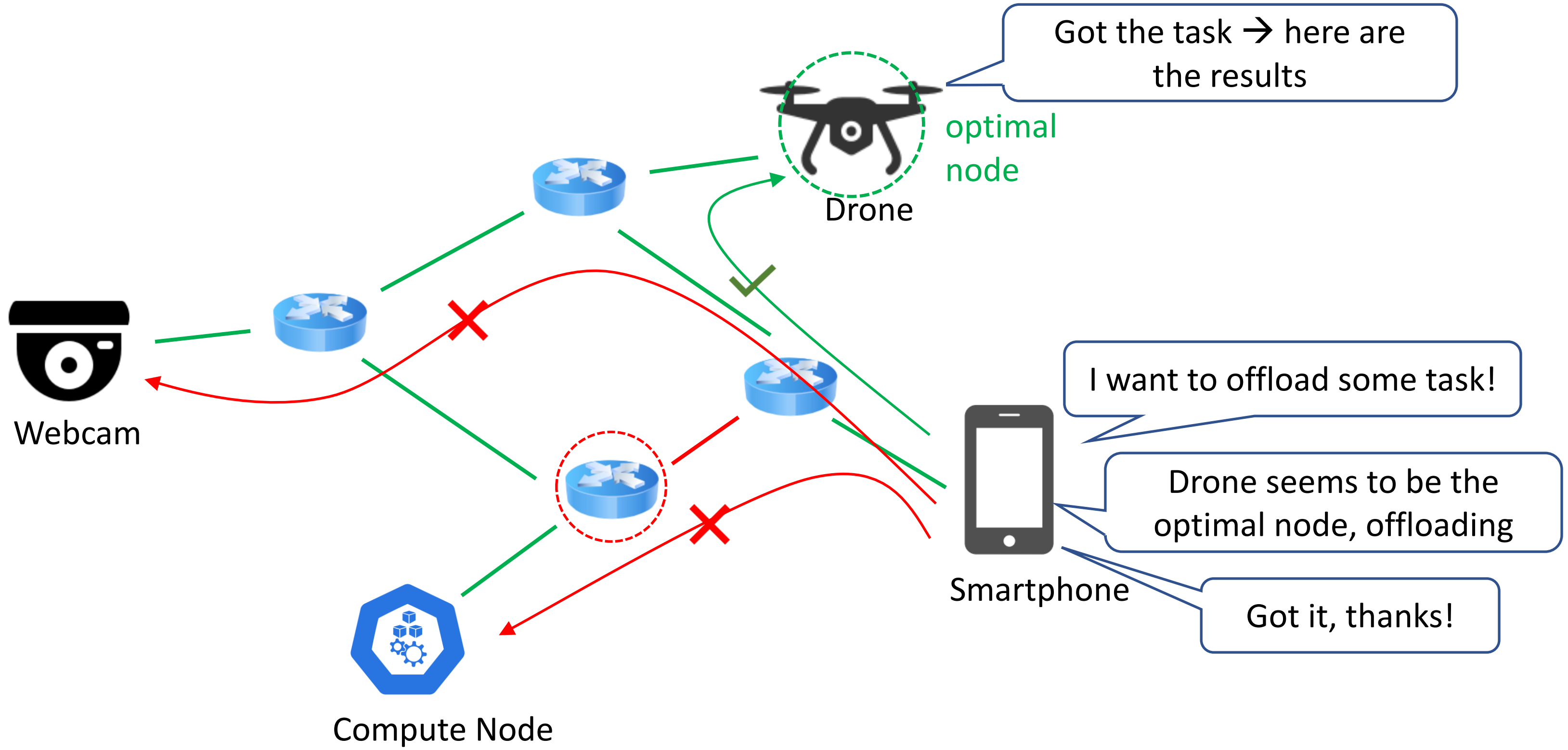
Proposed Solution

- Network-aware task scheduling
  - Consider network conditions to make optimal scheduling decisions



Proposed Solution

- Network-aware task scheduling
  - Consider network conditions to make optimal scheduling decisions



Proposed Solution

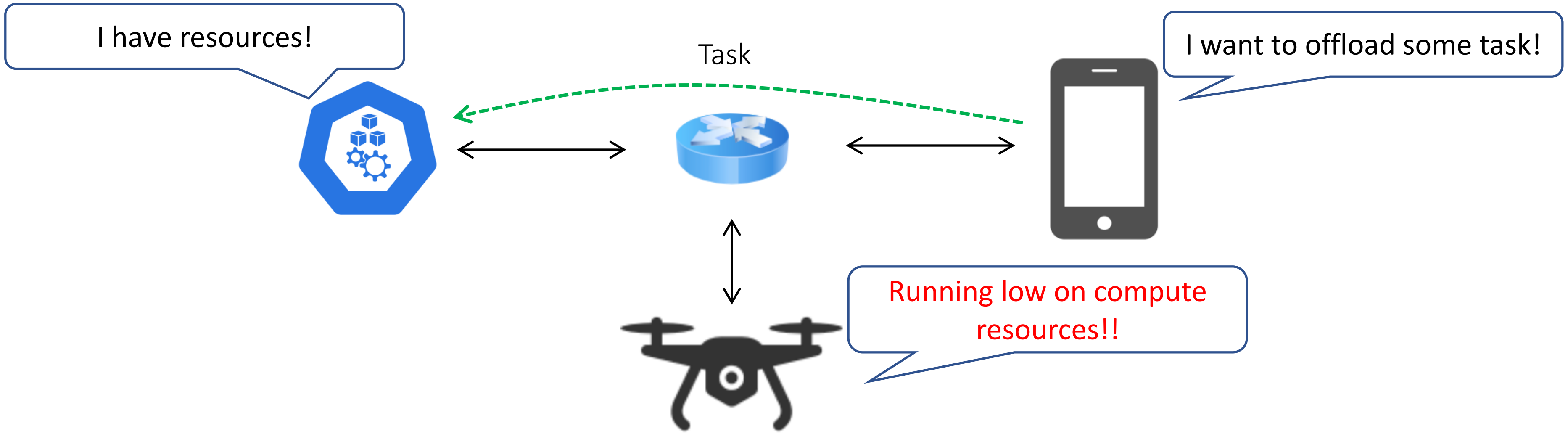
- Traditional methods of network monitoring are inadequate
  - SNMP, NetFlow
- Lower sampling rate → reduced network visibility → reduced capacity to make optimal decisions

Proposed Solution

- Traditional methods of network monitoring are inadequate
  - SNMP, NetFlow
- Lower sampling rate → reduced network visibility → reduced capacity to make optimal decisions
- Programmable data plane & In-band Network Telemetry (INT) to the rescue

Related Work

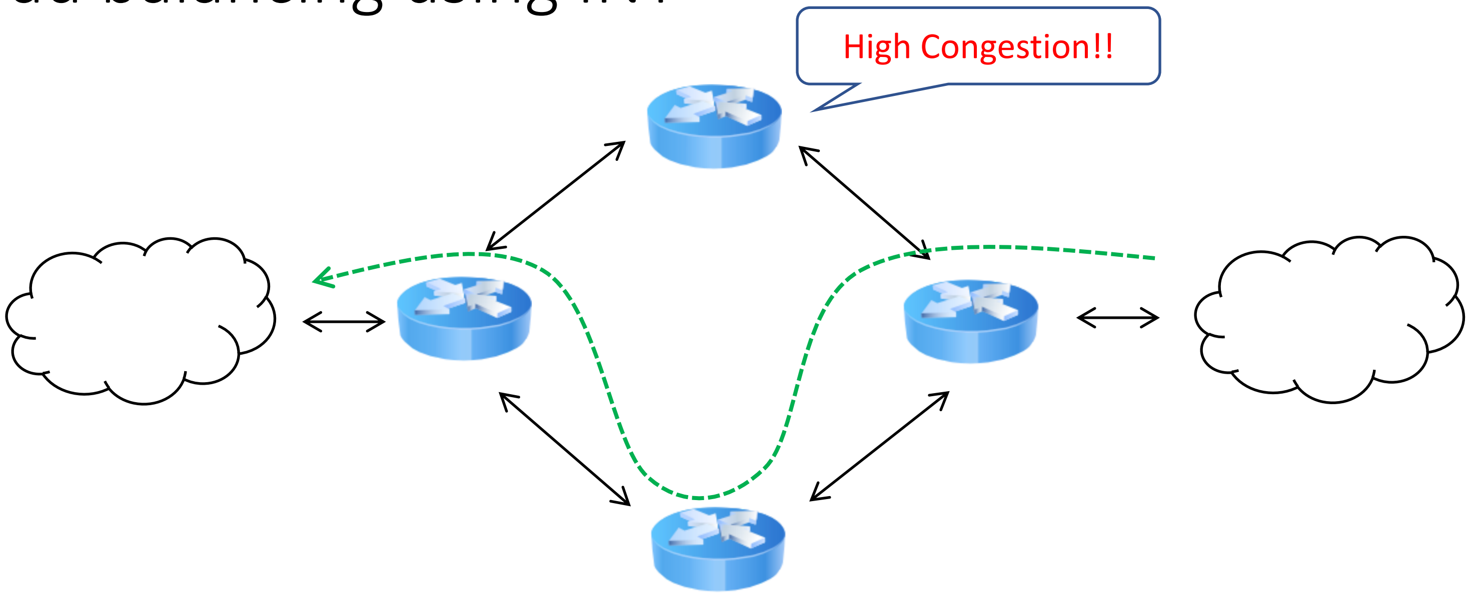
- Compute resources availability of edge node influences performance



- Cache-awareness can significantly improve the task completion time

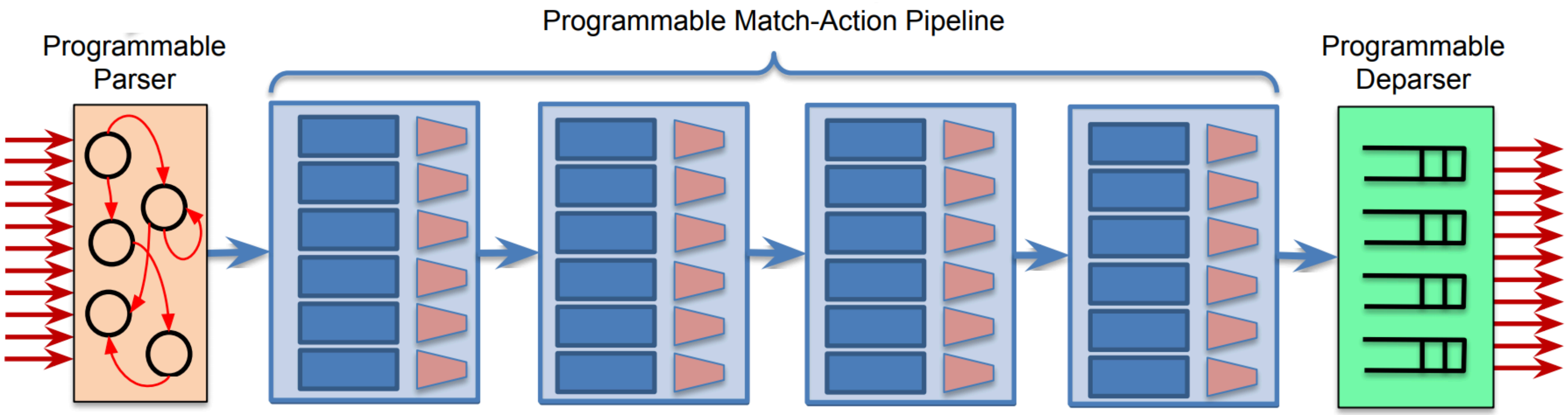
Related Work

- Production jobs are usually recurring with predictable characteristics
  - Planning the data and job placement → enhances job locality → enhances performance
- Network load balancing using INT



Programmable data plane

- Custom packet processing routine directly at data plane → line rate

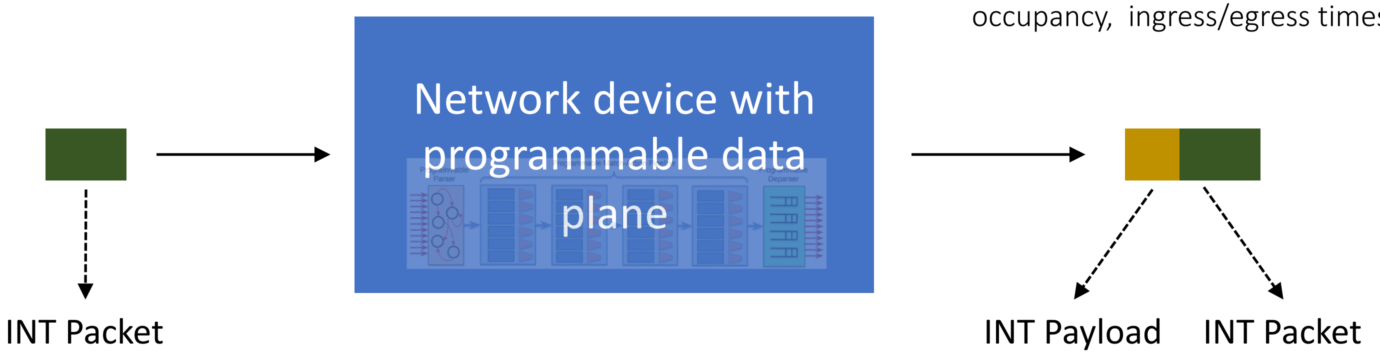


Protocol-Independent Switch Architecture (PISA)

In-band Network Telemetry (INT)

- Framework for collection and reporting of network data by data plane
- No intervention/work from control plane

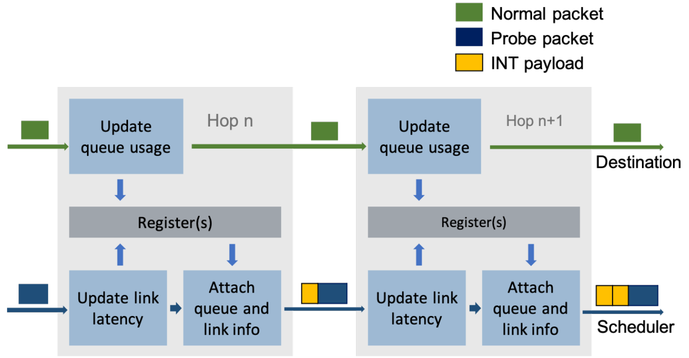
*INT Payload: switch id, ingress/egress queue occupancy, ingress/egress timestamps

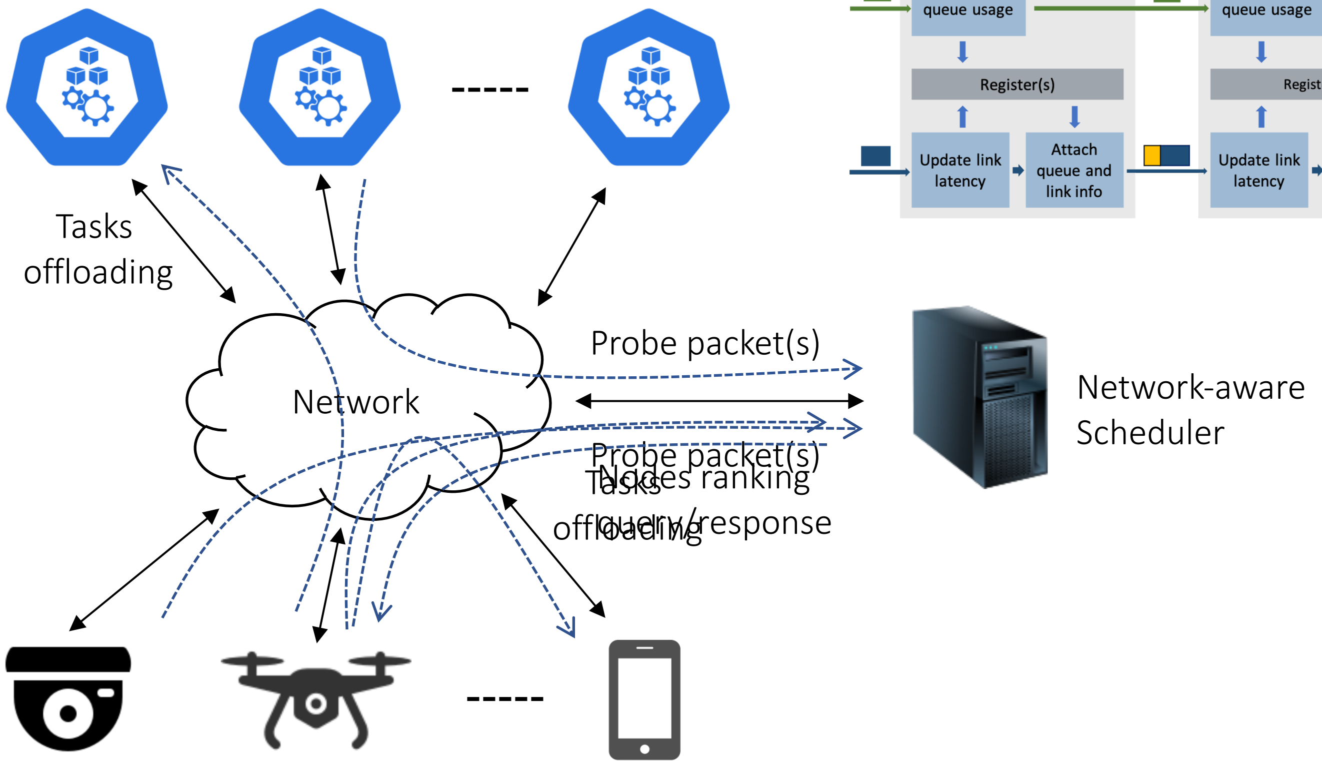


- Access to fine granular network telemetry at line rate → increased network visibility → increased ability to detect network changes

Network-Aware Task Scheduler

1. INT collection
2. Network mapping
3. Nodes ranking query
4. Task offloading



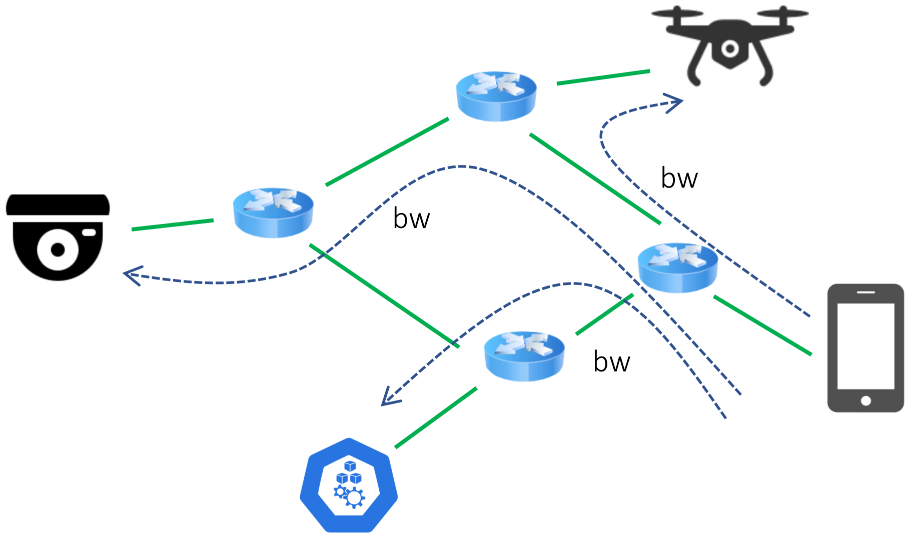


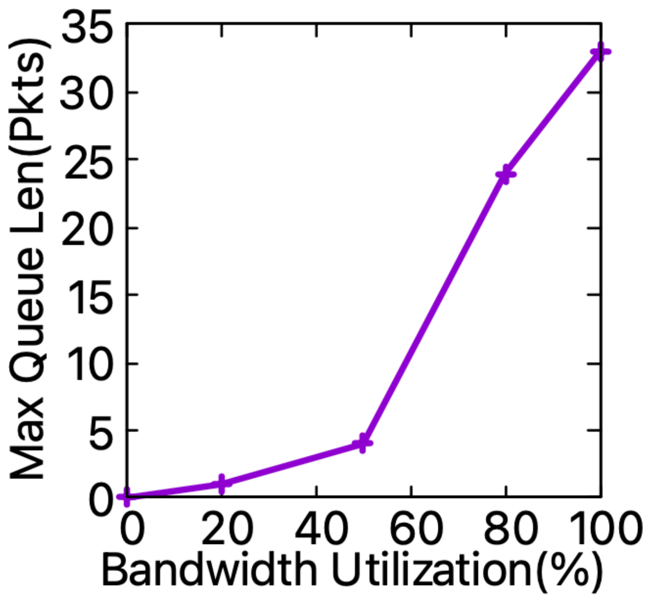
Ranking Algorithm

- Ranking algorithms uses available network capacity for ranking
- Two node ranking algorithms proposed
  - Bandwidth-based node ranking
  - Delay-based node ranking

Bandwidth-based node ranking

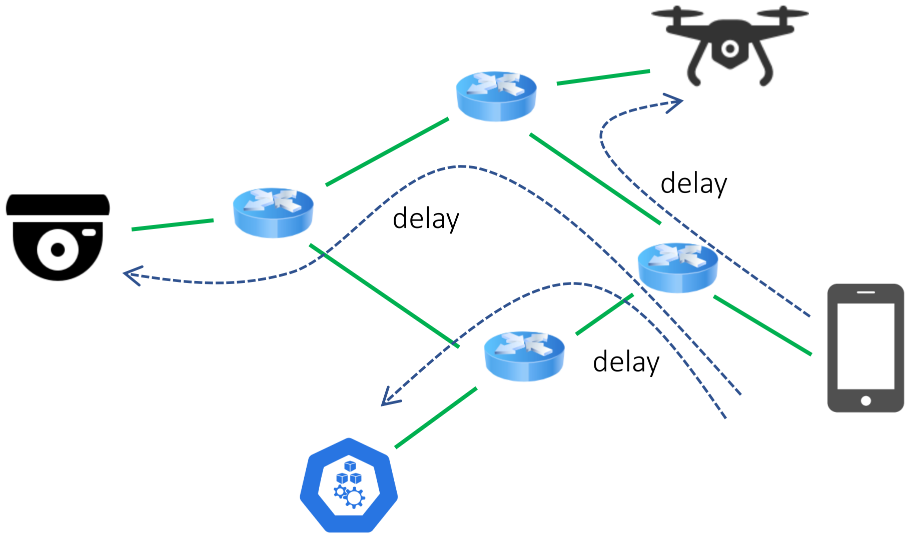
- Sort the available nodes based on the bandwidth availability of each node from the querying node

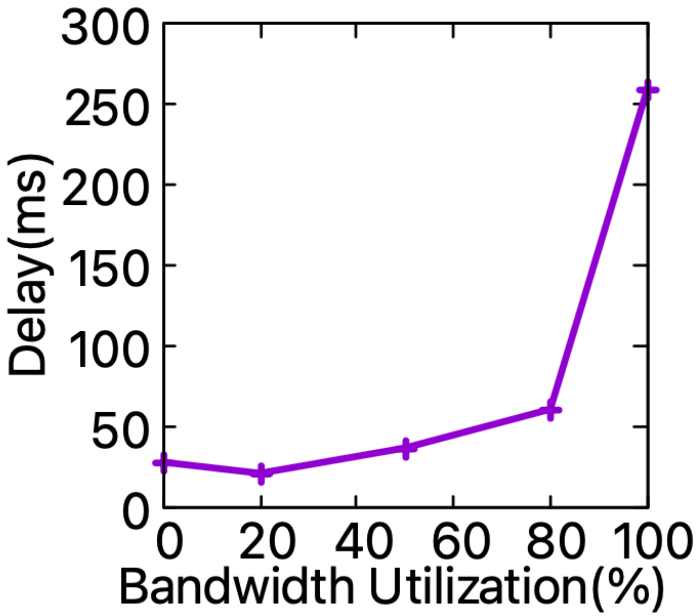




Delay-based node ranking

- Sort the available nodes based on the delay of each node from querying node

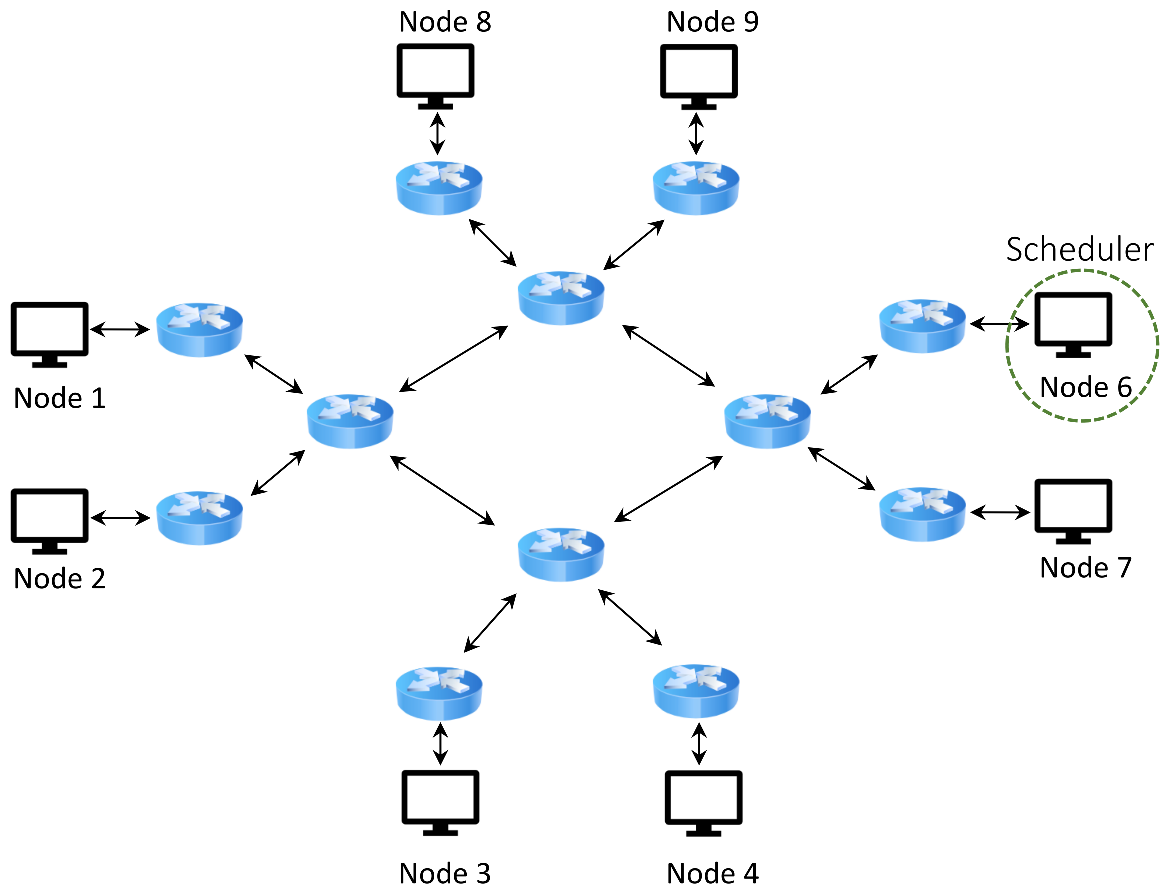




$$Delay(e_n, e_m) = \sum_{i=1}^{k} delay(l_i) + \sum_{i=1}^{k} delay(h_i)$$

Experiment Setup

- Mininet (distributed)
- Behavioral Model (BMv2) switch
- P4 programming language
- 4 x servers: 4 core CPU, 32GB RAM running Ubuntu 18.04
- HP Procurve switches to provide physical connectivity

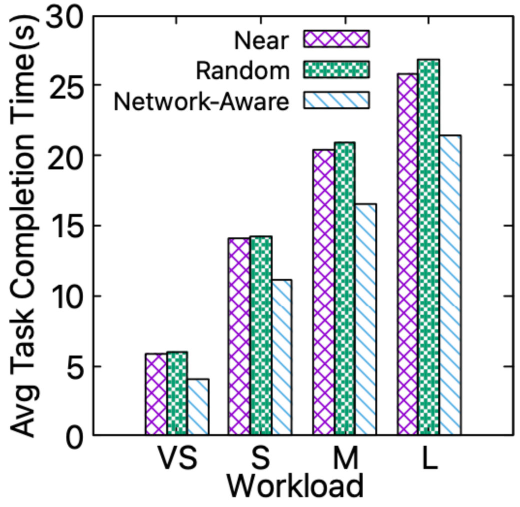


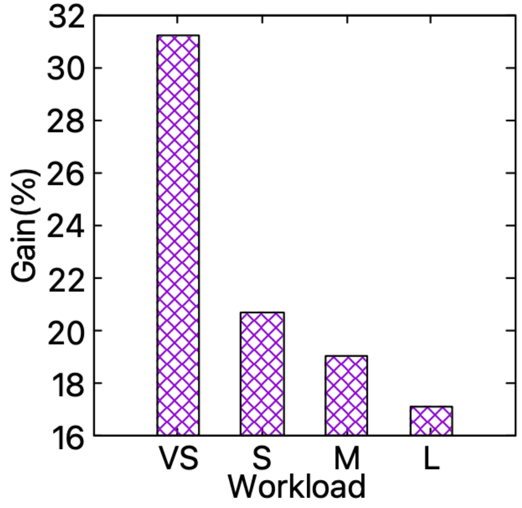
Experiment Setup

- Node selection methods in comparison
  - Physically near node selection
  - Random node selection
  - Network-Aware node selection (ranking method)

Results (Delay-based ranking)

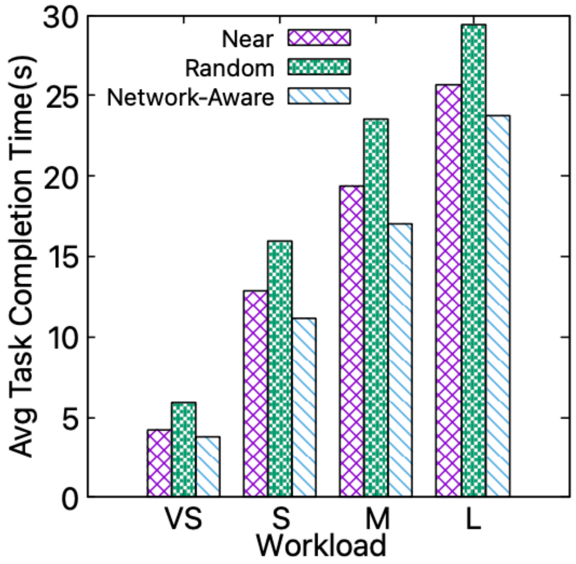
- Average task completion time on various workload sizes for *serverless computing workload*
- Avg task completion time reduced by **~31%** compared against near selection strategy for very small workloads

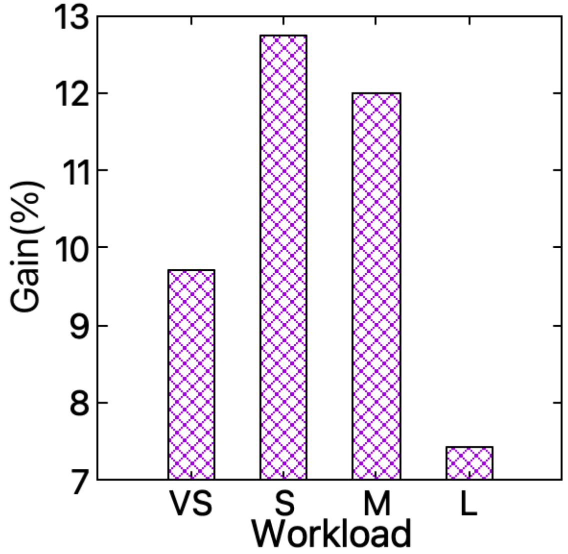




Results (Delay-based ranking)

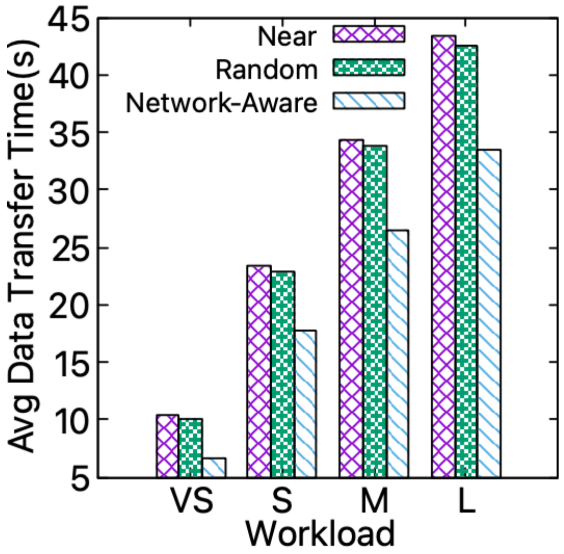
- Average task completion time on various workload sizes for *distributed computing workload*
- Avg task completion time reduced by **~13%** compared against near selection strategy for small workloads

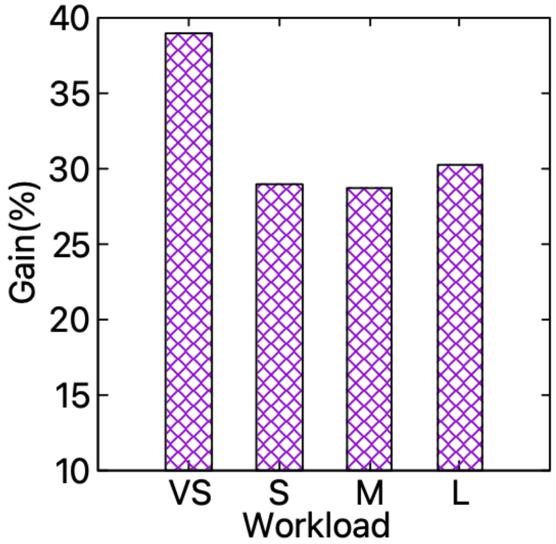




Results (Bandwidth-based ranking)

- Average data transfer time on various workload sizes for *distributed computing workload*
- Avg data transfer time reduced by **~40%** compared against near selection strategy for very small workloads



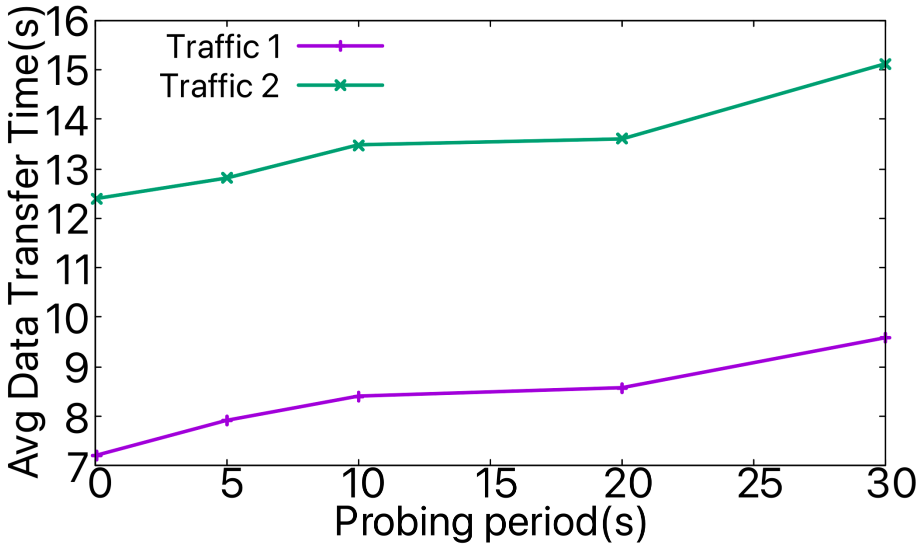


Impact of Probing frequency

- Determine the impact of the probing frequency on the results
- Experiment
  - Distributed workload
  - delay-based ranking strategy
  - Varying probing period [0.1s-30s]
  - Variable traffic scenario (frequent, infrequent changes, workload size)

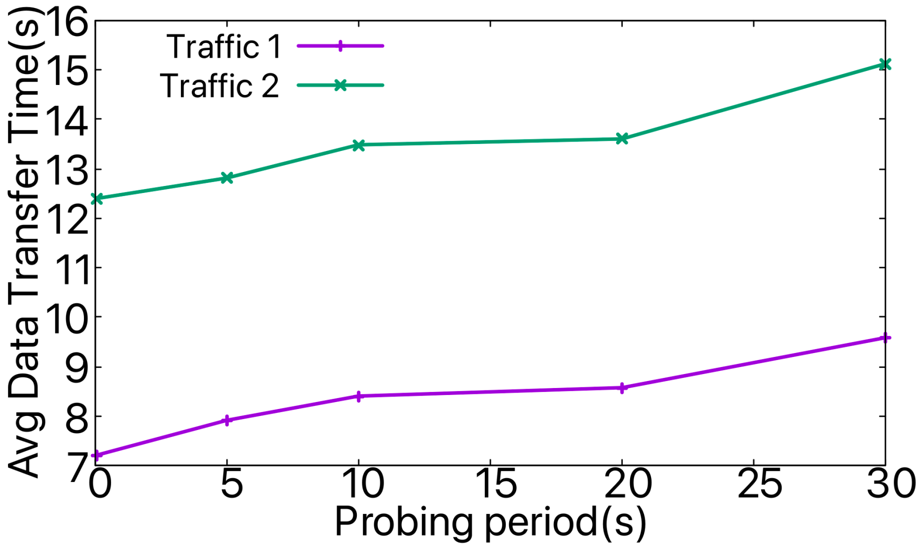
Impact of Probing frequency

- Lower probing period → Lower average data transfer time



Impact of Probing frequency

- Lower probing period → Lower average data transfer time
- Likelihood of capturing subtle changes in the network increased



Conclusion

- High precision telemetry received with INT at higher rate provides better picture of network → detect network congestion events
- Proposed two strategies to rank the available nodes based on the network state to implement network-aware task scheduling
- network-aware task scheduling
  - Up to 40% reduction in average data transfer time
  - Up to 30% reduction in average task completion time

Future Work

- Combine network-awareness and compute-awareness
- Improve delay and bandwidth usage inference with machine learning
- Heterogenous computing scenario where tasks might have requirements such as GPGPU
- Store information at each node → eliminate dependency on central controller for scheduling

Thank You!

Bibek Shrestha
bibek.shrestha@nevada.unr.edu
University of Nevada, Reno

Richard Cziva
richard@es.net
Lawrence Berkeley National Laboratory

Engin Arslan
earslan@unr.edu
University of Nevada, Reno

BERKELEY LAB