Z6110X0035:
Introduction to Cloud Computing
- Cloud–Edge Computing

Lecturer: Prof. Zichen Xu

Acknowledge Dr. Weisong Shi, ApproxIoT, and Tsinghua University

Recap from Previous Class

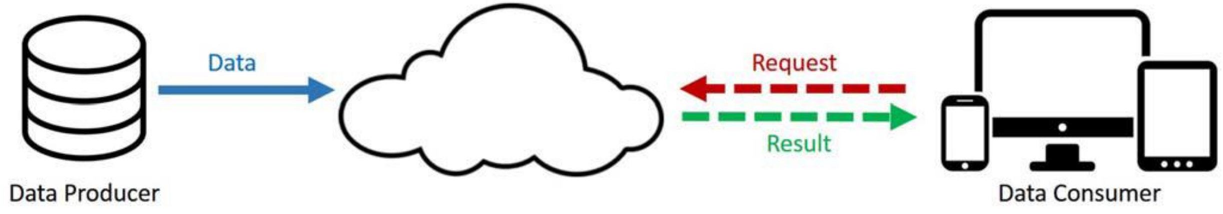
Finals!

Cloud-edge Computing Paradigm

物联网

云计算

Why do we need edge computing



Push from cloud services

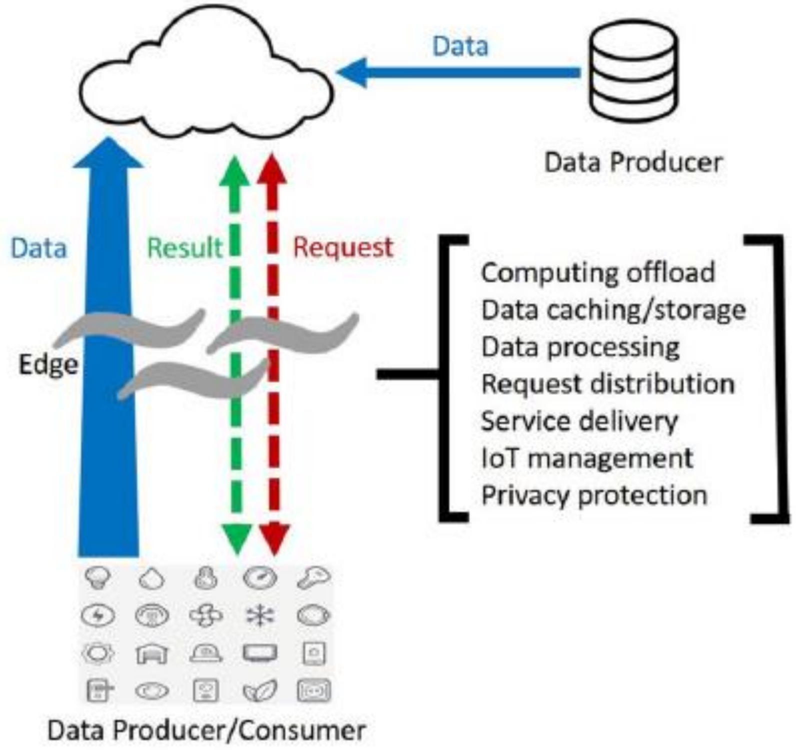
Pull from IoT

Change From Data Consumer to Producer

What is edge computing

We define "edge" as any computing and network resources along the path between data sources and cloud data centers.

Edge computing is interchangeable with fog computing.



Case study

1. Cloud offloading (online shopping services)
2. Video analytics (finding a lost child in the city)
3. Smart home
4. Smart city
5. Collaborative edge

Fig. 4. Collaborative edge example: connected health.

Challenges and opportunities

Programmability

Computing stream that is defined as a serial of functions/computing applied on the data along the data propagation path.
The function/computing distribution metric could be latency-driven, energy cost, TCO, and hardware/software specified limitations.

Challenges and opportunities

Naming

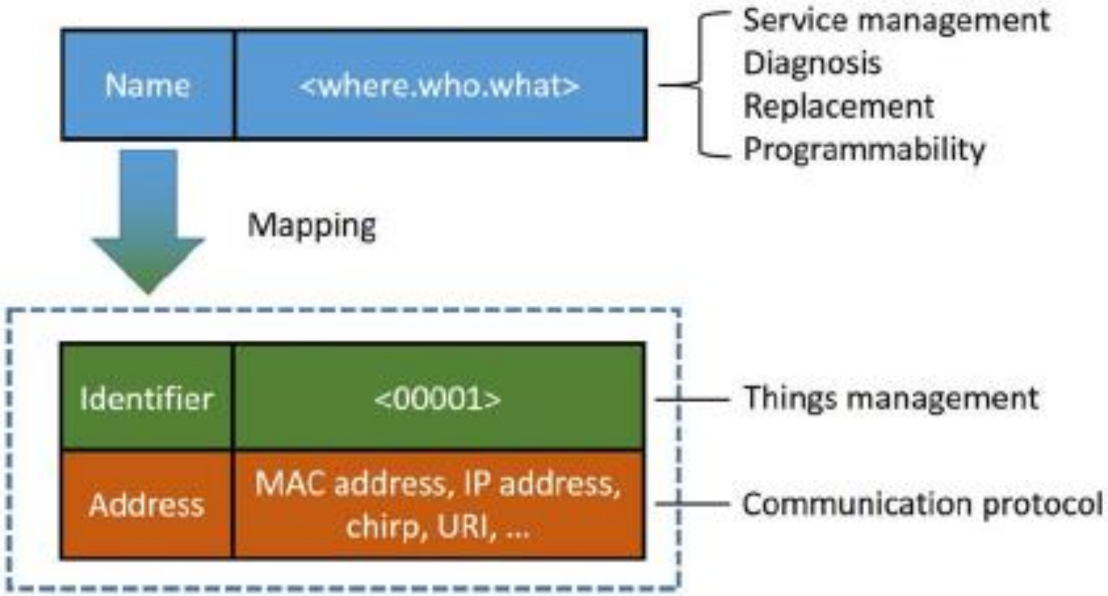


Fig. 5. Naming mechanism in edgeOS.

Challenges and opportunities

Data abstraction

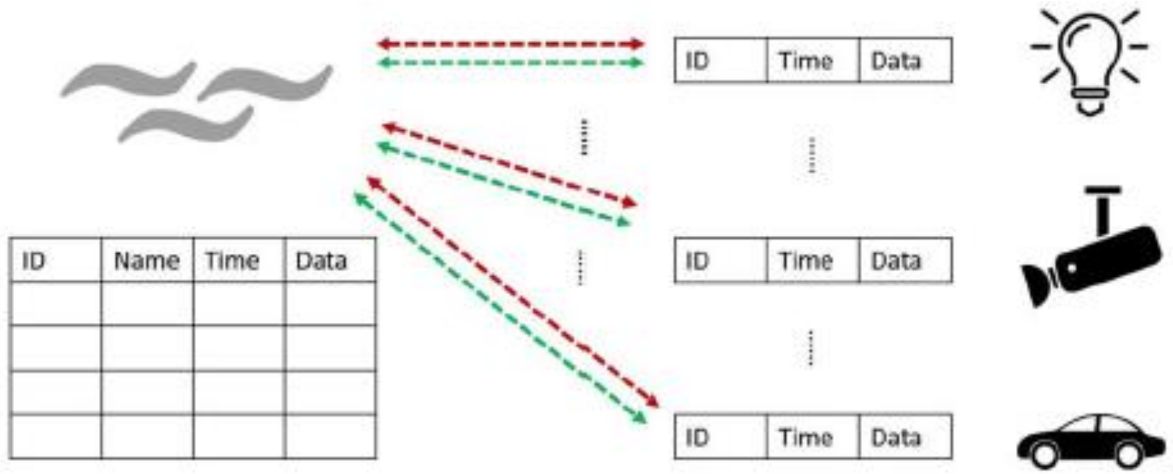


Fig. 6. Data abstraction issue for edge computing.

Challenges and opportunities

Service Management
differentiation, extensibility, isolation, and reliability

Privacy and Security

Optimization Metrics(latency、bandwidth、energy、cost)

Quantifying the Impact of Edge Computing on Mobile Applications

How much can edge computing actually benefit end users
Latency and energy consumption
Compute-intensive and latency-sensitive applications such as mobile augmented reality

Experimental applications

Comet: a existing tool that can transparently migrate threads from a mobile device to a remote server and back

Application	Request size (avg)	Response size (avg)
FACE	62 KB	< 60 bytes
MAR	26 KB	< 20 bytes
FLUID	16 bytes	25 KB

Figure 1: Network load of prepartitioned apps

Application	Total Transfer Size	# of Transfers †
Linpack	$\approx$ 10 MB	1
CPU Benchmark	$\approx$ 80 KB	1
PI Benchmark	$\approx$ 10 MB	15

† Number of thread migrations for each run

Figure 2: Network load of COMET apps

Experimental Setup

Cloudlet: a mobility-enhanced small-scale cloud datacenter that is located at the edge of the Internet.

Smartphone (Samsung Galaxy Nexus)	Netbook (Dell Latitude 2120)
ARM® Cortex-A9 1.2 GHz, 2 cores 1 GB RAM 32 GB Flash 802.11a/b/g/n WiFi	Intel® Atom™ N550 1.5 GHz, 2 cores 2 GB RAM 250 GB HDD 802.11a/g/n WiFi

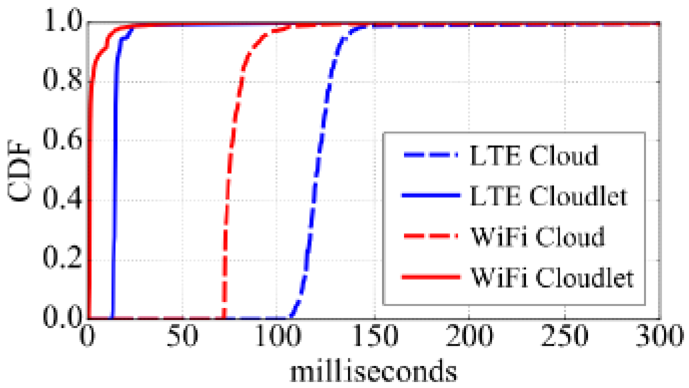
Figure 4: HW configuration of mobile devices

Cloudlet VM on Dell Optiplex 9010	Cloud (Amazon AWS) c3.2xlarge instance
Intel® Core® i7-3770 2.7 GHz†, 4 VCPUs 4 GB RAM 8 GB Virtual disk 1 Gbps Ethernet	Intel® Xeon E5-2680 v2 2.8 GHz, 8 VCPUs 15 GB RAM 160 GB SSD Amazon Enhanced Network

†We limit the CPU to 2.7 GHz and disable Turbo boost.

Figure 3: VM and HW specs at offloading sites

Experimental Setup



No offload
Cloud-WiFi
Cloudlet-WiFi
Cloudlet-LTE
Cloud-LTE

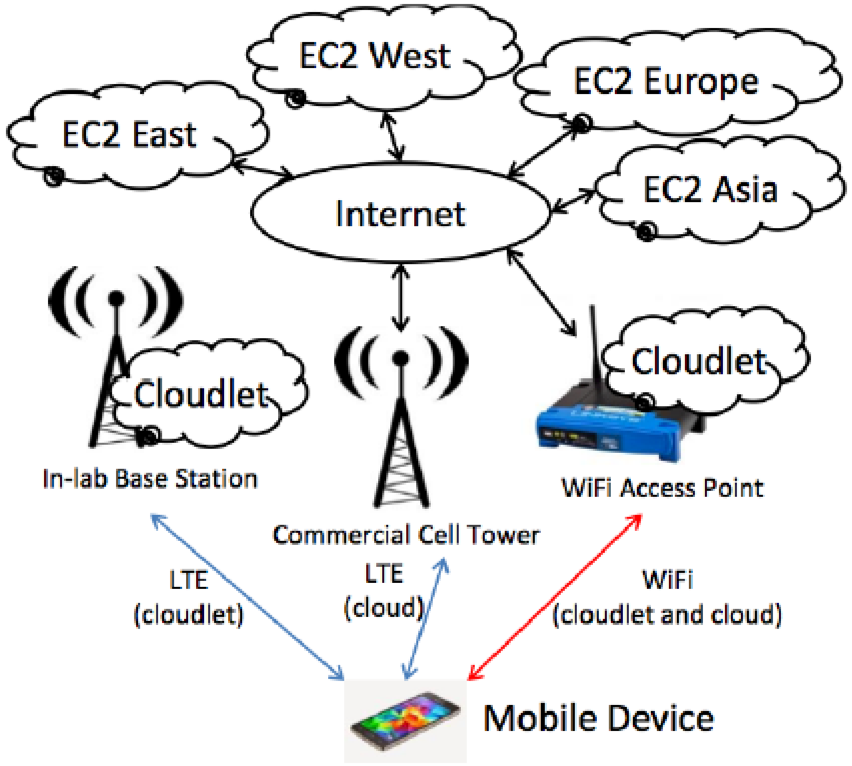
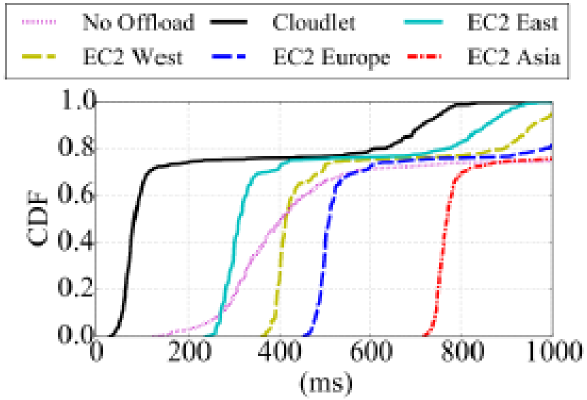
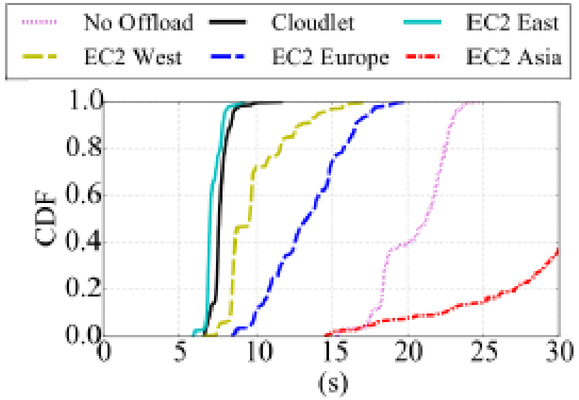


Figure 5: Network setup for the experiments

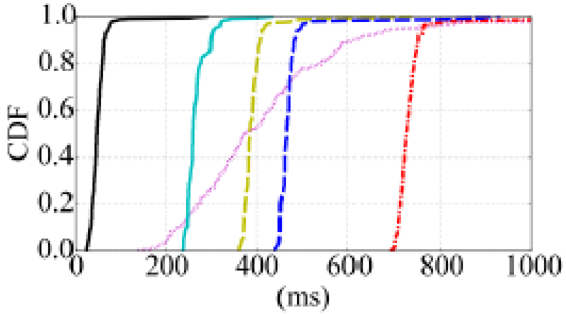
WiFi offloading performance



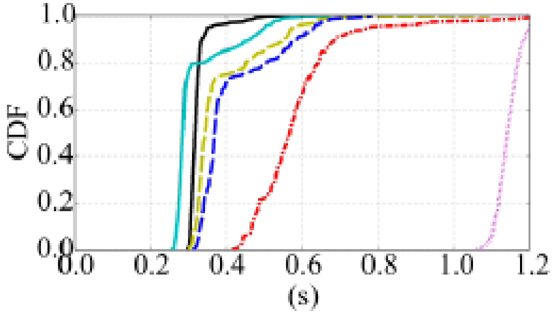
(a) FACE (300 images per run)



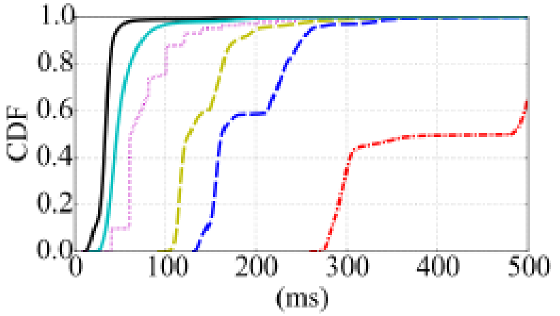
(a) Linpack



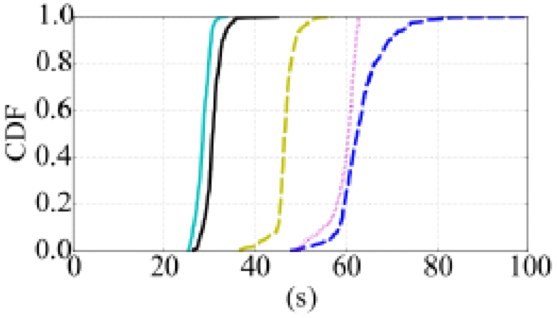
(b) MAR (100 images per run)



(b) CPU Benchmark



(c) FLUID (10 min runs)



(c) PI Benchmark

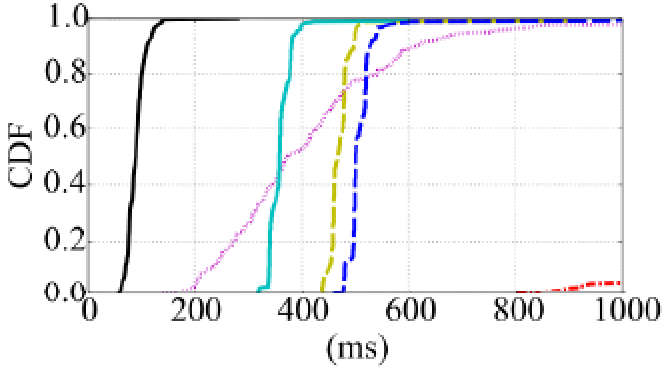
WiFi offloading and energy

	Offload	None	Cloudlet	East	West	Europe	Asia
Face†	(J/query)	12.4 (0.5)	2.6 (0.3)	4.4 (0.0)	6.1 (0.2)	9.2 (4.1)	9.2 (0.2)
Fluid†	(J/frame)	0.8 (0.0)	0.3 (0.0)	0.3 (0.0)	0.9 (0.0)	1.0 (0.0)	2.2 (0.1)
MAR†	(J/query)	5.4 (0.1)	0.6 (0.1)	3.0 (0.8)	4.3 (0.1)	5.1 (0.1)	7.9 (0.1)
Linpack	(J/run)	40.3 (2.6)	13.0 (0.7)	13.3 (2.3)	16.9 (1.8)	18.2 (1.9)	38.1 (4.1)
CPU	(J/run)	9.6 (1.4)	5.7 (0.3)	5.9 (0.3)	5.8 (0.3)	5.9 (0.2)	6.0 (0.2)
PI	(J/run)	129.7 (2.9)	53.9 (2.1)	57.6 (1.8)	107.6 (8.6)	162.8 (18.0)	203.4 (16.7)

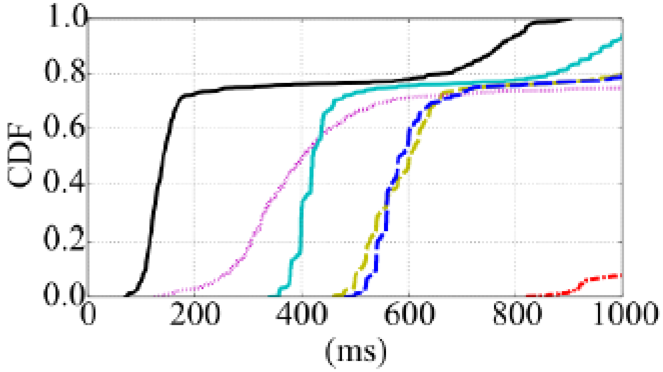
Numbers in parentheses are standard deviations from three runs. †The display is turned off during energy measurement.

Figure 9: Energy consumption on mobile devices

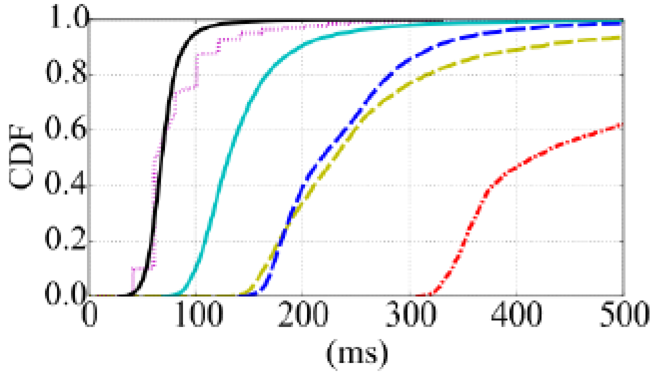
LTE offloading performance



(b) MAR (100 images per run)



(a) FACE (300 images per run)



(c) FLUID (10 min runs)

Mobile Edge Computing: Progress and Challenges

A Edge Analytics Demo

An Edge Analytics Demo

This demo is to showcase the following

1.How sensors and digitized elements get locally connected with one or more IoT gateway instances in order to gather and transmit any useful and usable data to the IoT gateway. In other words, multi-structed and massive data getting generated by various sensors and sensors-attached assets in a particular environment (say, homes, hotels, hospitals, etc.) are received and temporarily stocked by IoT gateways / middleware/brokers for purpose-specific data analytics.

2.By deploying an edge analytics and application development platform in the IoT gateway (Raspberry Pi was used for our demo), all kinds of data getting collected are getting cleansed and crunched in real-time in order to emit out actionable and timely insights.

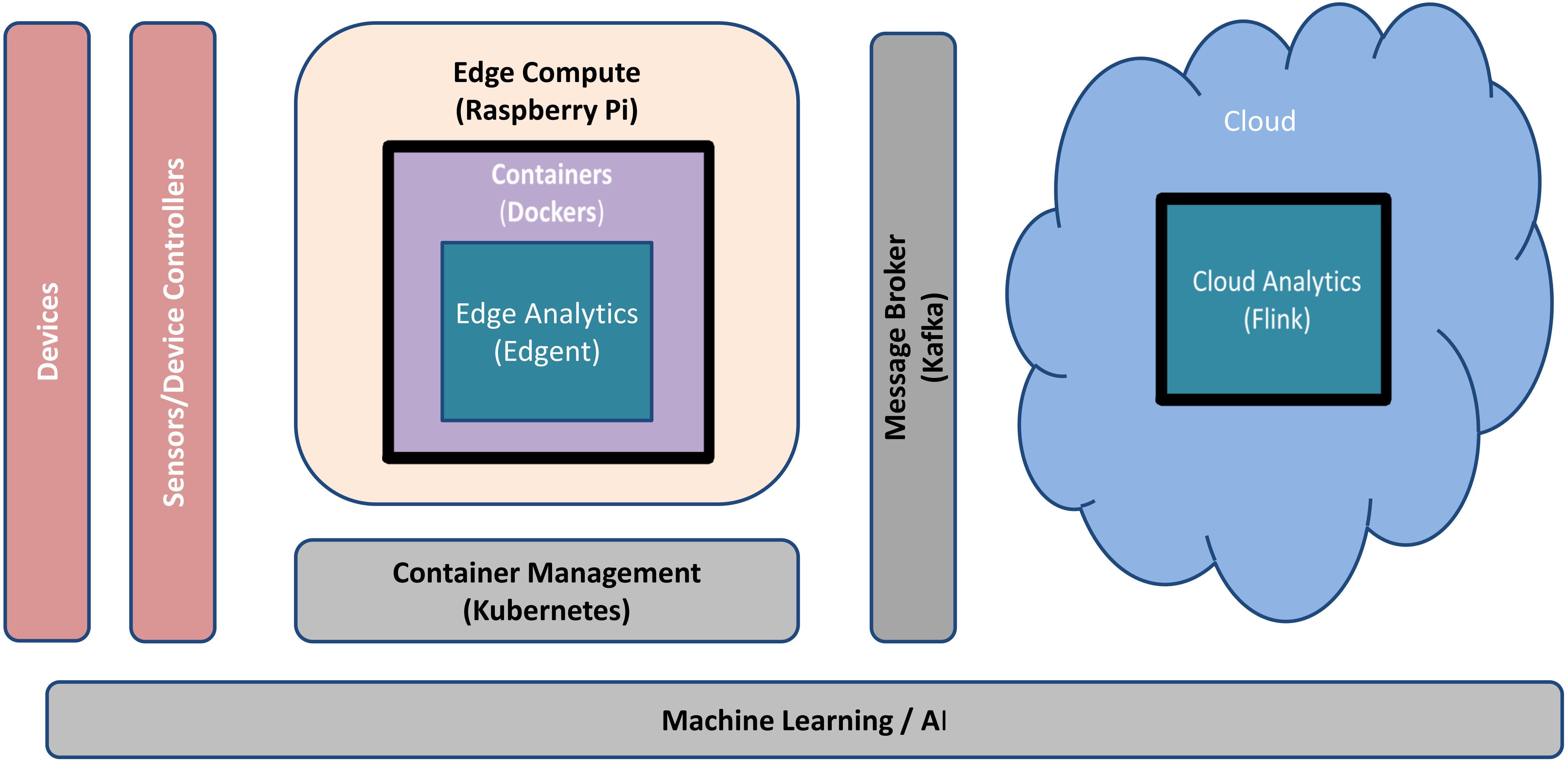
3.The IoT gateway also contributes in filtering out irrelevant data at the source itself so that a very limited amount of useful data gets transmitted to the faraway clouds to facilitate historical and comprehensive big data analytics. The IoT gateway acts as an intermediary between scores of on-premise edge systems and off-premise clouds.

4.IoT gateway modules (typically touted as fog devices) act as the master node/leader in monitoring, measuring and managing various dynamic edge devices and their operational parameters

5.IoT gateway modules seamlessly and spontaneously integrate the physical world with the cyber world (cloud services, applications, databases, platforms, etc.)

6.IoT gateway activates, augments, and adapts actuation devices (edge) based on the insights extricated through analytics in real time

The Macro-level Architecture of our Demo System



The Demo Components

- **Raspberry Pi Configuration Steps:**
  - https://www.raspberrypi.org/documentation/configuration/
  - Model 3 b+, Configuration – 1 GB RAM, 64GB SD card
  - Processor Type: Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4 GHz
  - Ports: 3 USBs, HDMI, 2 WLAN, 1 Ethernet, Bluetooth

- **IR/Motion Sensor / Pulse Rate Monitor**

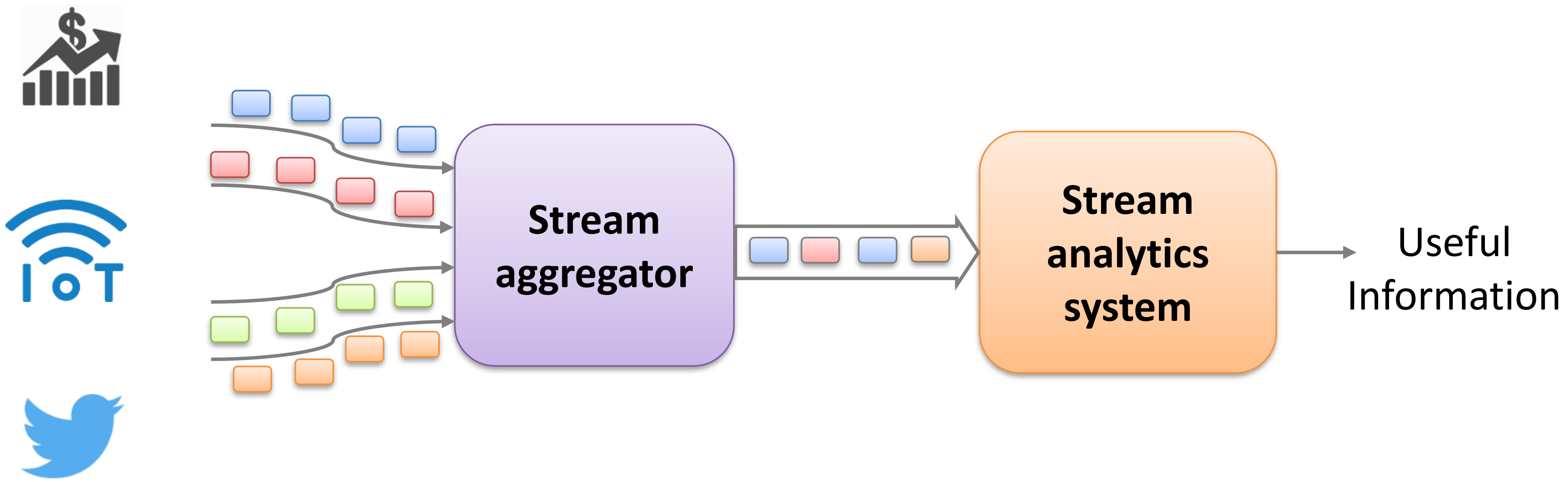
- **Pi4j** - http://pi4j.com/download.html

- **Apache Edgent 1.2.0**
  - https://developer.ibm.com/recipes/tutorials/setting-up-apache-edgent-on-my-raspberry-pi-3/
- **Docker Container -** through the clustering of heterogeneous edge / fog devices
- **AWS Compute Instance**
- **Apache Flink 1.4.2**
  - https://data-flair.training/blogs/install-configure-apache-flink-ubuntu/
- **Not used for this demo/workshop:**
  - Kafka
  - Kubernetes

The Raspberry Pi PIN Layout

	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7
DNC	27	28	DNC
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

Modern online services



Processing streaming data from different sources

Modern online services

Tension

Low latency

Efficient resource utilization

Approximate computing

Approximate computing

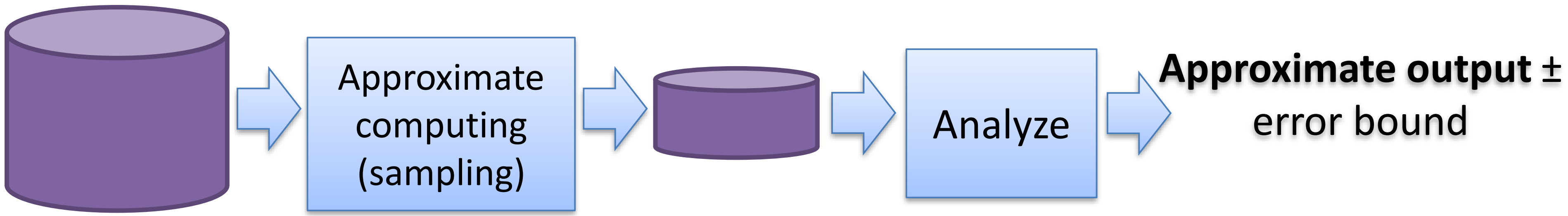
Many applications:
Approximate output is good enough!

The proportion of data is useful for this application

Live taxi heatmap

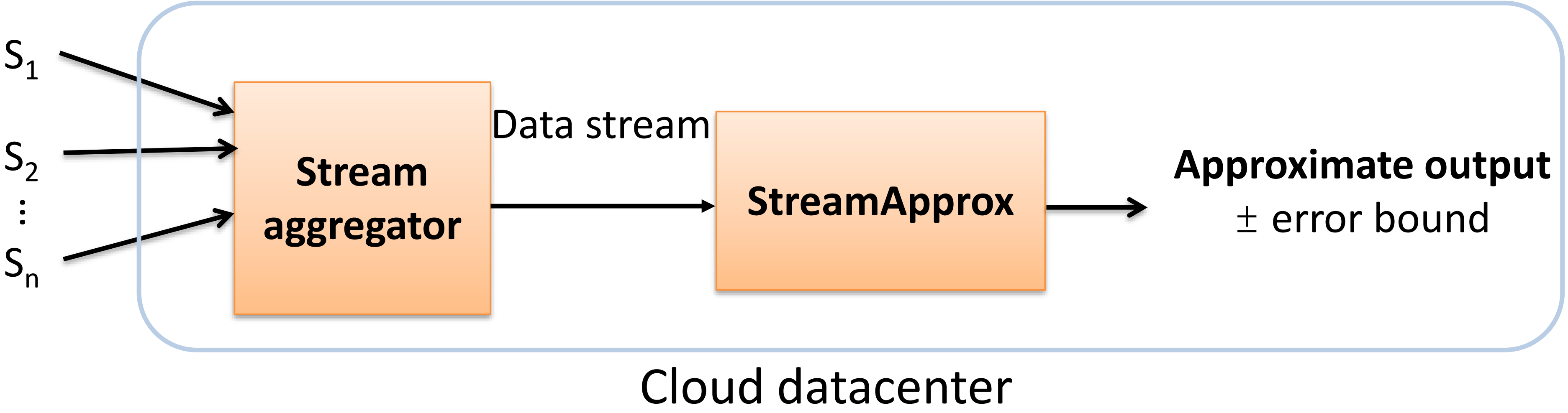
Approximate computing

Idea: To achieve low latency, compute over a sub-set of data items instead of the entire data-set



State-of-the-art system

StreamApprox [Middleware'17]

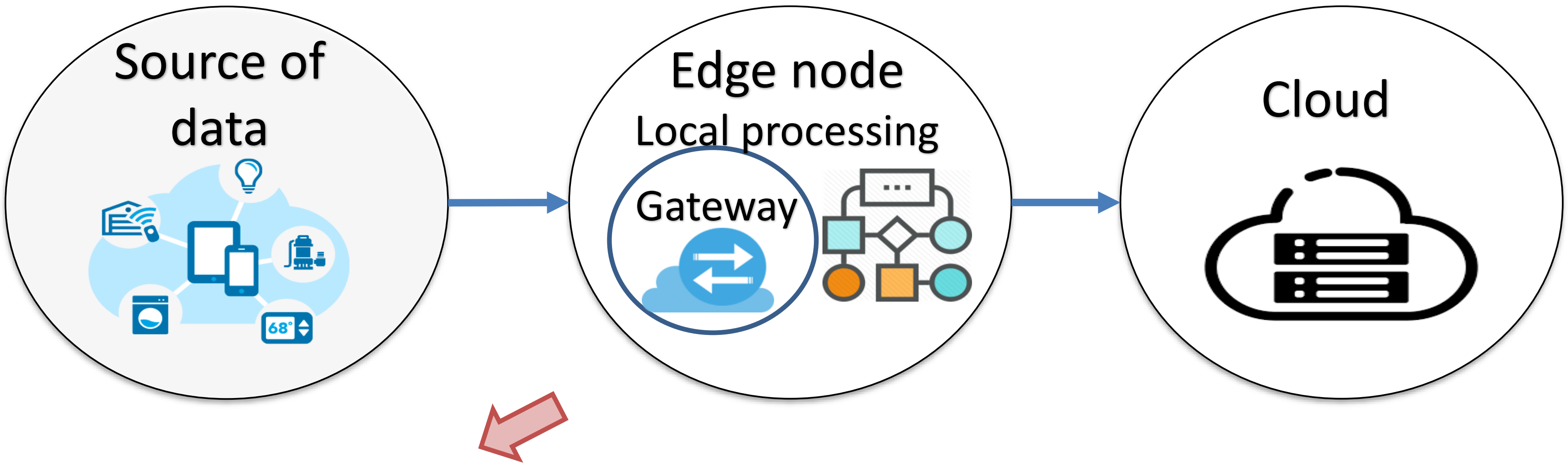


Limitations:

- It wastes bandwidth
- It utilizes only cloud datacenter resources

Edge computing

Allows data to be processed at the edge node before it's sent to the cloud

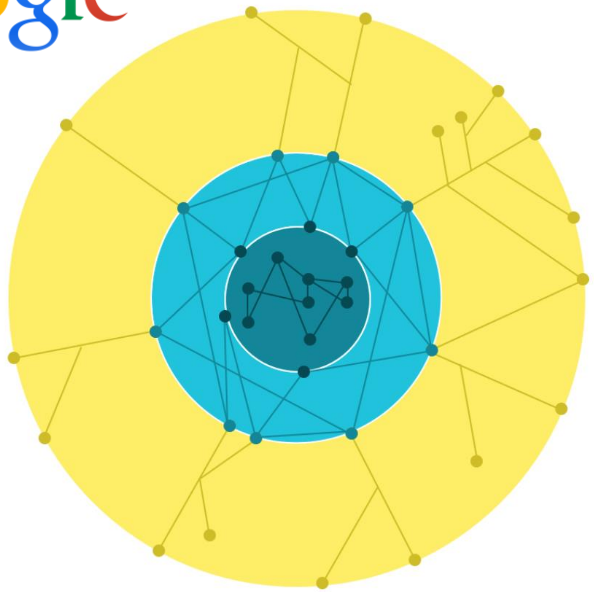


Opportunities:

- Providing more computing resources
- Saving bandwidth

Edge infrastructure

Google



Google's network infrastructure has three distinct elements:

- Core data centers
- Edge Points of Presence (PoPs)
- Edge caching and services nodes (Google Global Cache, or GGC)

Source: https://peering.google.com/#/infrastructure

Microsoft Azure IoT edge

IBM Watson IoT

amazon AWS IoT

Problem statement

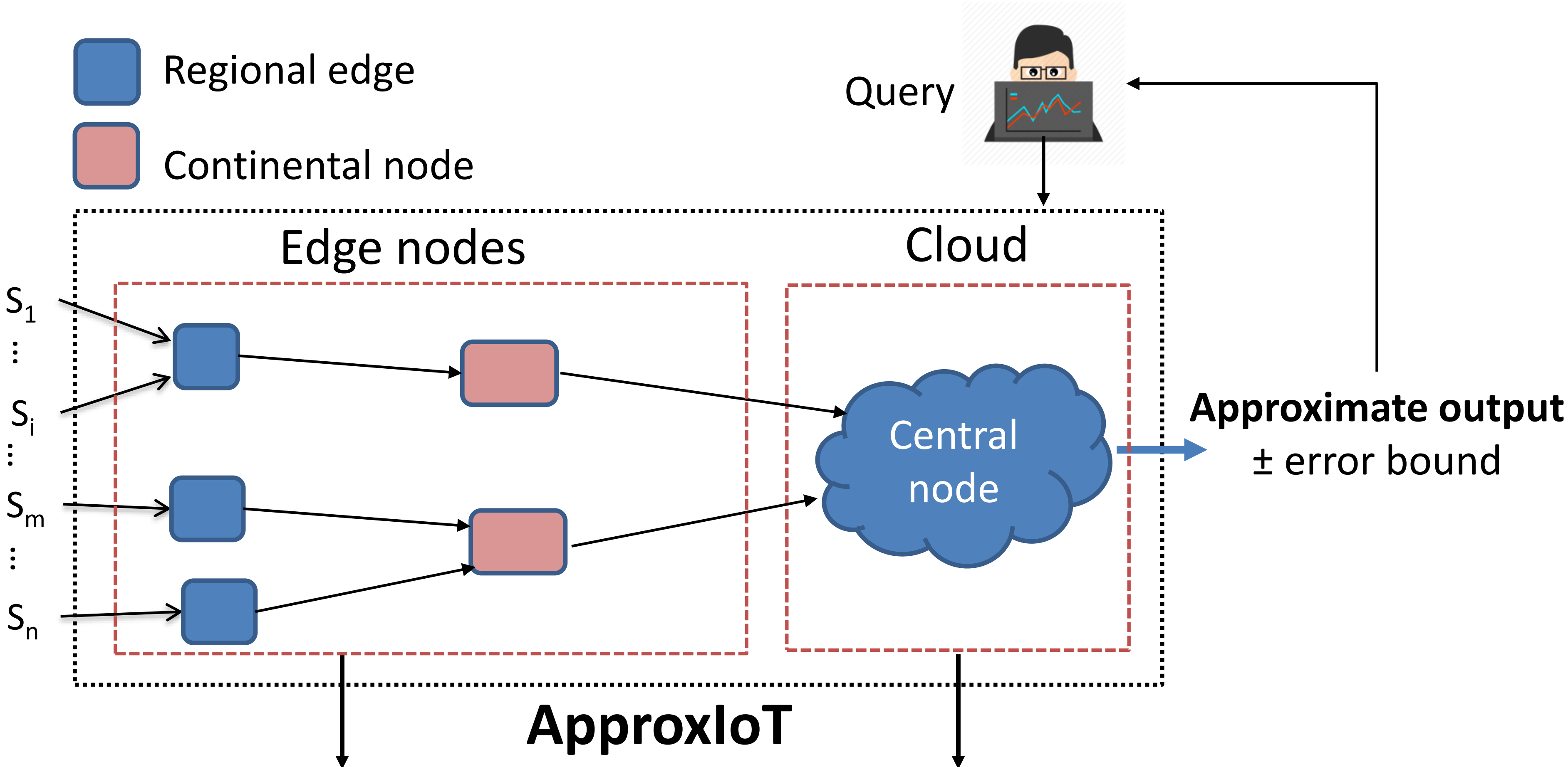
To build a stream analytics system

By utilizing the cloud and edge computing resources
By leveraging approximate computing

Design goals

Efficiency: Efficient utilization of computing resources
Adaptability: Adaptive execution based on the available resources
Transparency: No code change required and resource management

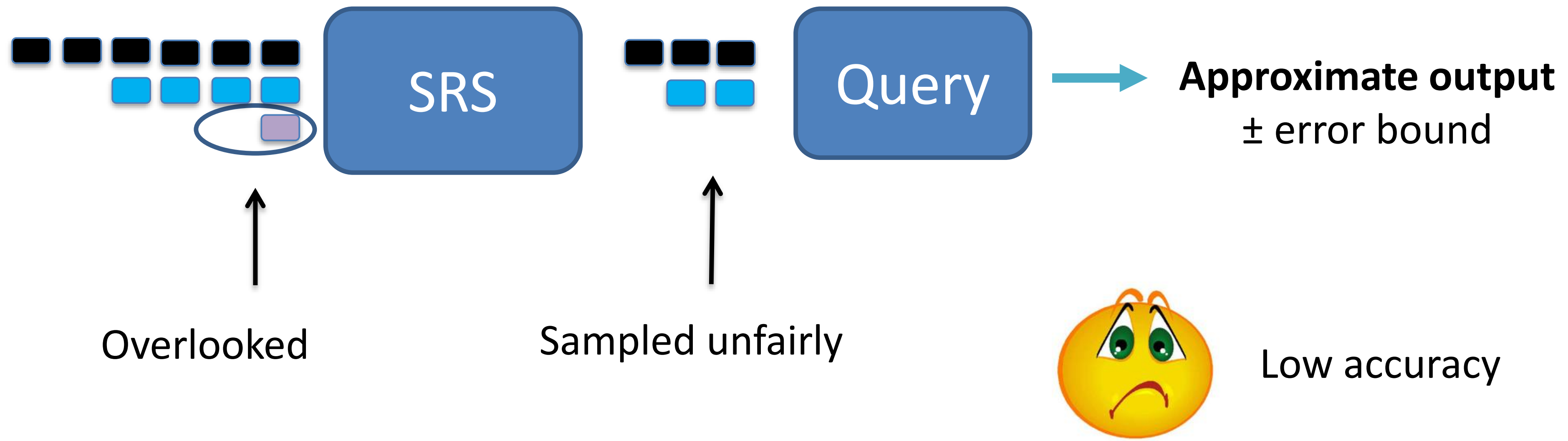
ApproxIoT: Overview



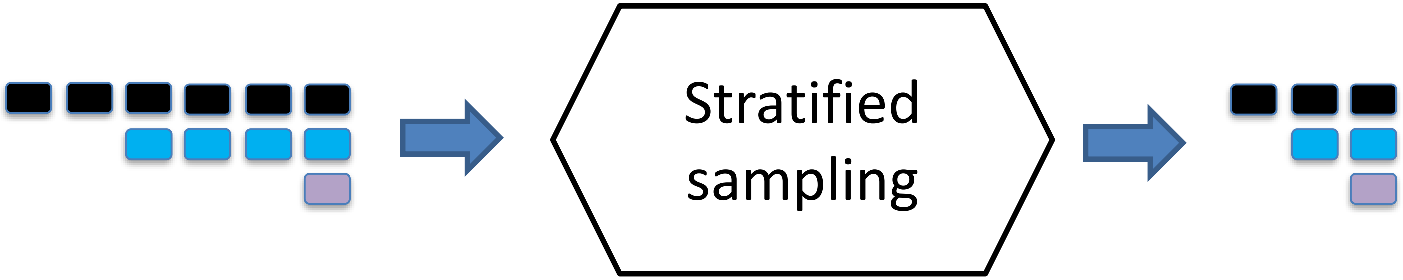
ApproxIoT employs sampling in the distributed environment of edge + cloud

Naïve algorithm

Simple random sampling (SRS)



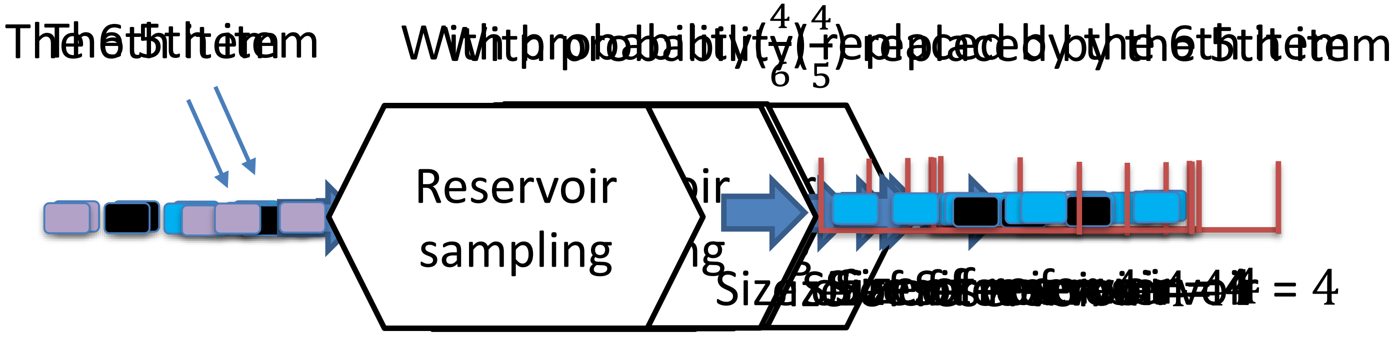
Background: Stratified sampling



Advantage: The sub-streams are sampled fairly

Disadvantage: Requires the knowledge of each sub-stream size

Background: Reservoir sampling



Advantage:

- No pre-knowledge required of sub-stream size

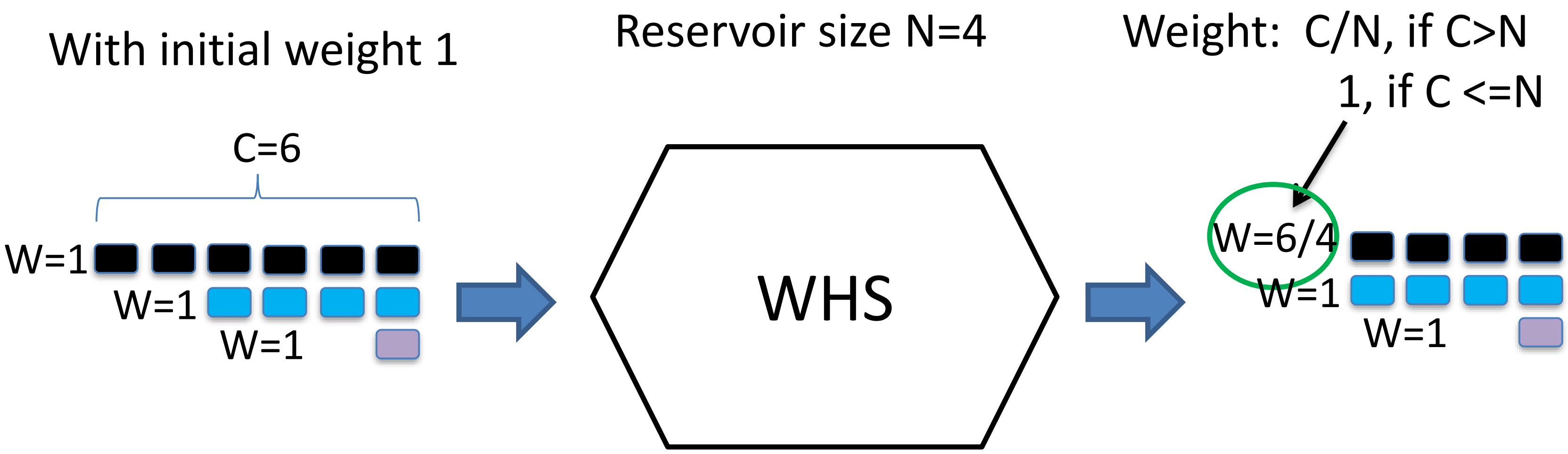
Disadvantages:

- The sub-streams are sampled unfairly
- Difficult to run on multiple nodes

ApproxIoT sampling algorithm

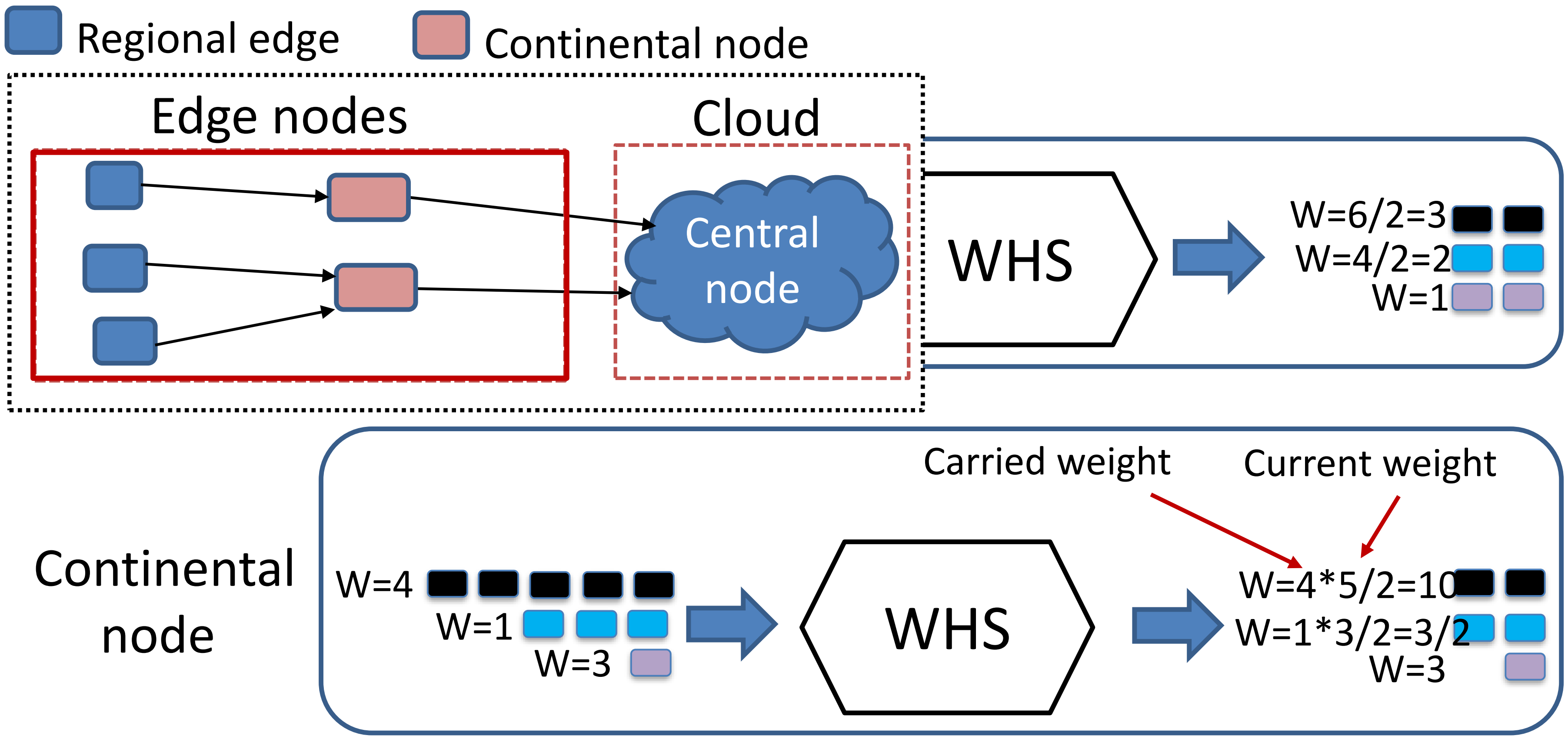
Weighted hierarchical sampling (WHS)

Combining stratified and reservoir sampling



Easy to parallelize, requires no synchronization between sub-streams

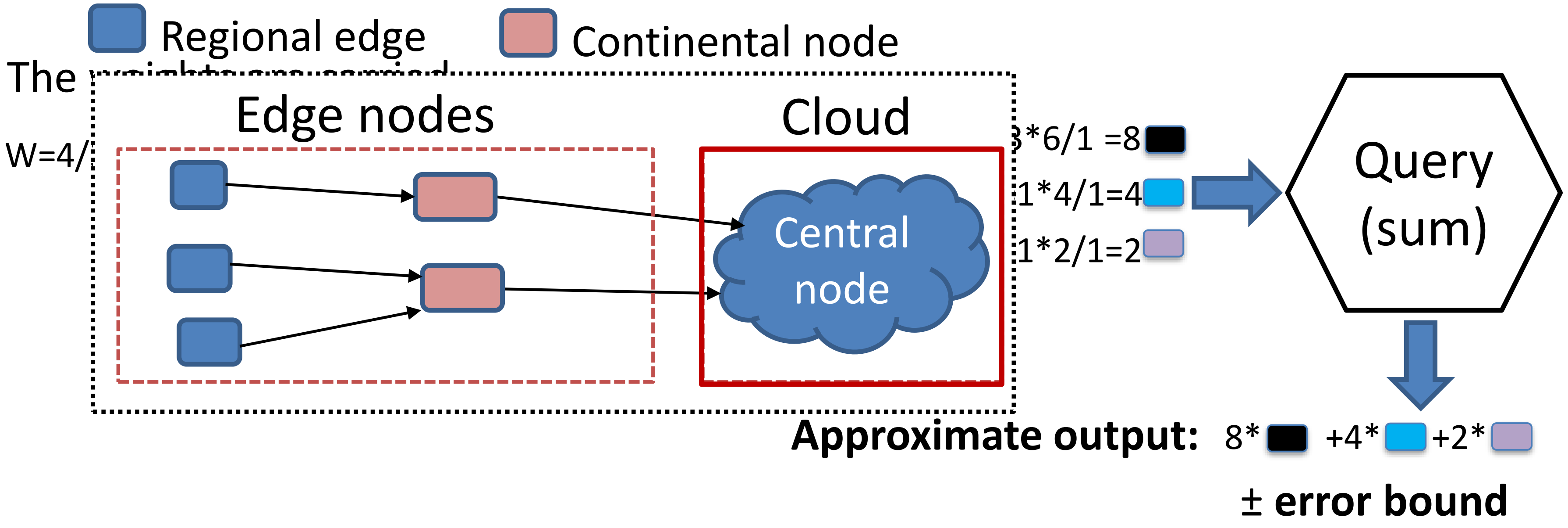
WHS on edge nodes



Reservoir size equals 2

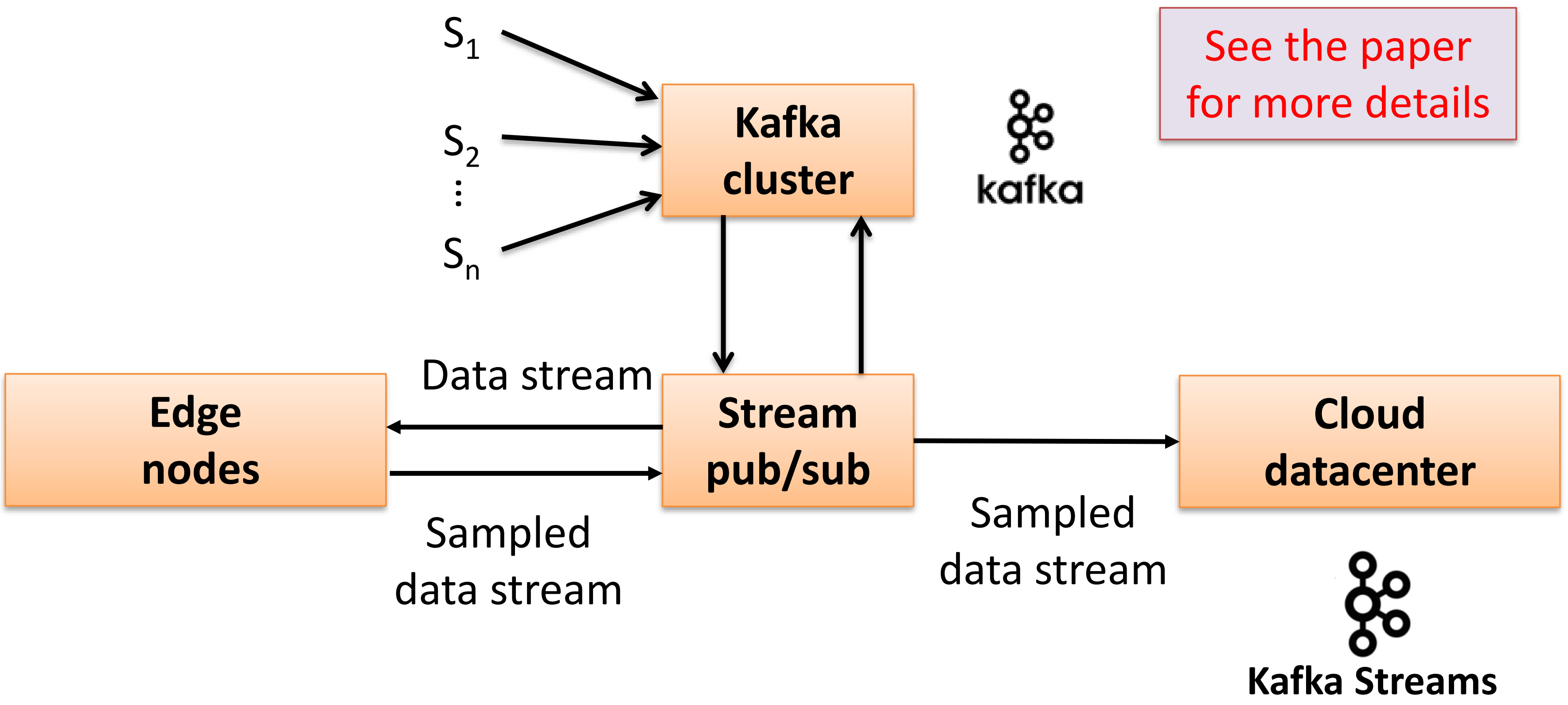
Easy to parallelize, requires no synchronization between computing nodes

ApproxIoT in the cloud



Reservoir size equals 1

Implementation



Experimental setup

Evaluation questions

Accuracy vs. sample size
Throughput vs. sample size

Testbed: 25 nodes

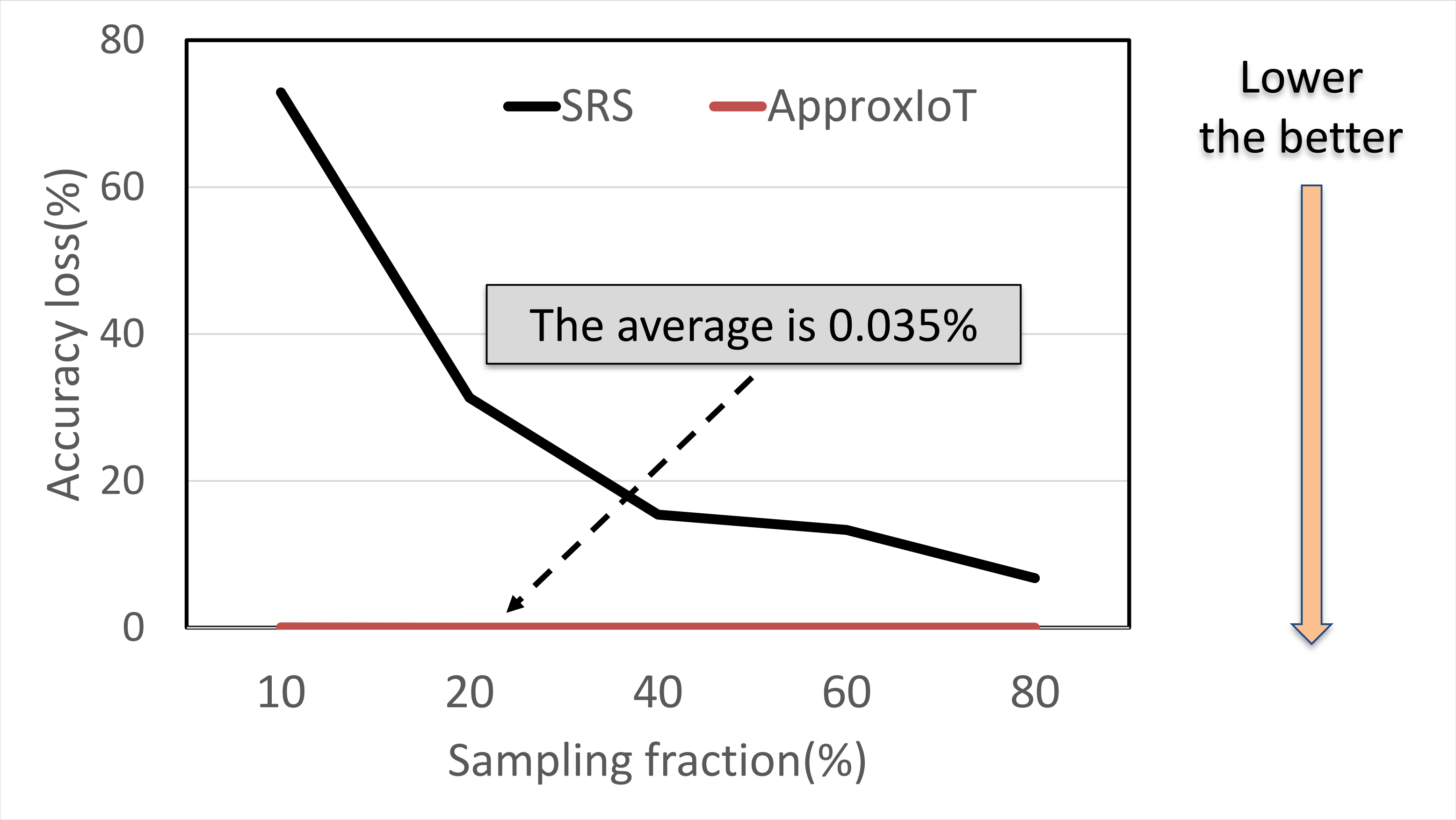
15 nodes for ApproxIoT deployment
10 nodes for Kafka cluster

Datasets:

Synthetic: Poisson and Gaussian distribution
Real: Brasvo pollution and New York Taxi Ride

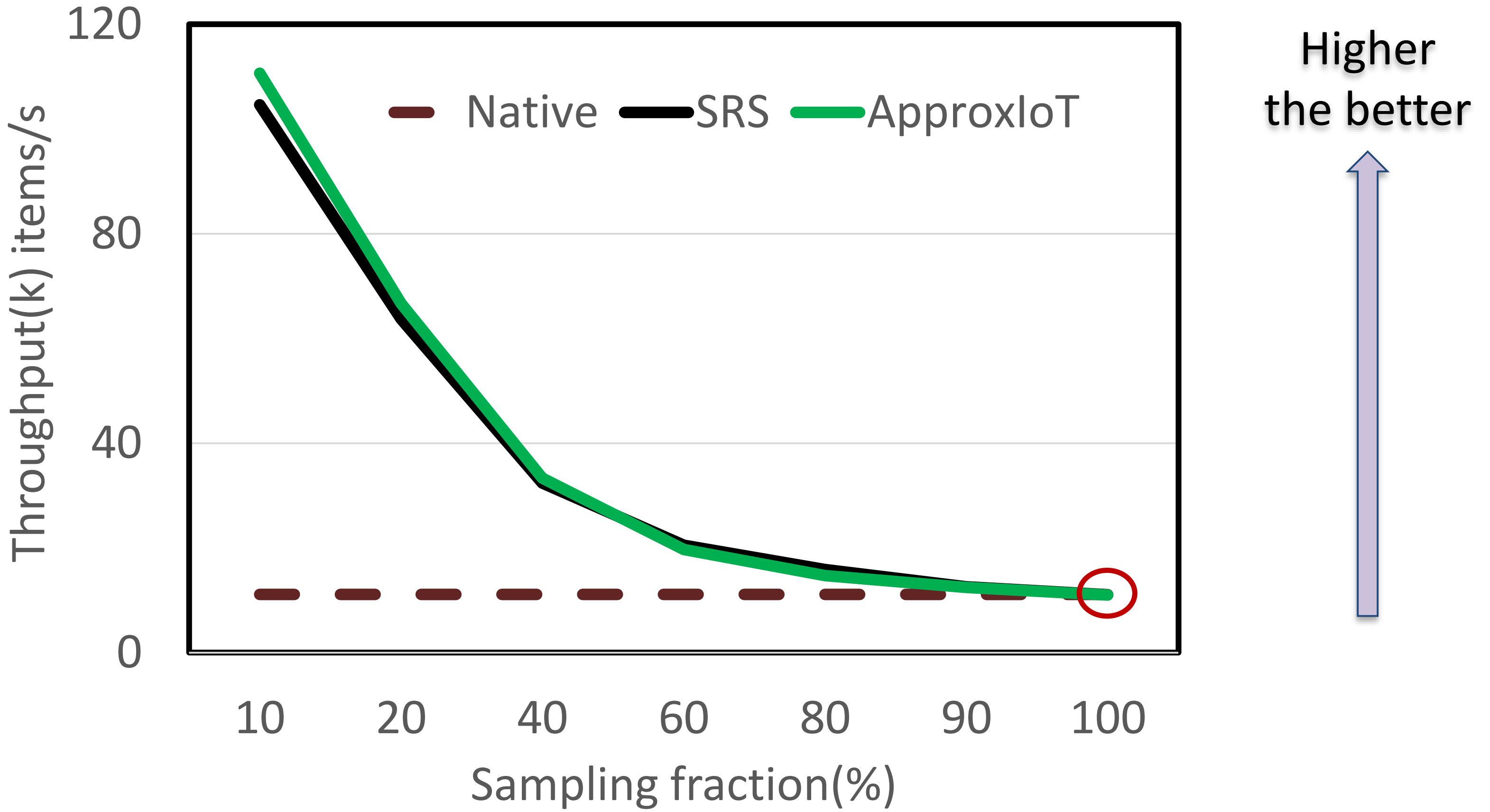
See the paper
for more results!

Accuracy vs. sample size



ApproxIoT: ~2600X higher accuracy over SRS

Throughput vs. sample size



- ApproxIoT has low overhead compared to the native execution
- ApproxIoT has similar throughput as SRS

Conclusion: The promise of edge computing

Infrastructure and design

Measurement works

Related applications

The Design and Implementation of a Wireless Video Surveillance System