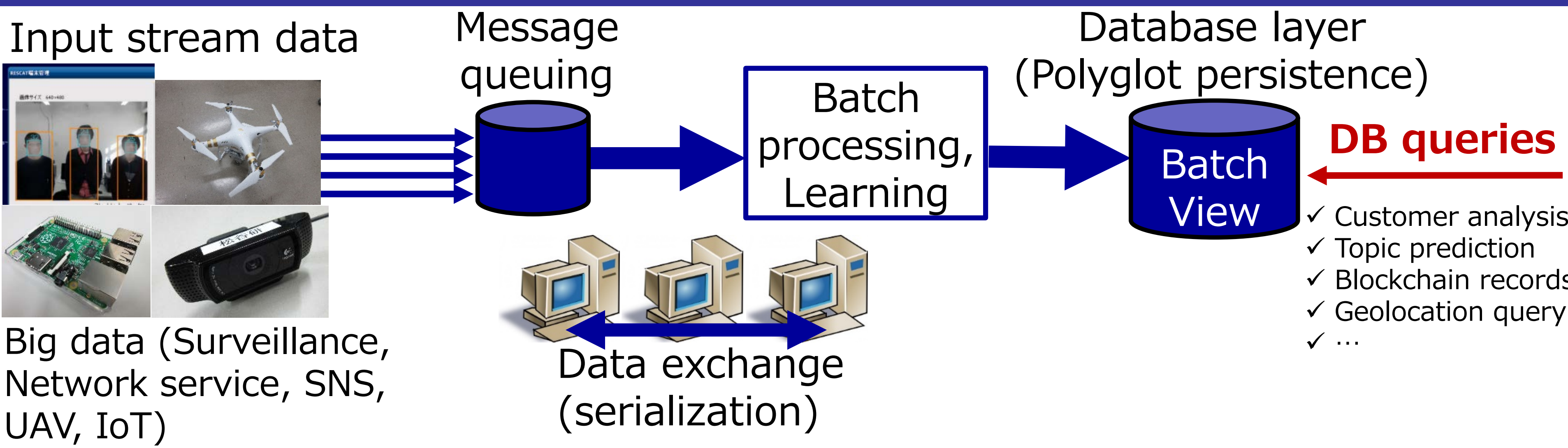
Accelerator Design for Big Data Processing Frameworks

Hiroki Matsutani

Dept. of ICS, Keio University

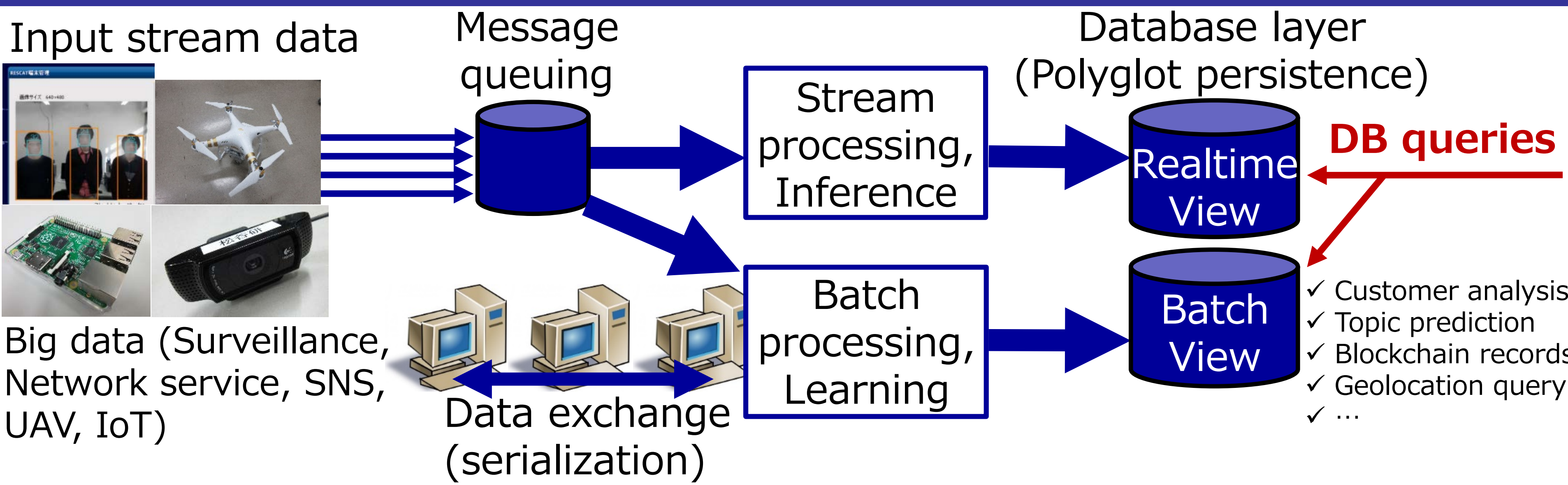
http://www.arc.ics.keio.ac.jp/~matutani

Batch processing



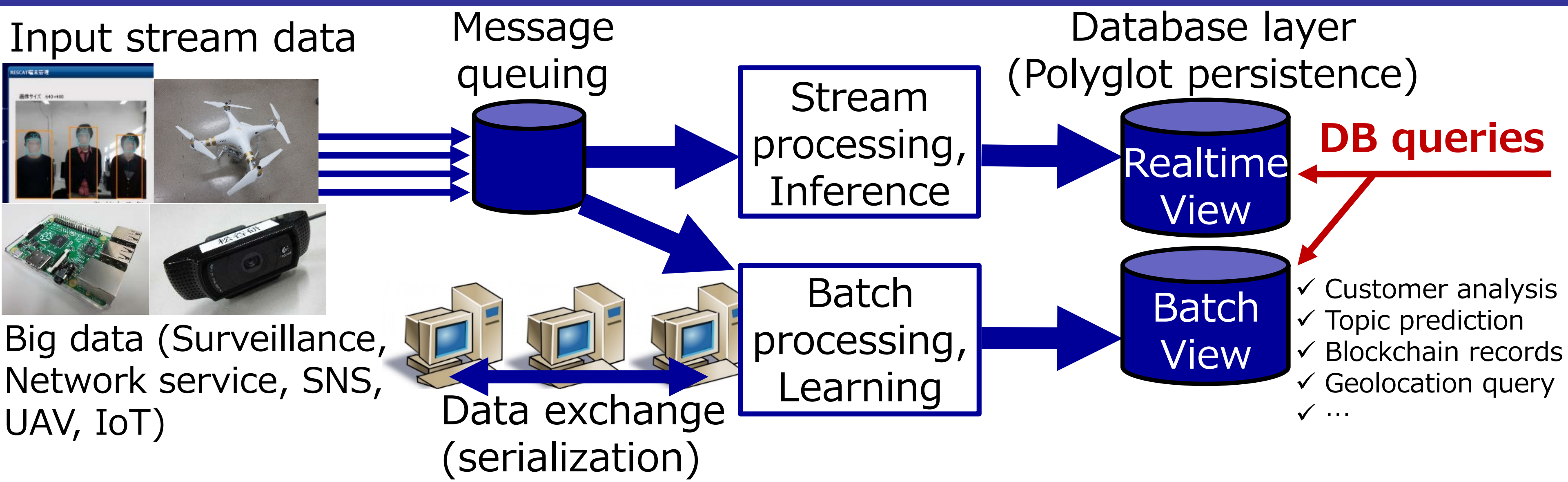
Batch processing requires time and thus recent data cannot be reflected to the analysis result
→ Combine batch and stream processing to make up the realtime capability

Batch + Stream processing



Nathan Marz, et.al., "Big Data: Principles and best practices of scalable realtime data systems", Manning Publications (2015).

Batch + Stream processing

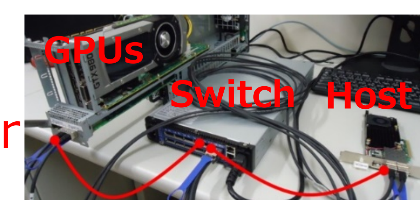


I/O intensive ⟷ **Compute intensive**

Message queuing middleware (Apache Kafka)	Stream processing (Apache Spark Streaming)	Online machine learning (Classification, Outlier detection, Change point detection, Abnormal behavior detection)	Document DB (MongoDB)	Batch processing (Apache Spark)
Serialization (Apache Thrift)	KVS / Column DB (Redis, HBase)		Graph DB (Neo4j), graph processing	Machine learning (Apache Spark MLlib)

Tight integration of I/O and compute →FPGA

Massive parallelism → Networked GPU cluster



Acceleration for data store

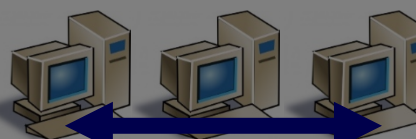
Input stream data

Big data (Surveillance, Network service, SNS, UAV, IoT)

Message queuing

Stream processing, Inference

Batch processing, Learning



Data exchange (serialization)

Database layer (Polyglot persistence)

Realtime View

Batch View

DB queries

- ✓ Customer analysis
- ✓ Topic prediction
- ✓ Blockchain records
- ✓ Geolocation query
- ✓ …

I/O intensive — **Compute intensive**

Message queuing middleware (Apache Kafka)

Serialization (Apache Thrift)

Stream processing (Apache Spark Streaming)

KVS / Column DB (Redis, HBase)

Online machine learning (Classification, Outlier detection, Change point detection, Abnormal behavior detection)

Document DB (MongoDB)

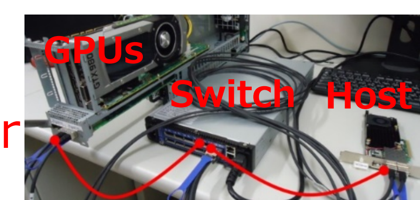
Graph DB (Neo4j), graph processing

Batch processing (Apache Spark)

Machine learning (Apache Spark MLlib)

Tight integration of I/O and compute →FPGA

Massive parallelism → Networked GPU cluster



Multilevel NOSQL cache: FPGA NIC

MPSoC16

[IEEE HoTI'16]

Normal DB Access

Result

In-Kernel KVS Cache (L2$) Hit

NIC Cache (L1$) Hit

Add

Request

Reply

User Space

mongoDB

APACHE HBASE

Neo4j

NetFPGA-10G Device Driver

In-Kernel KVS (Key-Value Store) Cache 512GB

NetFPGA-10G

NIC HW Cache

Machine Learning

DRAM 288MB

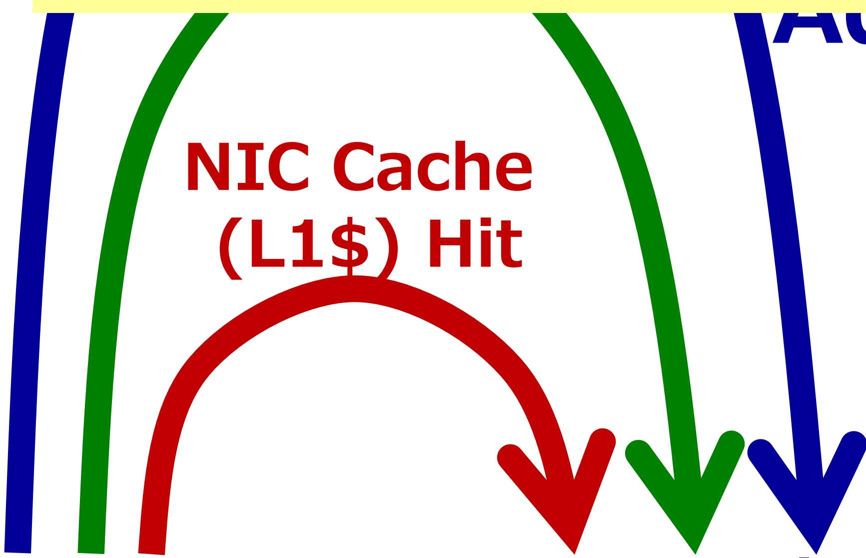
10GbE x4

Multilevel NOSQL cache: FPGA NIC

MPSoC16

Multilevel NOSQL cache:
L1 NOSQL cache ··· FPGA-based hardware cache
L2 NOSQL cache ··· In-kernel software cache

Tradeoffs between capacity and speed:
L1 NOSQL cache ··· Very fast/efficient but small
L2 NOSQL cache ··· Fast and large
Design space exploration → [IEEE HoTI'16]



Cache 512GB

NetFPGA-10G

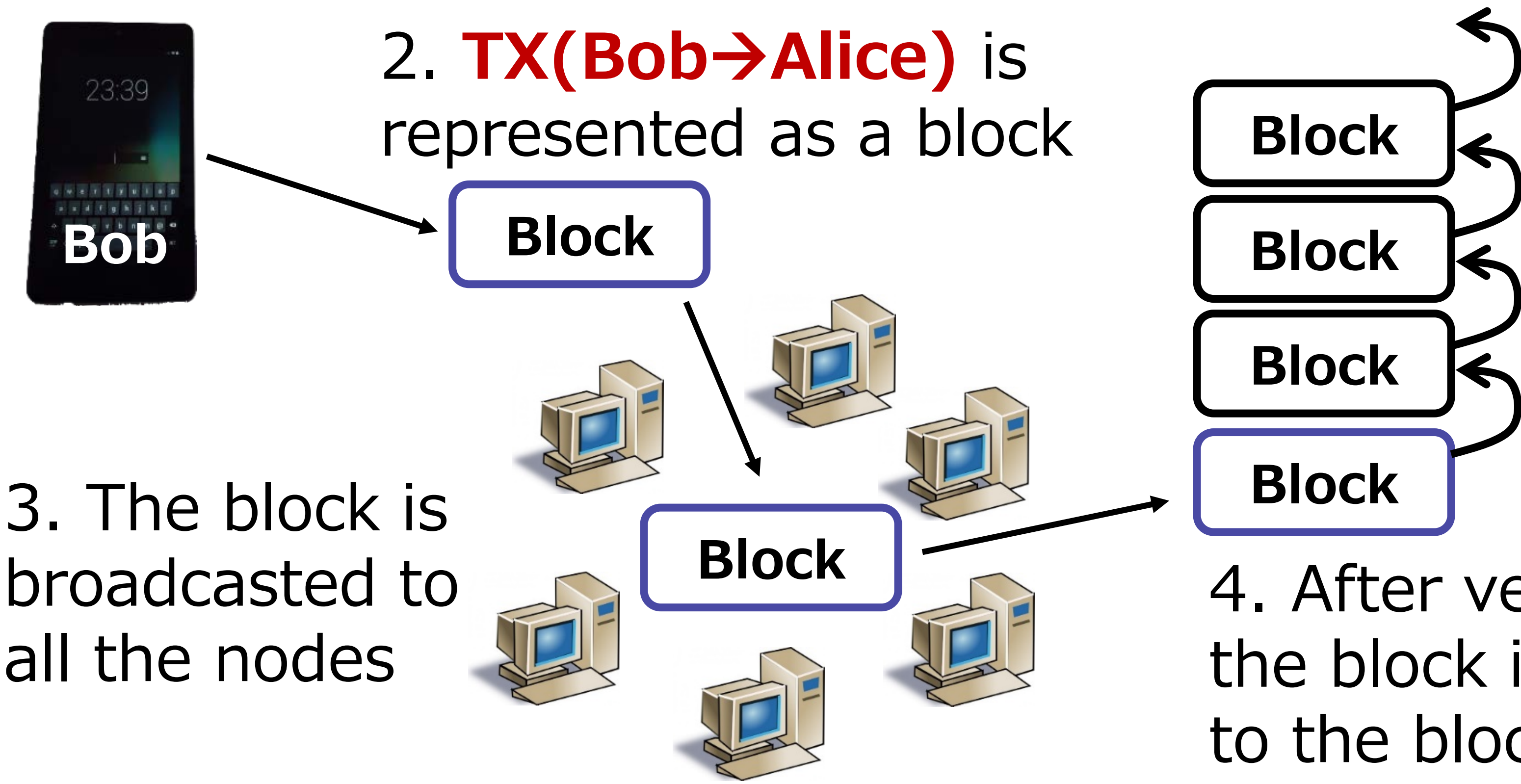
NIC HW Cache
Machine Learning

DRAM 288MB
10GbE x4

FPGA NIC Cache for Blockchain

- Blockchain
  - A chain of blocks each contains transactions verified and shared by all the parties

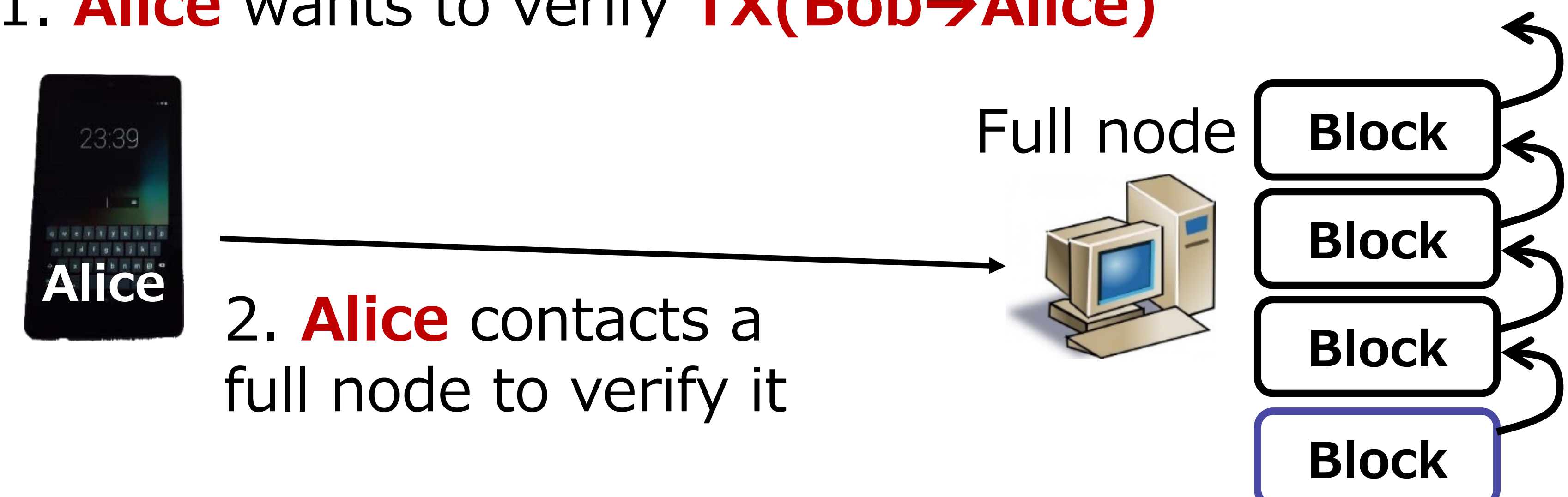
1. **Bob** wants to send money to **Alice**



FPGA NIC Cache for Blockchain

- IoT devices (SPV nodes)
  - Cannot maintain whole the blockchain data (>100GB) due to resource limitation
- Simple payment verification
  - Ask full node to check whether a transaction of interest has been completed or not

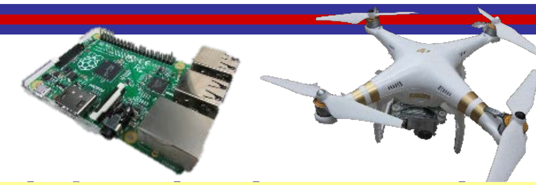
1. **Alice** wants to verify **TX(Bob→Alice)**



2. **Alice** contacts a full node to verify it

FPGA NIC Cache for Blockchain

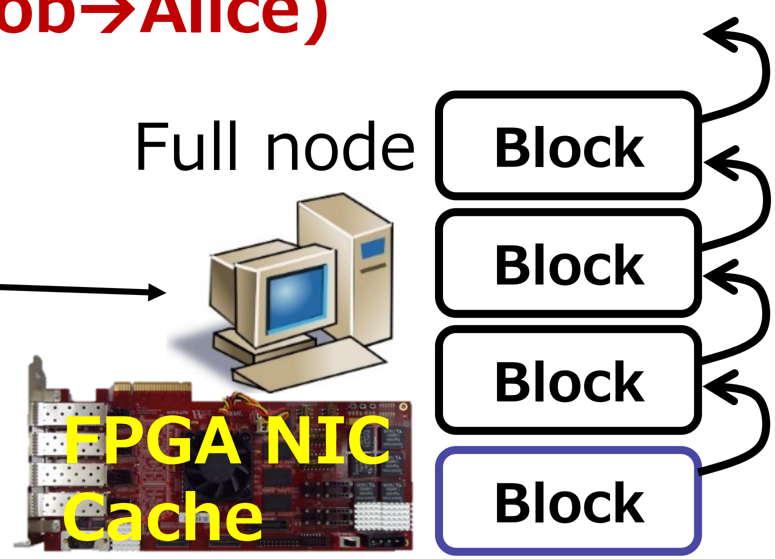
- IoT devices (SPV nodes)



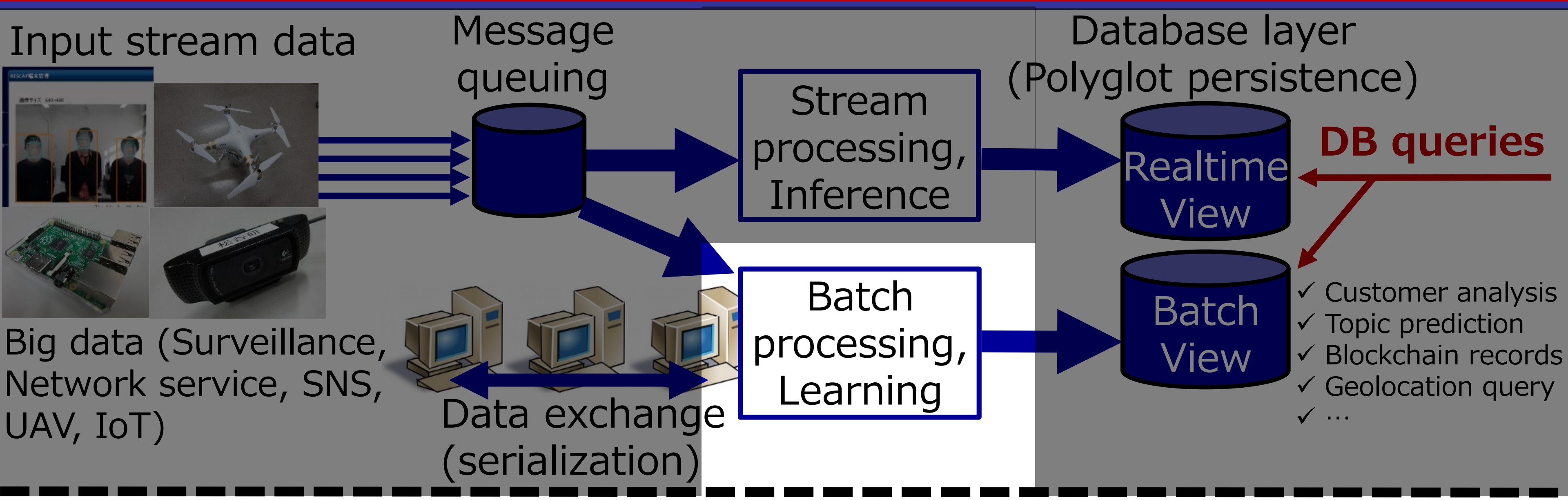
The number of IoT devices that join blockchain will increase
To reduce full node accesses from SPV nodes, FPGA NIC KVS is used as "cache" of blockchain [HEART'17]

1. **Alice** wants to verify **TX(Bob→Alice)**

2. **Alice** contacts a full node to verify it



Acceleration for data processing

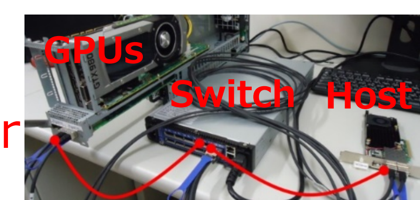


I/O intensive ⟷ **Compute intensive**

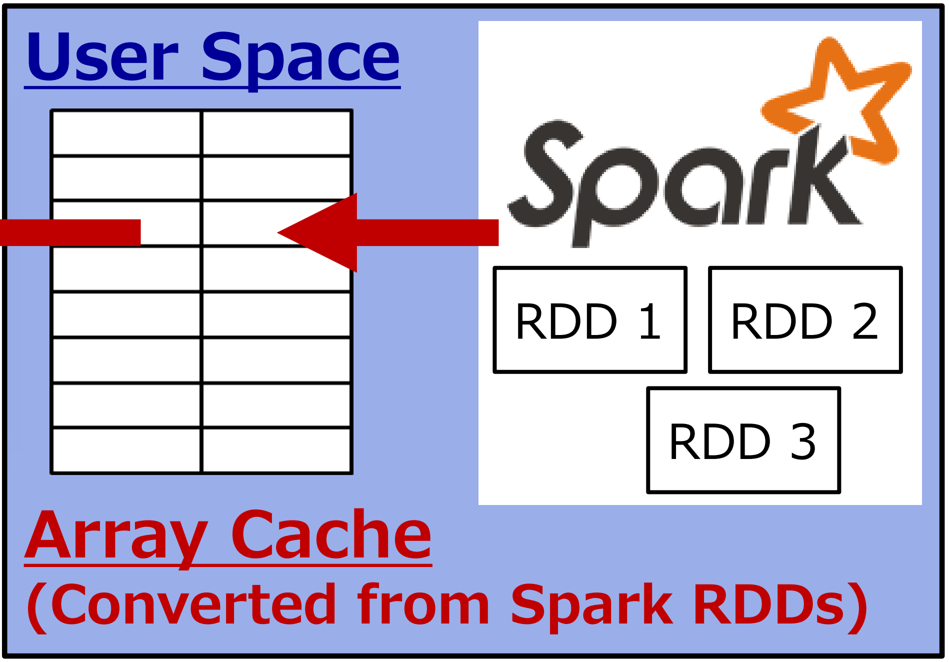
Message queuing middleware (Apache Kafka)	Stream processing (Apache Spark Streaming)	Online machine learning (Classification, Outlier detection, Change point detection, Abnormal behavior detection)	Document DB (MongoDB)	Batch processing (Apache Spark)
Serialization (Apache Thrift)	KVS / Column DB (Redis, HBase)		Graph DB (Neo4j), graph processing	Machine learning (Apache Spark MLlib)

Tight integration of I/O and compute →FPGA

Massive parallelism → Networked GPU cluster

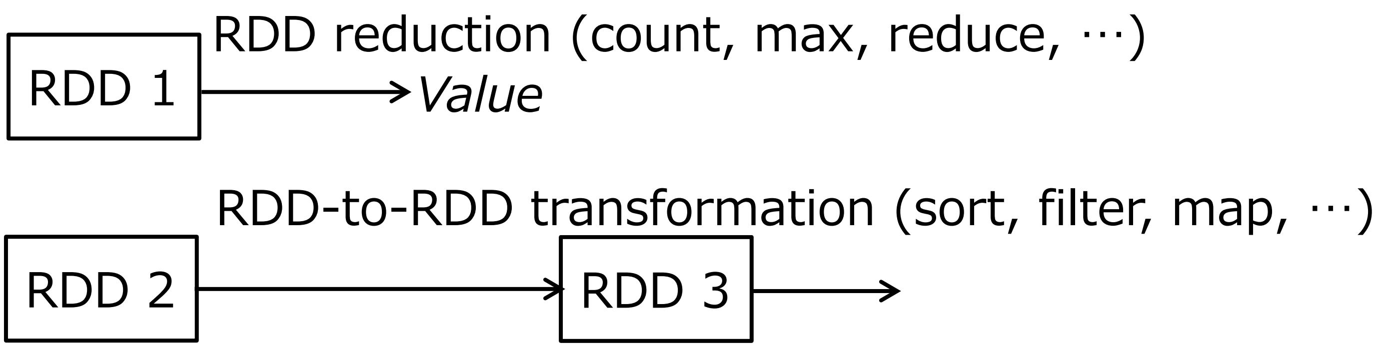


Data processing w/ Spark+GPU



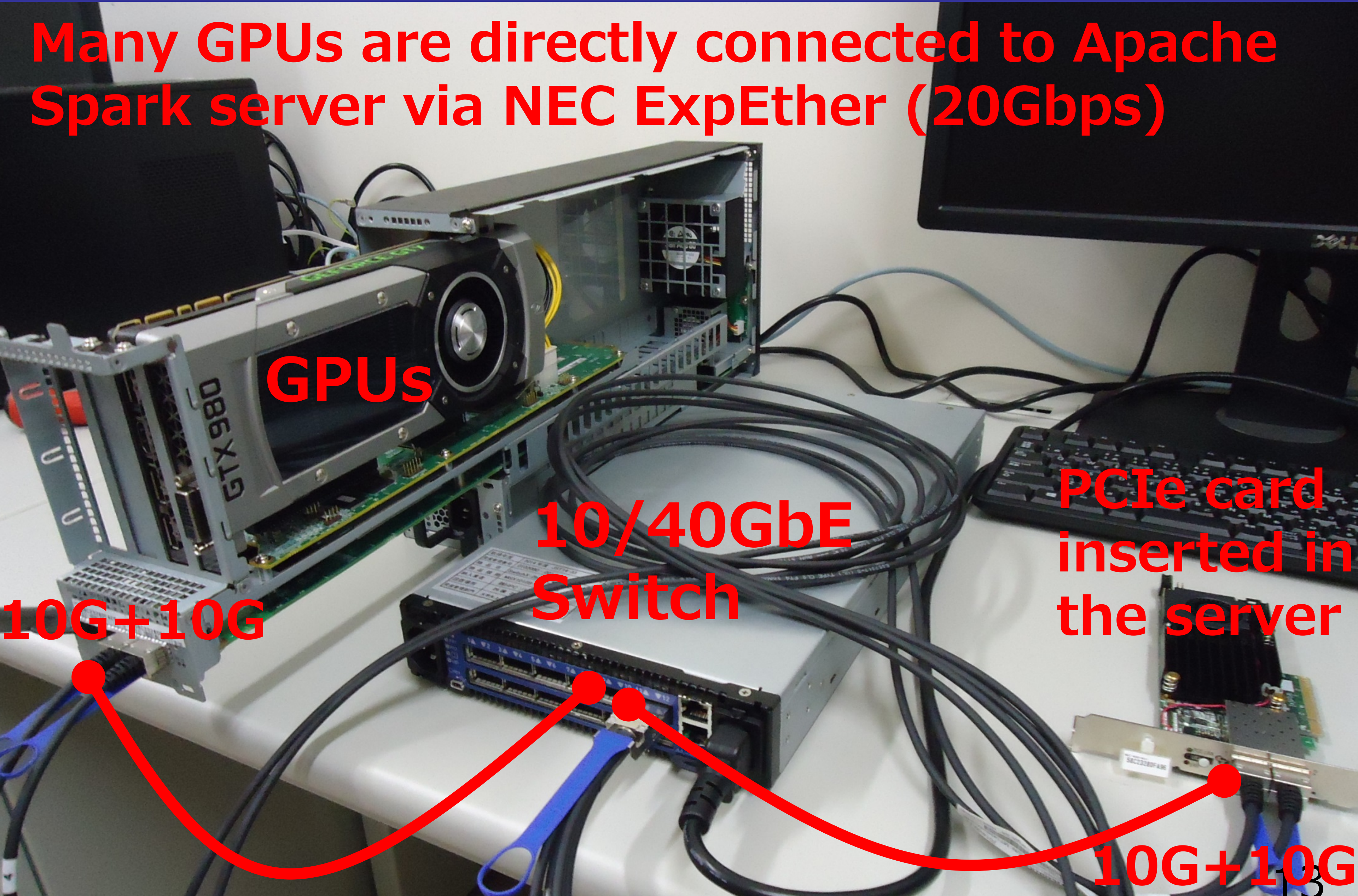
Reduction & transformation of RDDs offloaded to GPU

[HEART'16]

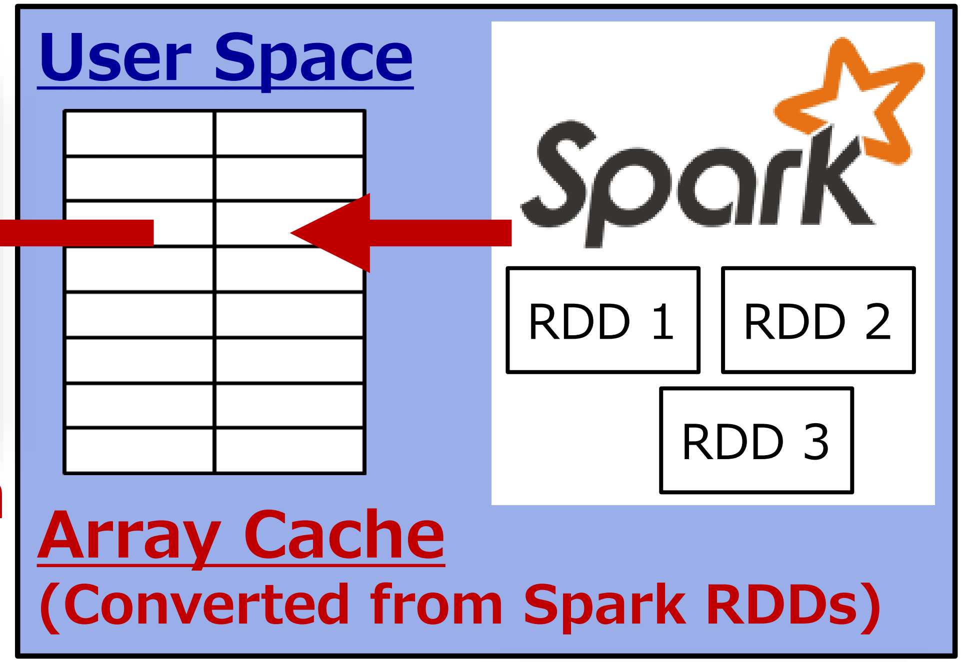


Data are stored in RDDs (i.e., distributed shared memory)
RDDs are converted to array structure & transferred to GPU

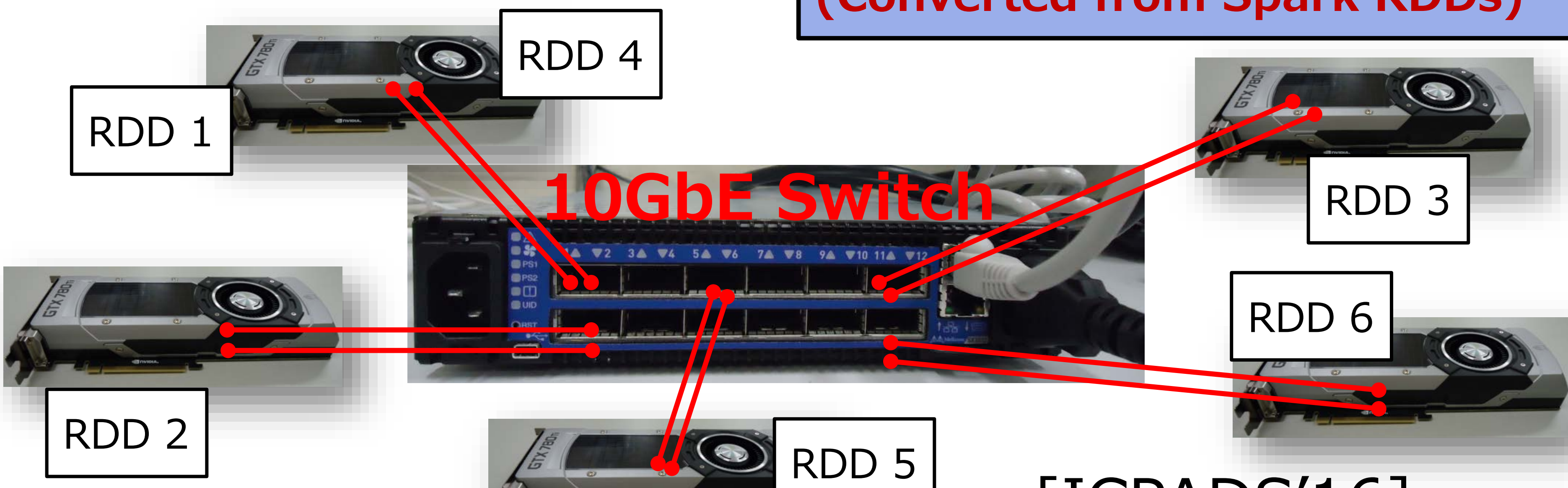
Data processing w/ Spark+GPUs



Data processing w/ Spark+GPUs

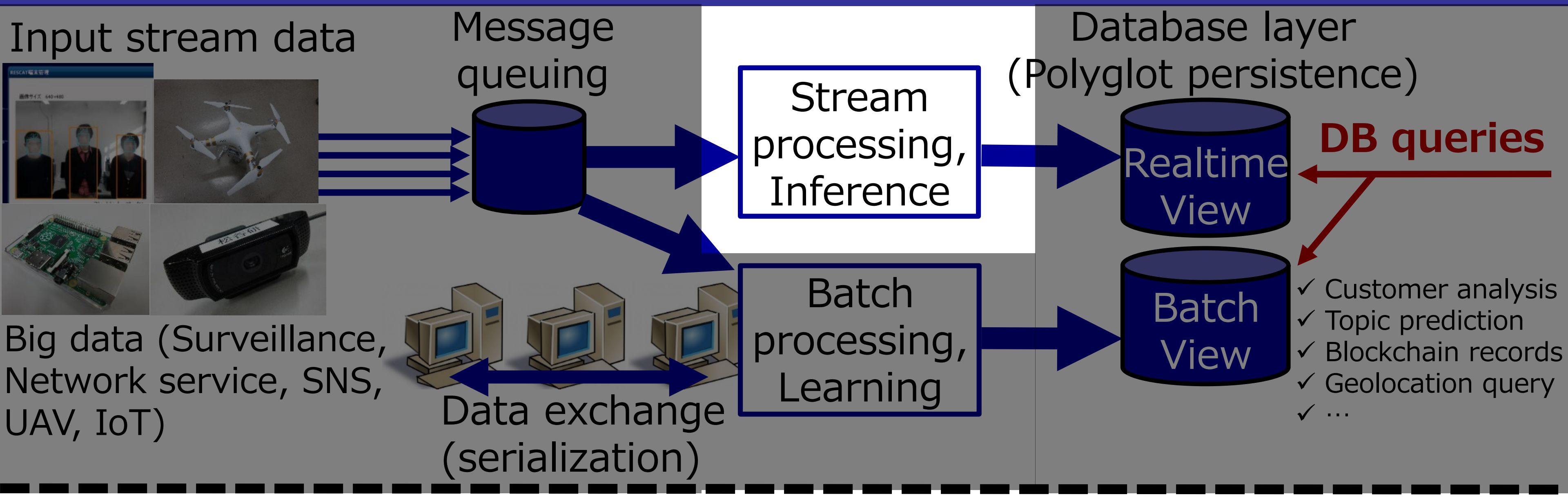


Reduction & transformation of RDDs offloaded to GPUs



[ICPADS'16]

Acceleration for data processing



I/O intensive ⟷ **Compute intensive**

Message queuing middleware (Apache Kafka)

Serialization (Apache Thrift)

Stream processing (Apache Spark Streaming)

KVS / Column DB (Redis, HBase)

Online machine learning (Classification, Outlier detection, Change point detection, Abnormal behavior detection)

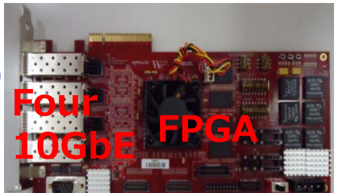
Document DB (MongoDB)

Graph DB (Neo4j), graph processing

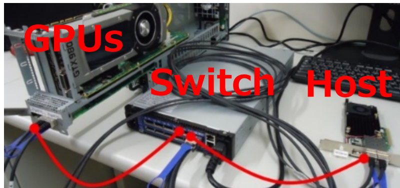
Batch processing (Apache Spark)

Machine learning (Apache Spark MLlib)

Tight integration of I/O and compute →FPGA



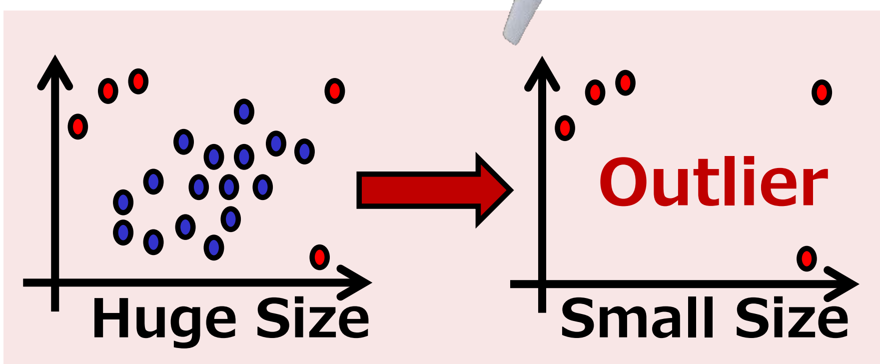
Massive parallelism → Networked GPU cluster



10Gbps outlier filtering: FPGA NIC

MPSoC16

Data Explosion



Machine learning algorithms

- ✓ Mahalanobis Distance
- ✓ **Local Outlier Factor(LOF)**
- ✓ **K-Nearest Neighbor(KNN)**

User Space

hadoop

Spark

NetFPGA-10G Device Driver

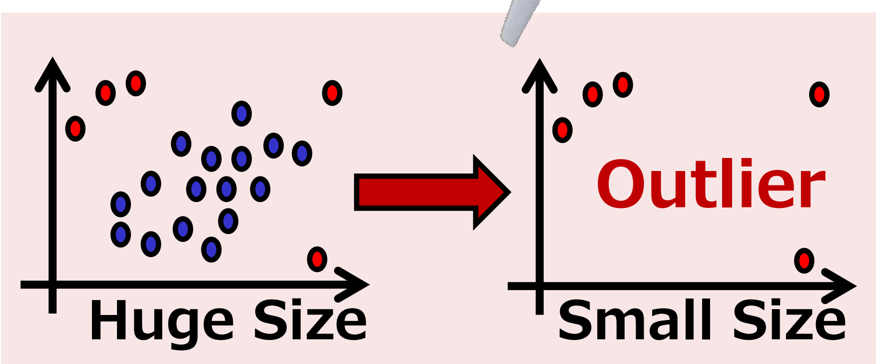
Only anomaly-valued packets are received

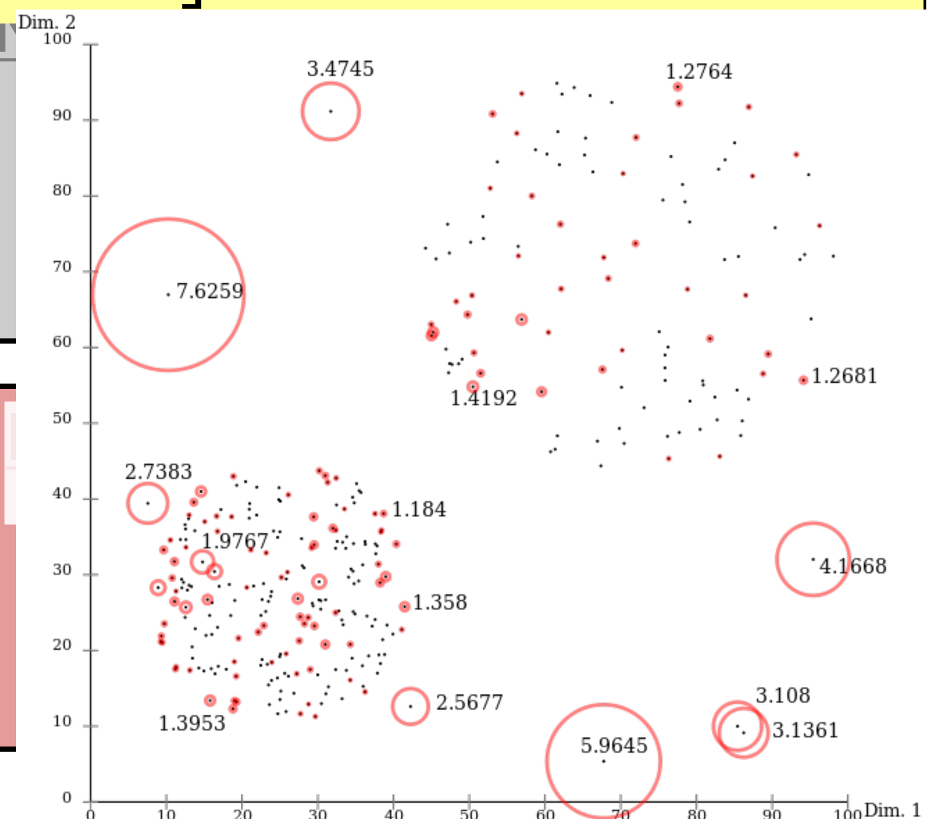
Data Mining

10GbE x4

10Gbps outlier filtering: FPGA NIC

Density-based approach to find outliers (e.g., higher LOF value when k neighbors are distant)
All reference data needed for density computation
→ Frequently-accessed reference data clusters are cached in FPGA NIC [PDP'17]



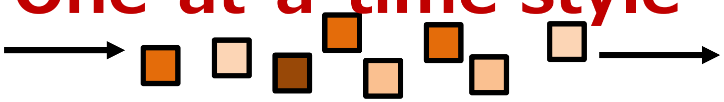


Machine learning algorithms
- ✓ Mahalanobis Distance
- ✓ **Local Outlier Factor(LOF)**
- ✓ **K-Nearest Neighbor(KNN)**

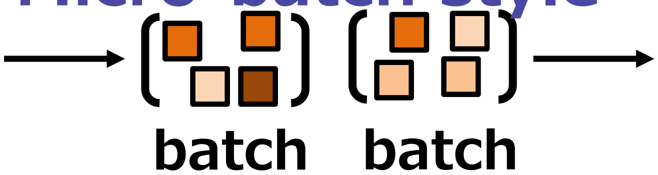
Spark Streaming: FPGA NIC

- Stream processing

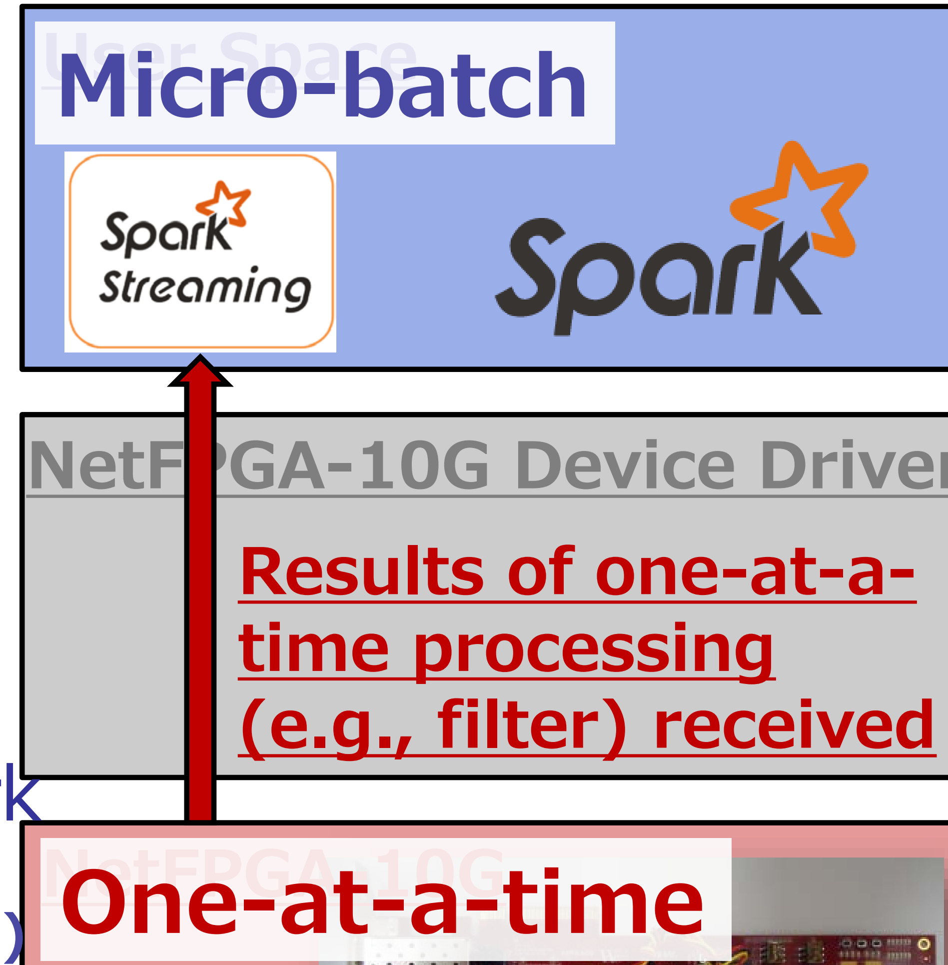
One-at-a-time style



Micro-batch style

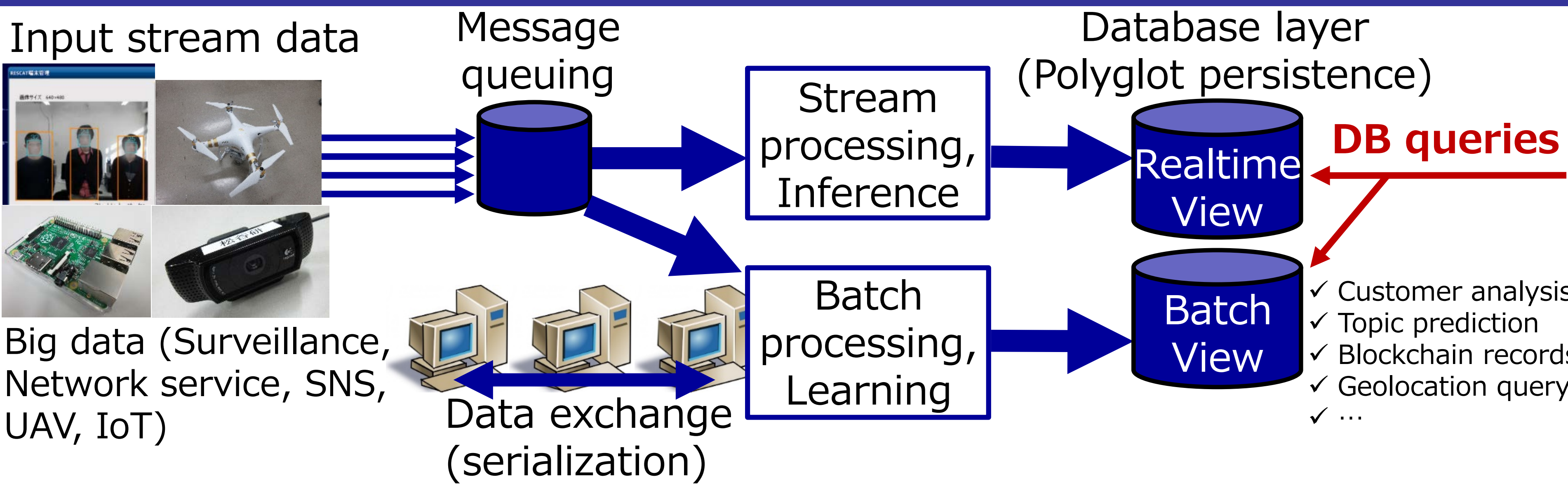


- Spark Streaming
  - Micro-batch style for compatibility w/ Spark
  - Large latency $_{\text{(e.g, 1sec)}}$



→ Stream processing components which can be executed as “one-at-a-time style” are offloaded to FPGA NIC [IEEE BigData WS’16]

Summary

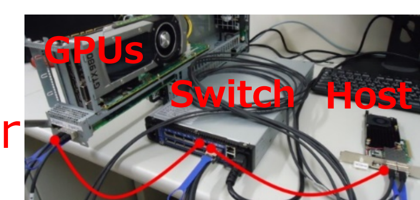


I/O intensive ⟷ **Compute intensive**

- Message queuing middleware (Apache Kafka)
- Serialization (Apache Thrift)
- Stream processing (Apache Spark Streaming)
- KVS / Column DB (Redis, HBase)
- Online machine learning (Classification, Outlier detection, Change point detection, Abnormal behavior detection)
- Document DB (MongoDB)
- Graph DB (Neo4j), graph processing
- Batch processing (Apache Spark)
- Machine learning (Apache Spark MLlib)

Tight integration of I/O and compute →FPGA

Massive parallelism → Networked GPU cluster



References (1/2)

- FPGA-based KVS accelerator
  - Yuta Tokusashi, et.al., "A Multilevel NOSQL Cache Design Combining In-NIC and In-Kernel Caches", IEEE Hot Interconnects 2016.
- FPGA-based Blockchain cache
  - Yuma Sakakibara, et.al., "An FPGA NIC Based Hardware Caching for Blockchain", HEART 2017.
- FPGA-based Spark Streaming accelerator
  - Kohei Nakamura, et.al., "An FPGA-Based Low-Latency Network Processing for Spark Streaming", IEEE BigData Workshops 2016.

References (2/2)

- FPGA-based machine learning accelerator
  - Ami Hayashi, et.al., "An FPGA-Based In-NIC Cache Approach for Lazy Learning Outlier Filtering", PDP 2017.
  - Ami Hayashi, et.al., "A Line Rate Outlier Filtering FPGA NIC using 10GbE Interface", ACM SIGARCH CAN (2015).
- GPU-based acceleration of Apache Spark
  - Yasuhiro Ohno, et.al., "Accelerating Spark RDD Operations with Local and Remote GPU Devices", ICPADS 2016.

Thank you for listening

Acknowledgement:
This work was supported by MIC SCOPE, NEDO IoT, JSPS Kaken B, and SECOM