Accelerator Design for Big Data Processing Frameworks

Hiroki Matsutani
Dept. of ICS, Keio University
http://www.arc.ics.keio.ac.jp/~matutani
Batch processing requires time and thus recent data cannot be reflected to the analysis result. 
→ Combine batch and stream processing to make up for the realtime capability.
Batch + Stream processing

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

Input stream data Message queuing Data exchange (serialization)

Database layer (Polyglot persistence)

Stream processing, Inference

Realtime View

Batch View

Data exchange

• Customer analysis
• Topic prediction
• Blockchain records
• Geolocation query
• ...

Batch processing, Learning

Batch + Stream processing

Input stream data

Message queuing

Database layer (Polyglot persistence)

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

Batch processing, Learning

Batch View

Stream processing, Inference

Realtime View

DB queries

 ✓ Customer analysis
 ✓ Topic prediction
 ✓ Blockchain records
 ✓ Geolocation query
 ✓ ...

I/O 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)

Compute intensive

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 \( \rightarrow \) FPGA

Massive parallelism \( \rightarrow \) Networked GPU cluster

FPGA

GPUs

Switch Host

Four 10GbE
Acceleration for data store

Input stream data

Message queuing

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

Data exchange (serialization)

Database layer
(Polyglot persistence)

Stream processing, Inference

DB queries

Realtime View

Customer analysis

Topic prediction

Blockchain records

Geolocation query

... 

Batch View

Batch processing, Learning

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

I/O intensive

Compute intensive

Message queuing middleware (Apache Kafka)

Stream processing (Apache Spark Streaming)

Online machine learning (Apache Spark MLlib)

Document DB (MongoDB)

Batch processing (Apache Spark)

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

GPUs

Switch

Host

Four 10GbE

FPGA
Multilevel NOSQL cache: FPGA NIC

[IEEE HoTI’16]

Normal DB Access

Request

In-Kernel KVS Cache (L2$) Hit

Add

Result

NIC Cache (L1$) Hit

Reply

User Space

mongoDB

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:
- 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]
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**

2. TX(Bob→Alice) is represented as a block

3. The block is broadcasted to all the nodes

4. After verified, the block is added to the blockchain
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

Input stream data

Message queuing

Data exchange (serialization)

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

Database layer (Polyglot persistence)

DB queries

- Customer analysis
- Topic prediction
- Blockchain records
- Geolocation query
- ...

I/O intensive

Message queuing middleware (Apache Kafka)

Stream processing (Apache Spark Streaming)

KVS / Column DB (Redis, HBase)

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

Batch processing (Apache Spark)

Document DB (MongoDB)

Graph DB (Neo4j), graph processing

Machine learning (Apache Spark MLlib)

Massive parallelism → Networked GPU cluster

Tight integration of I/O and compute → FPGA

Compute intensive

Four 10GbE FPGA

GPUs

Switch

Host
Data processing w/ Spark+GPU

Array Cache (RDDs)

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
Many GPUs are directly connected to Apache Spark server via NEC ExpEther (20Gbps)

Data processing w/ Spark+GPUs
Data processing w/ Spark+GPUs

Array Cache (RDDs)

Reduction & transformation of RDDs offloaded to GPUs

User Space

Array Cache (Converted from Spark RDDs)

10GbE Switch

[ICPADS’16]
Input stream data

Message queuing

Database layer (Polyglot persistence)

Realtime View

Batch View

Database layer

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

Message queuing middleware (Apache Kafka)

Stream processing (Apache Spark Streaming)

Document DB (MongoDB)

Stream processing, Inference

Batch processing, Learning

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

Graph DB (Neo4j), graph processing

Machine learning (Apache Spark MLlib)

Tight integration of I/O and compute → FPGA

Four 10GbE FPGA

Massive parallelism → Networked GPU cluster

Database layer

Batch processing (Apache Spark)

Message queuing (Apache Kafka)

Stream processing (Apache Spark Streaming)

Document DB (MongoDB)

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

Graph DB (Neo4j), graph processing

Machine learning (Apache Spark MLlib)

Tight integration of I/O and compute → FPGA

Four 10GbE FPGA

Massive parallelism → Networked GPU cluster

Transfer of information: Message queuing

Transfer of information: Data exchange (serialization)

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

Client servers: I/O intensive

Client servers: Compute intensive

I/O intensive

I/O intensive

Compute intensive

Compute intensive
10Gbps outlier filtering: FPGA NIC

Data Explosion

User Space

NetFPGA-10G Device Driver

Only anomaly-valued packets are received

Data Mining

Machine learning algorithms

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

**Input stream data**

- Message queuing

**Message queuing**

**Database layer** (Polyglot persistence)

- Stream processing, Inference
- Batch processing, Learning

**DB queries**

- Customer analysis
- Topic prediction
- Blockchain records
- Geolocation query
- ...

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

- I/O intensive
- Compute intensive

**I/O intensive**

- Message queuing middleware (Apache Kafka)
- Stream processing (Apache Spark Streaming)
- KVS / Column DB (Redis, HBase)

**Compute intensive**

- 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**

- **Four 10GbE FPGA**

- **GPUs**
- **Switch**
- **Host**
• FPGA-based KVS accelerator

• FPGA-based Blockchain cache

• FPGA-based Spark Streaming accelerator
• FPGA-based machine learning accelerator

• GPU-based acceleration of Apache Spark
Acknowledgement:
This work was supported by MIC SCOPE, NEDO IoT, JSPS Kaken B, and SECOM