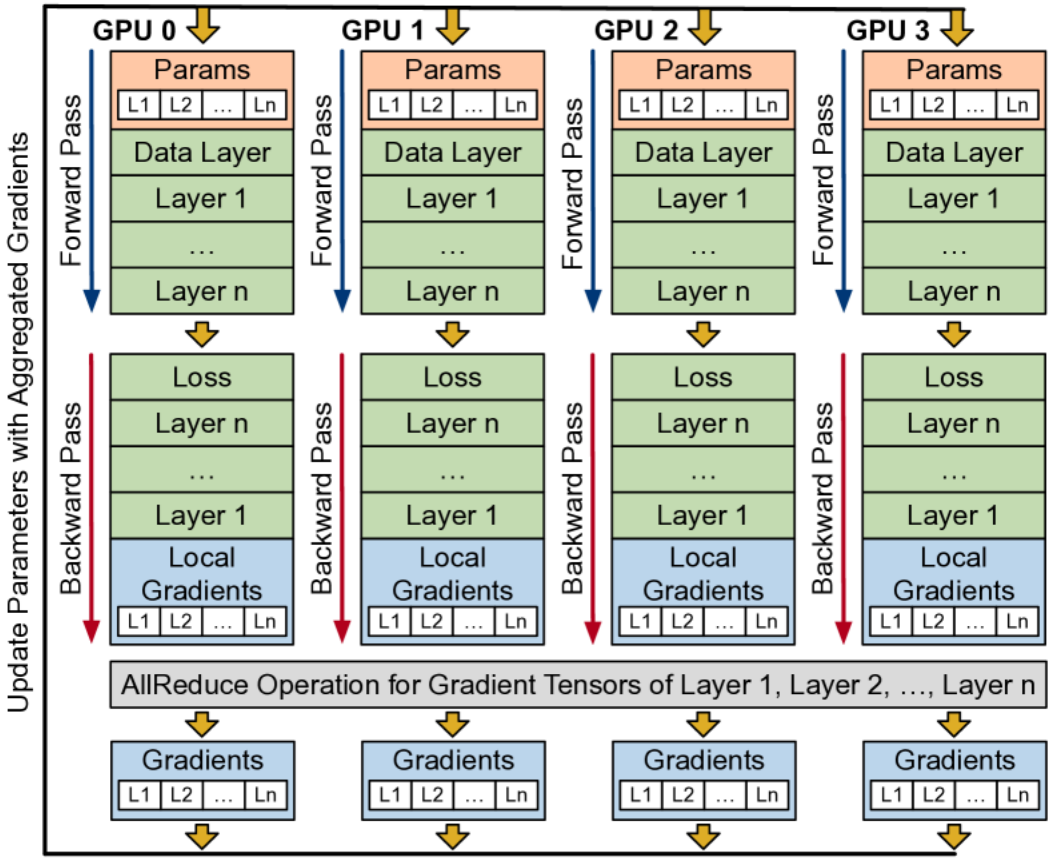
OmNICreduce-FPGA: Offload Sparse Collective Communication to FPGA-based SmartNIC

T. Gu

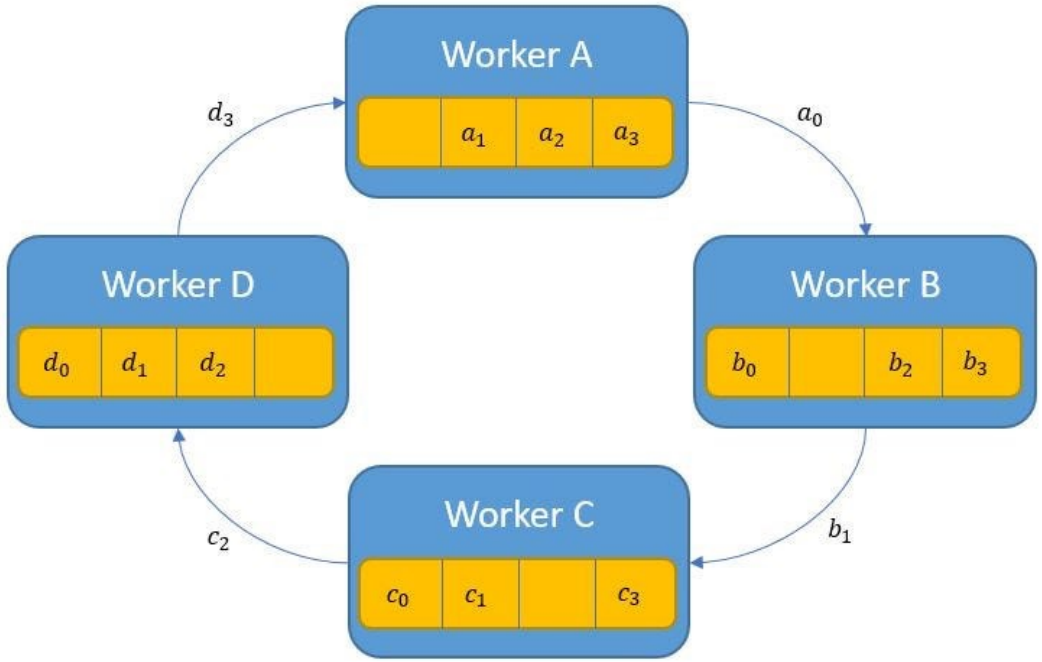
Motivation

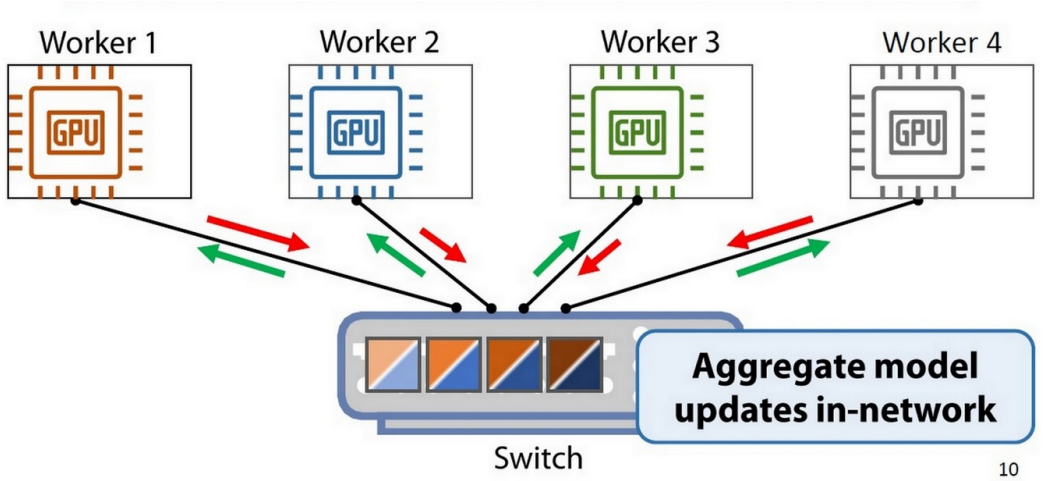
- AllReduce is commonly used by ML and HPC workloads
  - Exchange gradients for Data Parallelism in Deep Learning
- Gradients of certain models are highly sparse



Motivation

- Two AllReduce Algorithms
  - Ring AllReduce
    - Hard to leverage sparsity
  - In-Network Streaming AllReduce
    - Improve performance
      - Scalability
      - Throughput
      - Latency
    - Possible to leverage sparsity
    - Requires additional hardware
      - FPGA is efficient





Related Works

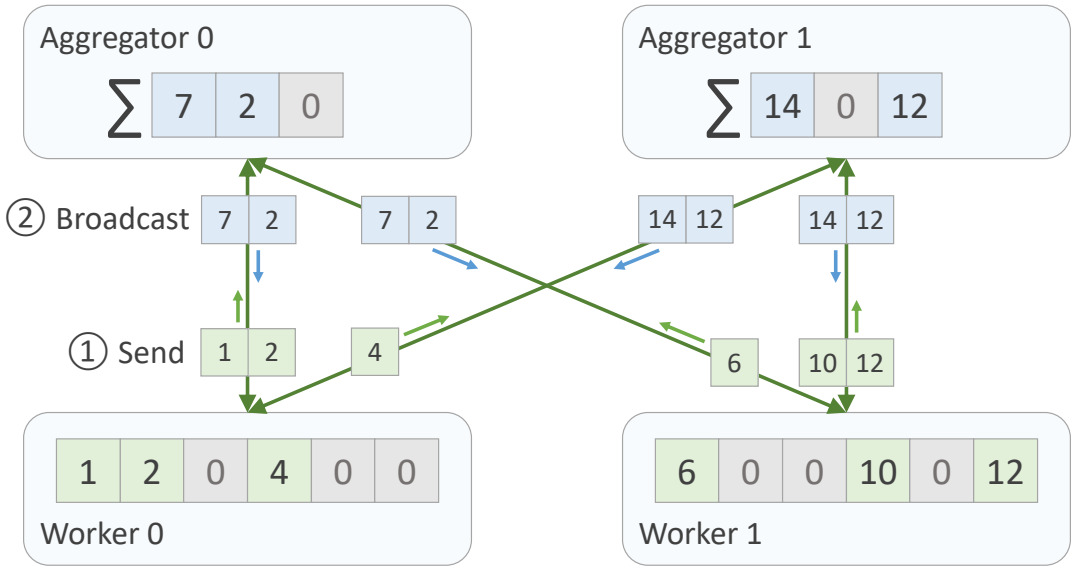
- Hardware-Offloaded Ring Allreduce:
  - MVAPICH2-DPU, DOCA: Commercial Solution on SoC-based SmartNICs
    - Improve overlapping rate of non-blocking AllReduce
- In-Network Streaming AllReduce
  - Mellanox SHARP: Commercial Solution on Proprietary Switches
  - SwitchML (NSDI'21): Impl. on P4 Programmable Switches
    - Propose Streaming Aggregation Algorithm
  - ATP (NSDI'21), A2TP (EuroSys'23): Add multi-tenant support
  - Flare (SC'21), NetReduce (ASPLOS'23): Impl on FPGA-based Switches

Related Works

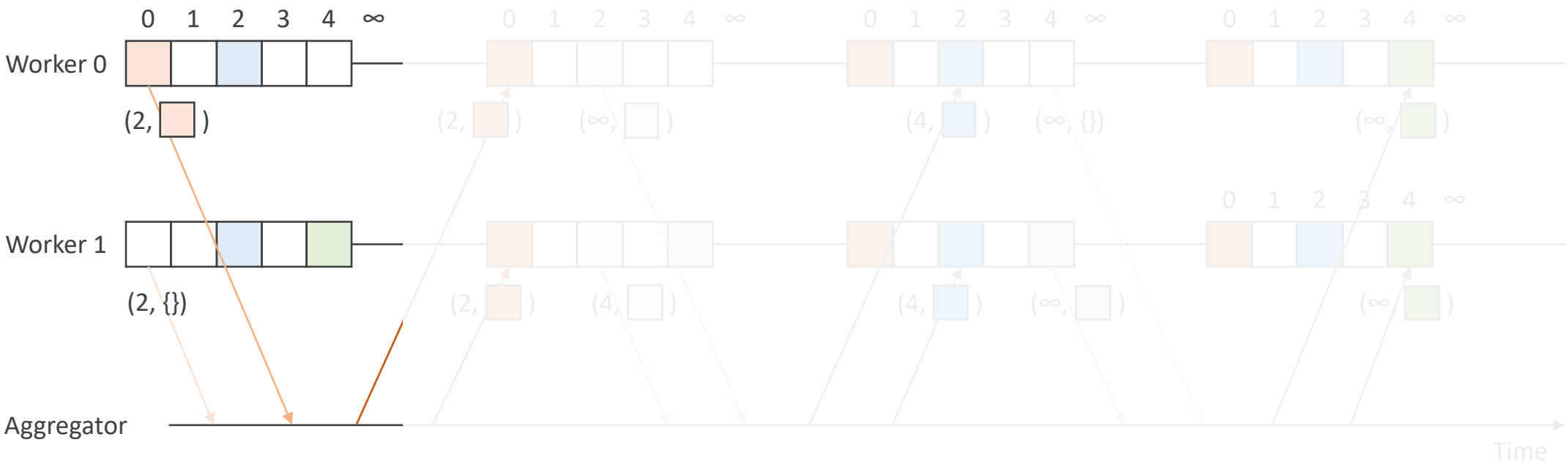
- Sparse In-Network Streaming AllReduce
  - OmniReduce (SIGCOMM'21):
    - OmniReduce Algorithm: Impl. on Linux servers with RoCE Network
    - OmniReduce-P4 Algorithm: Impl. on P4 Programmable Switches
  - OmNICreduce (Ongoing): Impl. on SoC-based SmartNICs

OmNICreduce-FPGA

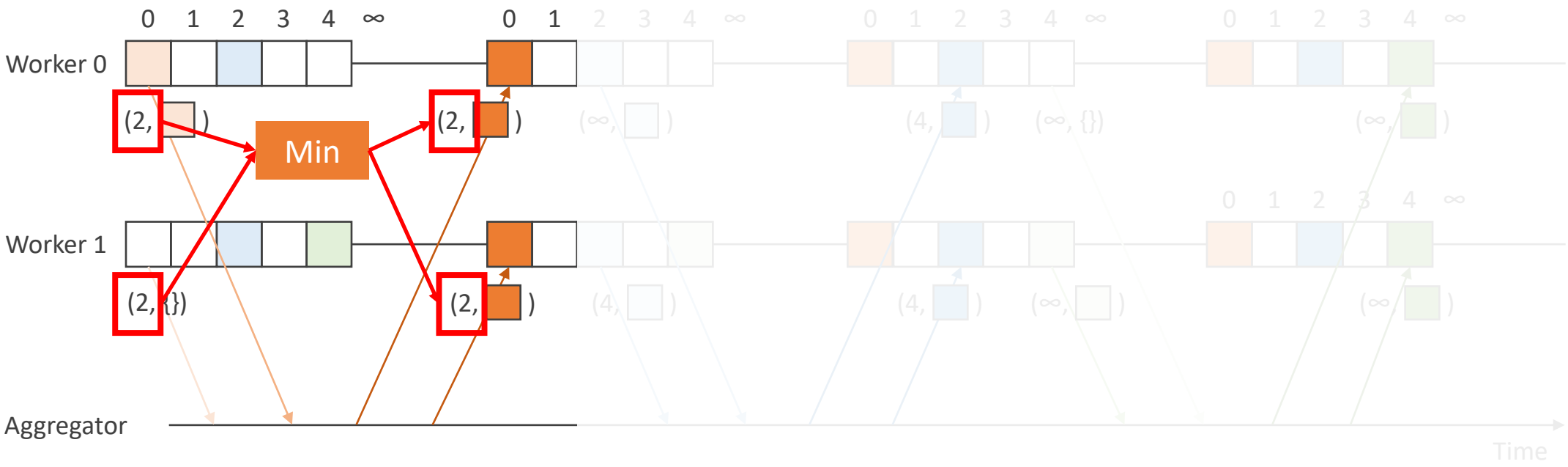
- Implement OmniReduce-P4 Algorithm on FPGA accelerators
  - OmniReduce-P4 was proposed alongside OmniReduce Algorithm
    - A greatly simplified version
    - Originally runs on Programmable Switches
  - Utilize FPGAs as aggregators
    - Efficient than x86 servers



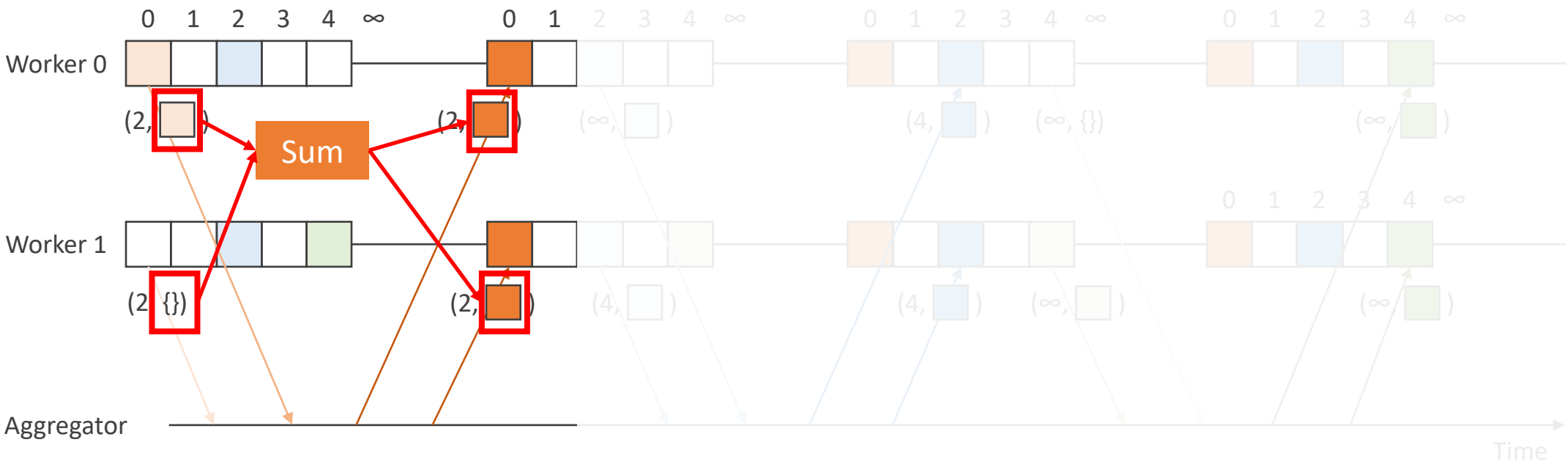
OmniReduce-P4 Algorithm



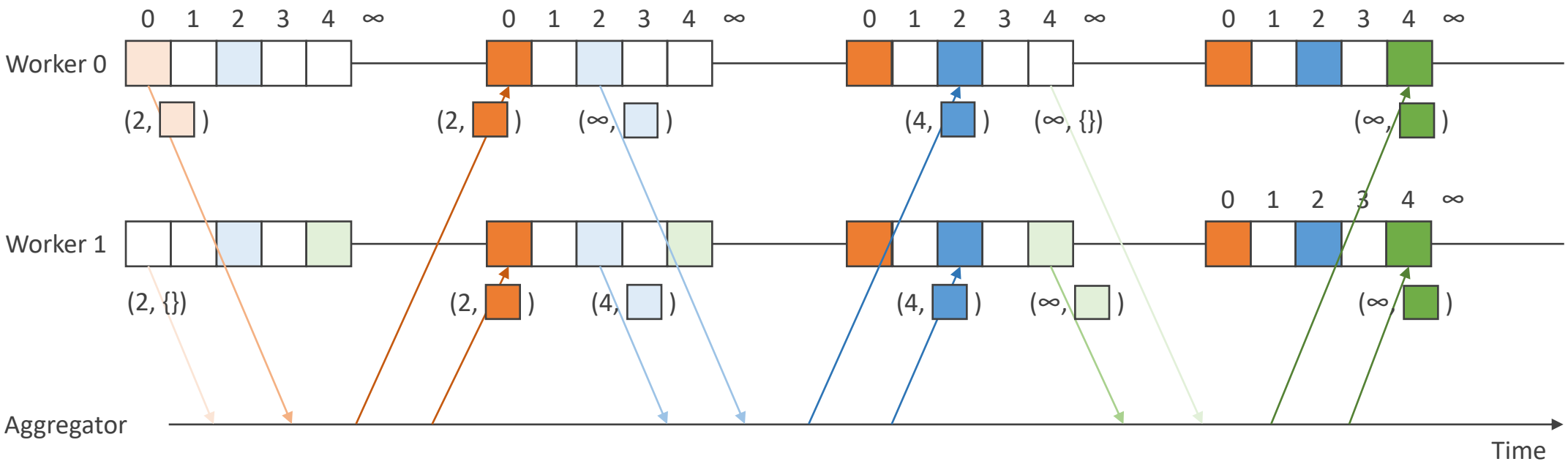
OmniReduce-P4 Algorithm



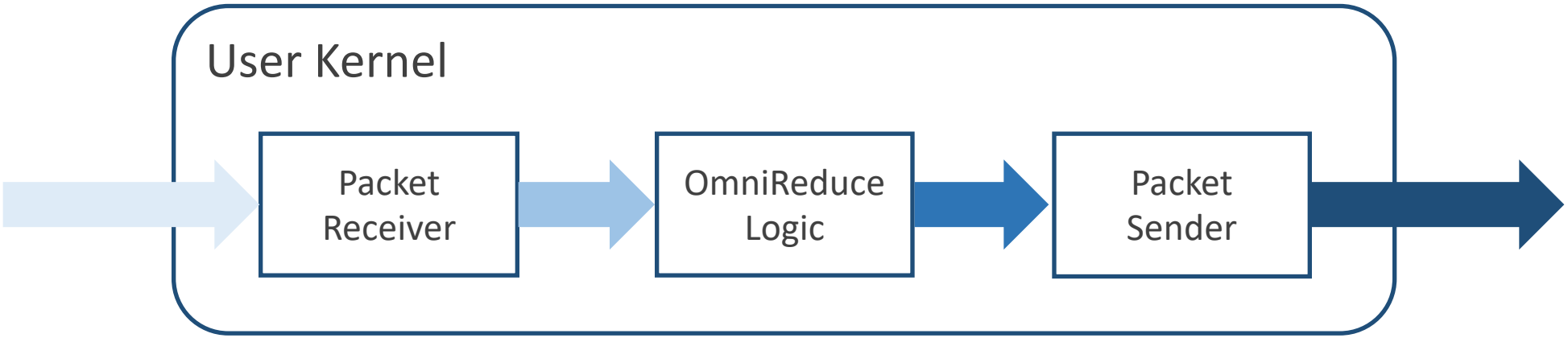
OmniReduce-P4 Algorithm



OmniReduce-P4 Algorithm

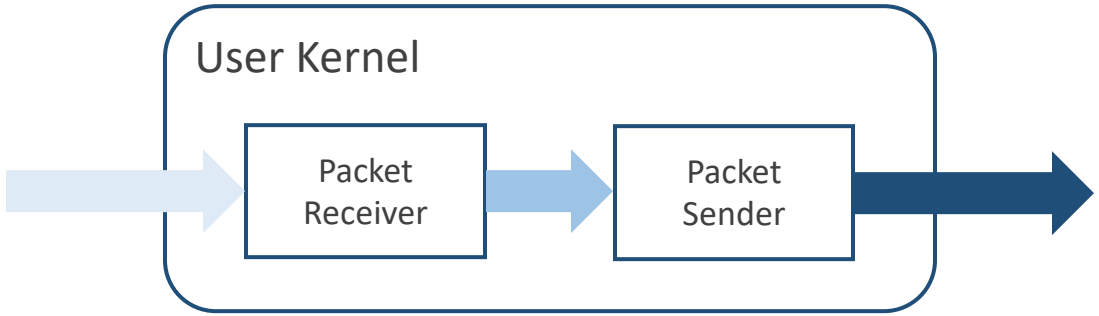


Trivial Design



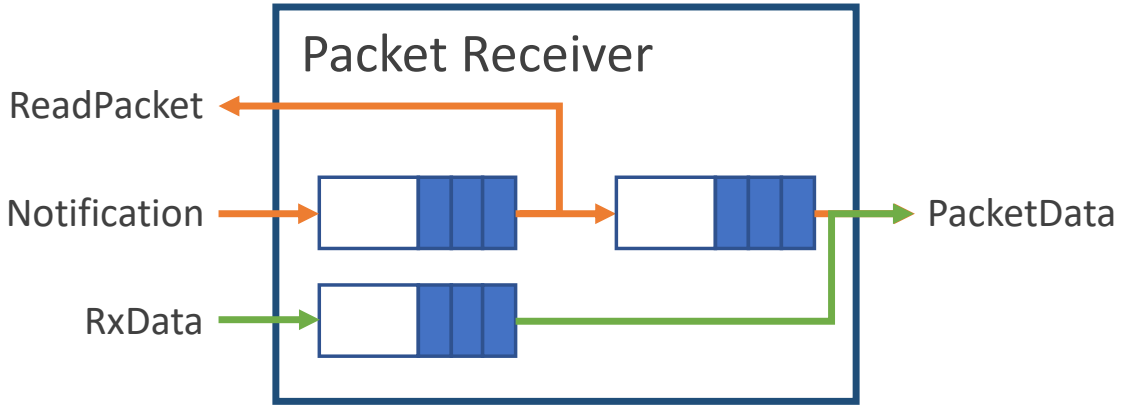
TCP Echo

- Echo is the simplest user kernel
  - Firstly implemented
- Built upon EasyNet Vitis 100Gbps TCP/IP Stack
  - Lacks documentation
  - Need to figure out handshake logics by ourselves



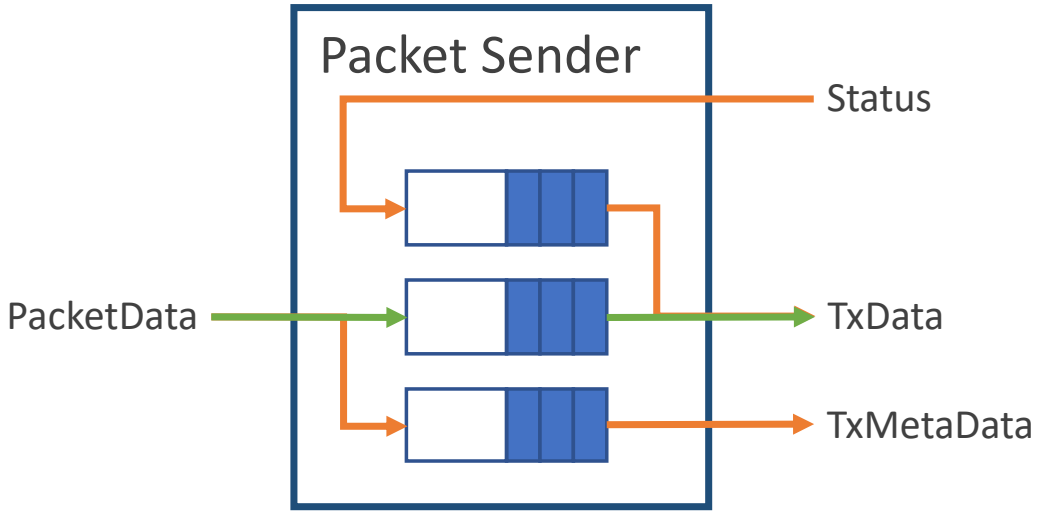
Handshake of Receiving TCP Packets

- Procedure
  - Framework → User Krnl: Notification
    - Conn ID, IP Addr, Src Port
  - User Krnl → Framework: ReadPkt
    - Conn ID
  - Framework → User Krnl: RxData
    - Payload
- Exception:
  - Close connection notification



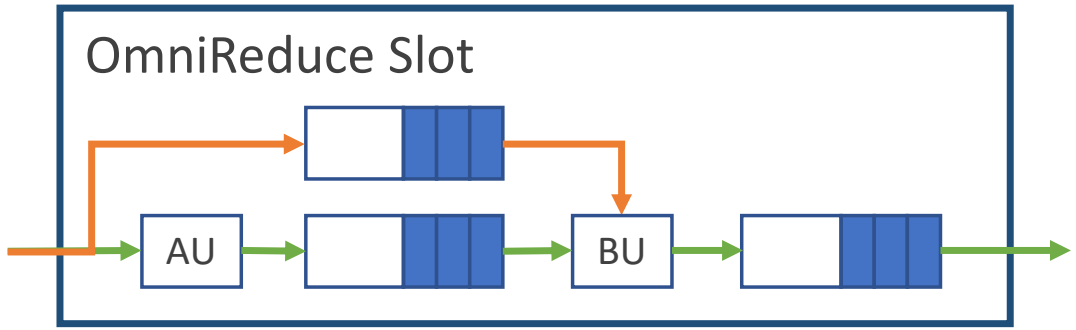
Handshake of Sending TCP Packets

- Procedure
  - User Krnl → Framework : TxMetaData
    - Conn ID, Packet Size
  - Framework → User Krnl: Status
    - Error code
  - User Krnl → Framework: TxData
    - Payload
- Exception:
  - Send to a closed connection

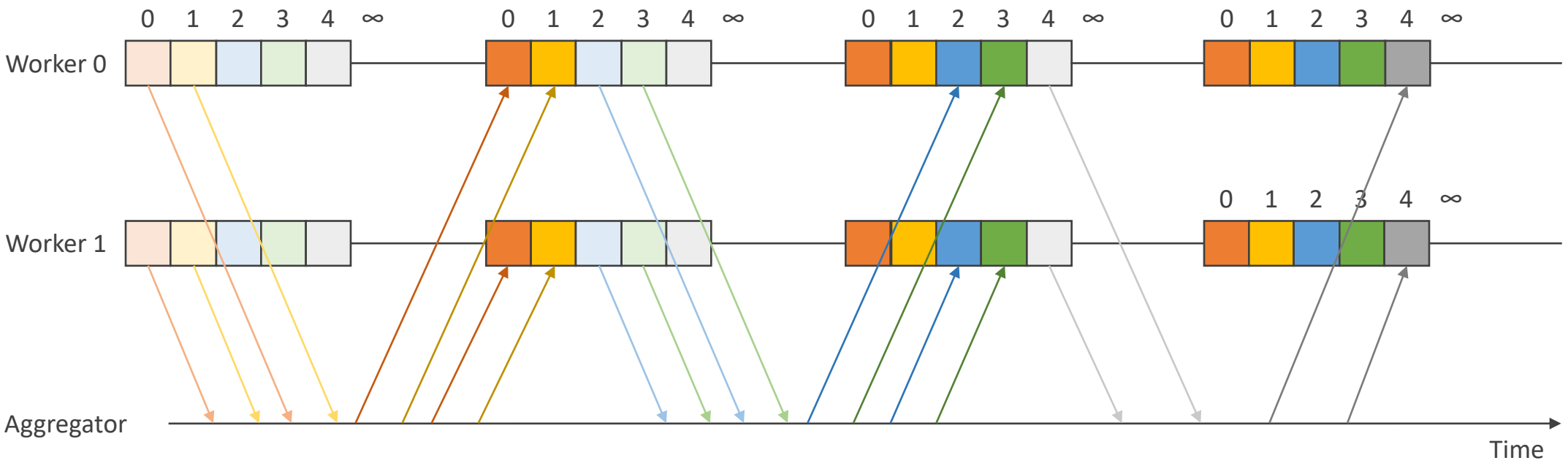


OmniReduce Slot

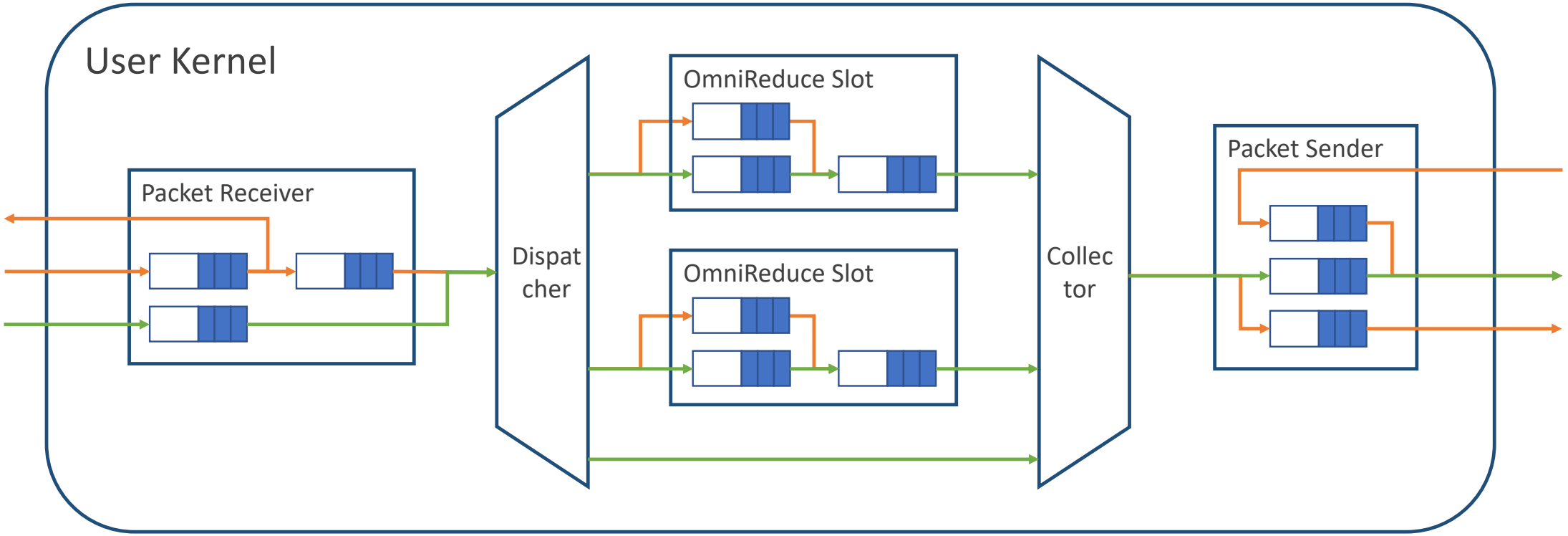
- Aggregation Unit (AU)
  - 480-bit Accumulator
    - Cumulate block values
  - 32-bit Comparator
    - Find minimal next block ID
- Broadcast Unit (BU)
  - Broadcast results to all workers
    - Connection IDs are stored in FIFO



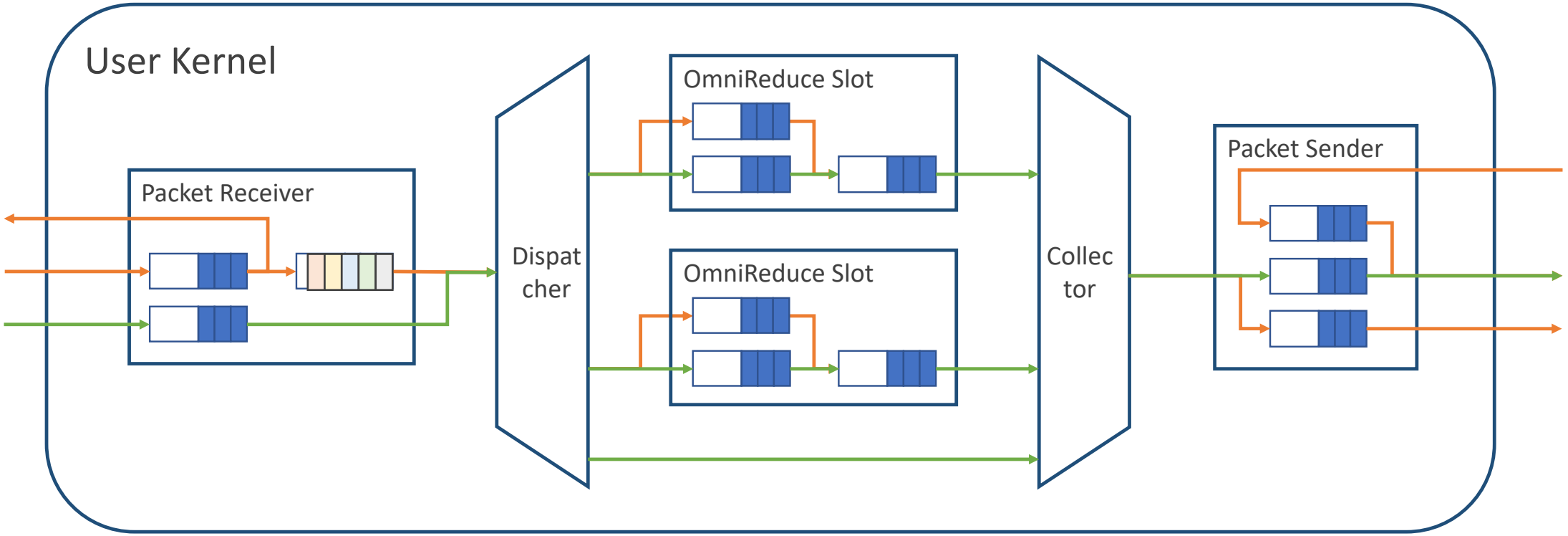
Concurrent Aggregation



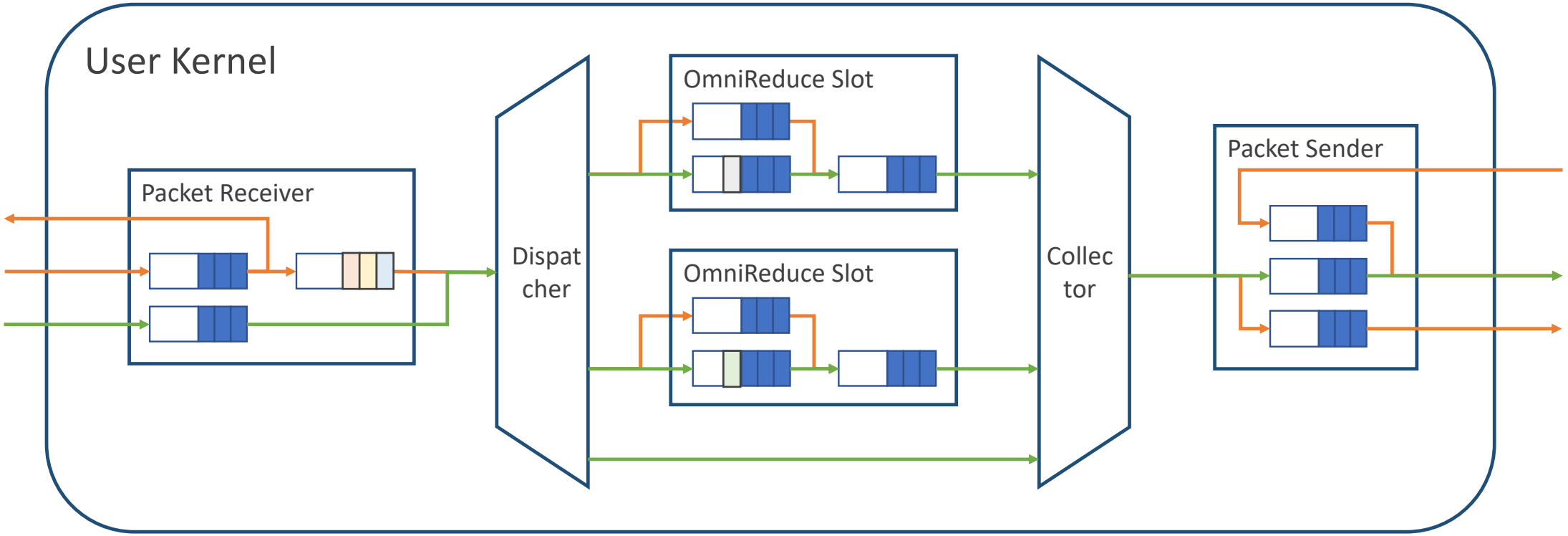
Hardware Architecture



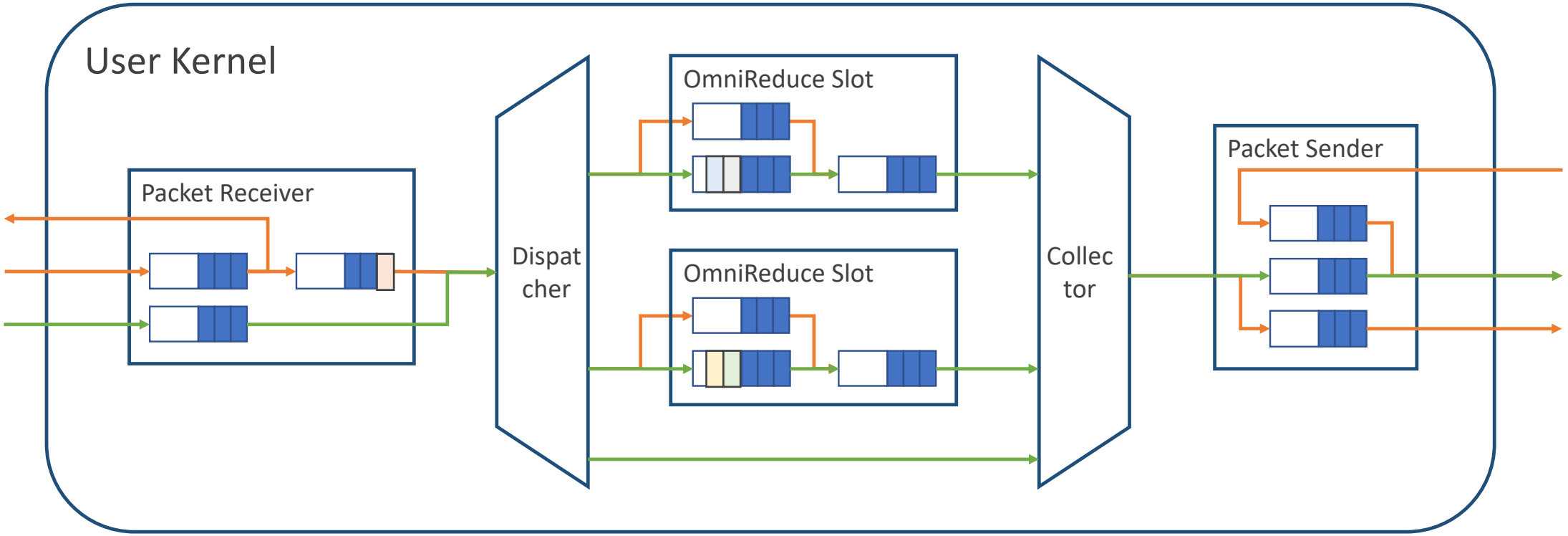
Hardware Architecture



Hardware Architecture



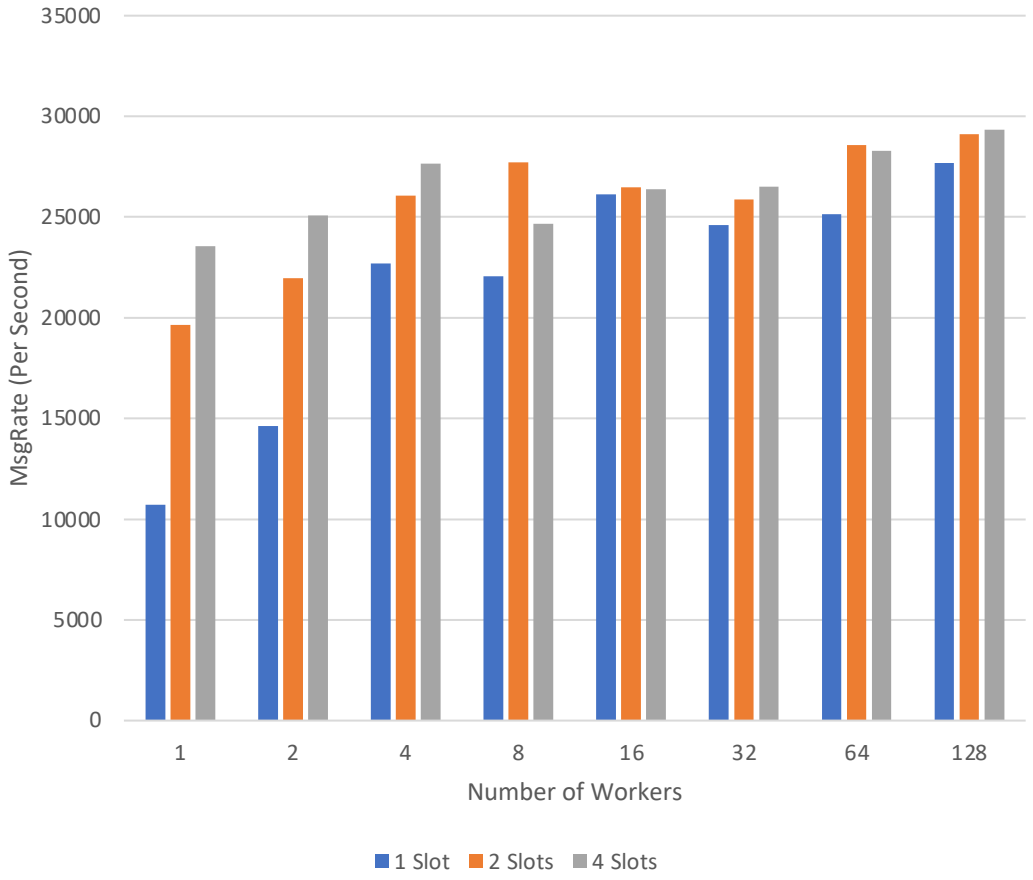
Hardware Architecture



Results

- Specifications of Testbed:
  - CPU: AMD Ryzen Threadripper PRO 3975WX 32-Cores
  - NIC: Mellanox ConnectX-6 100Gbps
  - FPGA: Xilinx Alveo U280 Data Center Accelerator Card
- Software
  - Implemented with Python Async.io
  - Software Aggregator: Fully functional
  - Software Worker: Send mock packets

Results


35000
30000
25000
20000
15000
10000
5000
0
MsgRate (Per Second)
1
2
4
8
16
32
64
128
Number of Workers
1 Slot
2 Slots
4 Slots


Latency of processing 100 messages per worker per slot, 4 slots
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
Latency (Unit: s)
1
2
4
8
16
32
64
128
Number of Workers
FPGA Aggregator
x86 Aggregator

Conclusions

- OmNICreduce-FPGA could deliver stable and comparable performance compared to Software Aggregators
- OmNICreduce-FPGA is capable of handling up to 512 concurrent TCP connections
- Multi-Slot design brings performance improvement
  - When number of workers is small

Future Works

- Allow aggregators to handle large packets (>1024 Bytes)
  - Currently packet size is restricted to 64 Bytes
- Move to RDMA Network Stack
  - TCP/IP stack incurs a huge overhead