

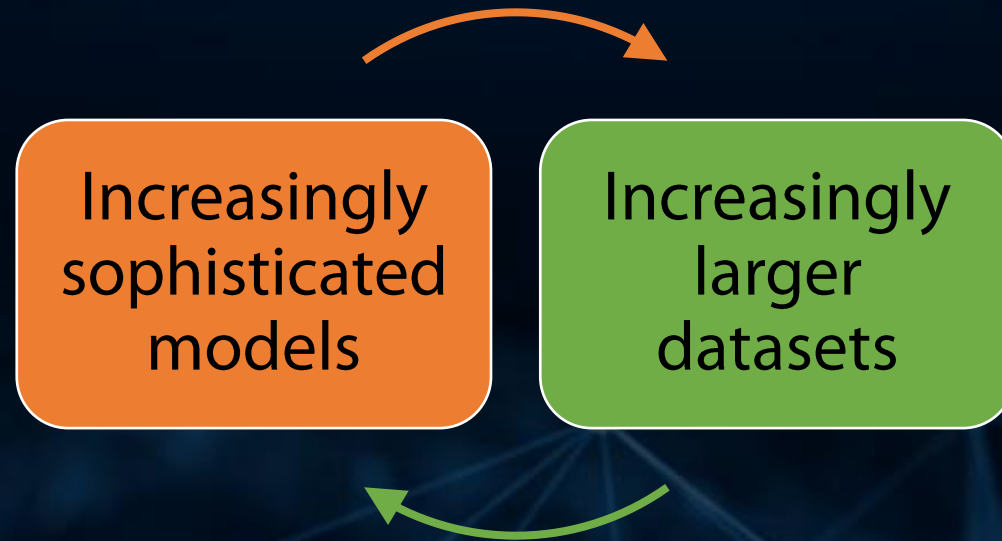
# SwitchML

## Scaling Distributed Machine Learning with In-Network Aggregation

**Amedeo Sapiro**, Marco Canini, Chen-Yu Ho, Jacob Nelson, Panos Kalnis,  
Changhoon Kim, Arvind Krishnamurthy, Masoud Moshref, Dan Ports, Peter Richtarik



# Machine Learning



Innovation fueled by leaps in (costly) infrastructure:

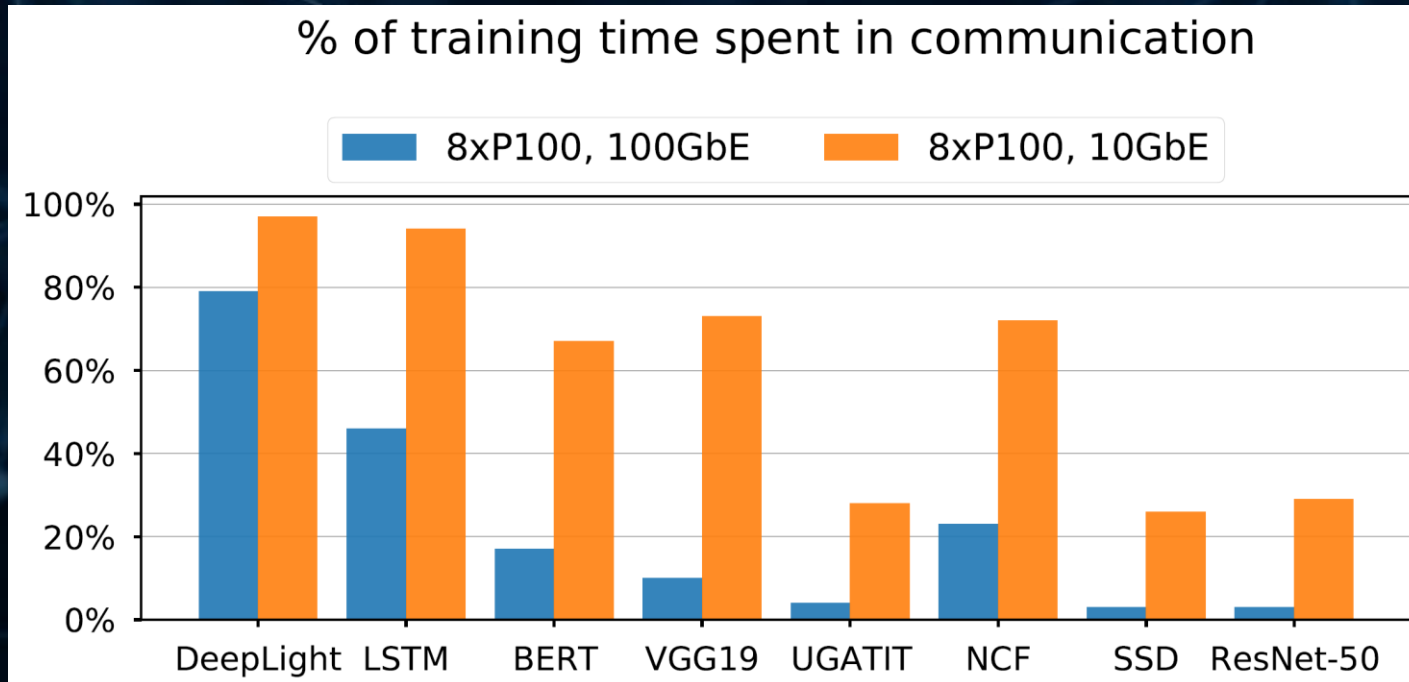
**Clusters with hundreds of machines,  
each with many HW accelerators (GPUs)**

Compute requirements **doubling every 3 months!**

Training models is still **very time-consuming**: days or even weeks!

# Scaling Machine Learning

Make **efficient** use of combined  
resources at multiple worker nodes



Can the network be  
the ML accelerator?

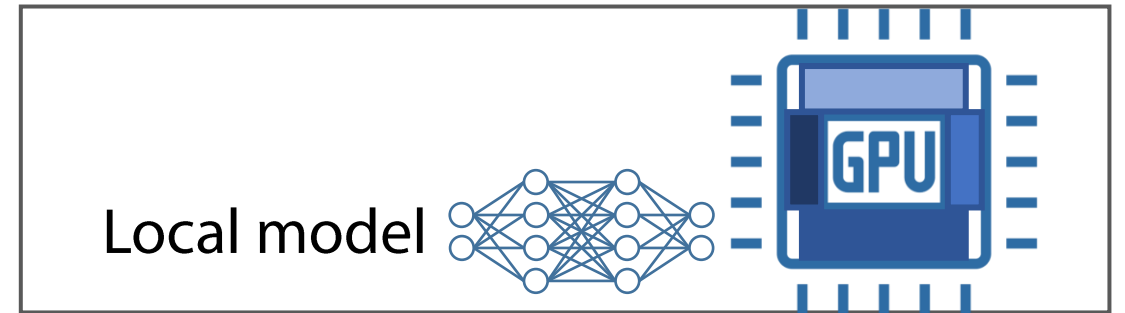
# Distributed ML (data-parallel)

---

Worker 1



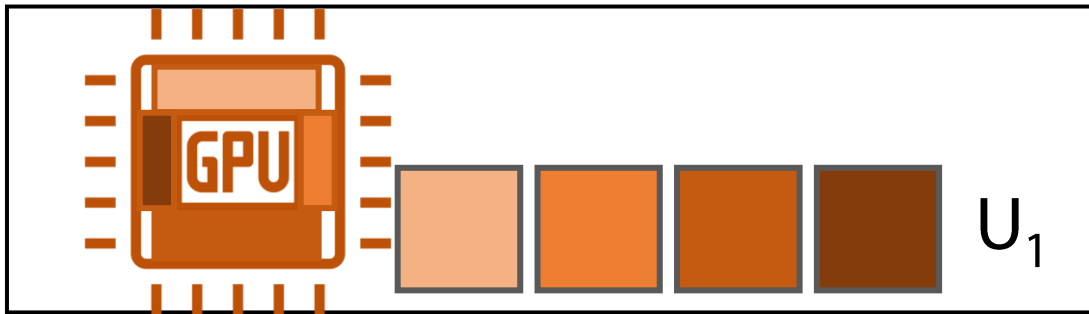
Worker 2



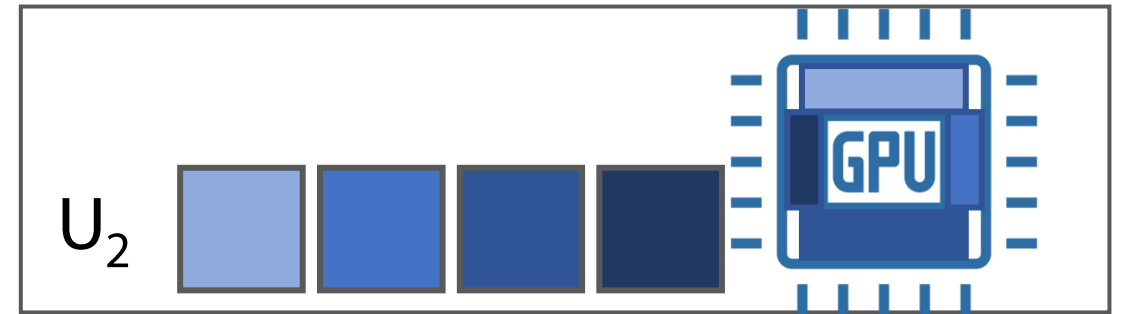
# Distributed ML (data-parallel)

---

Worker 1

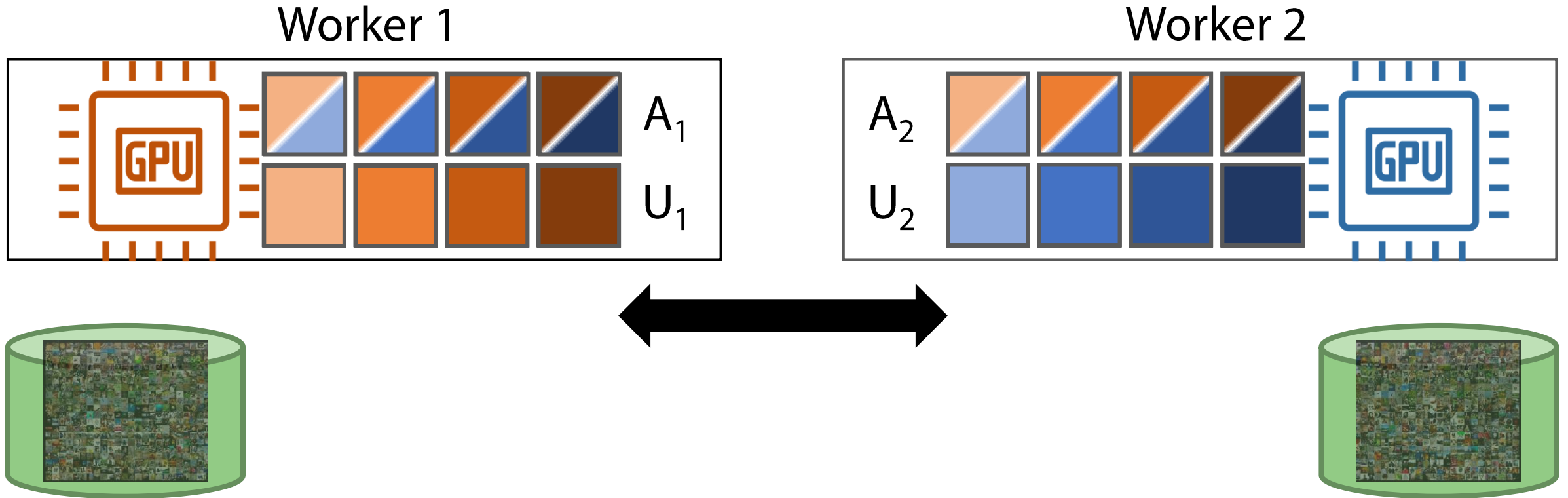


Worker 2

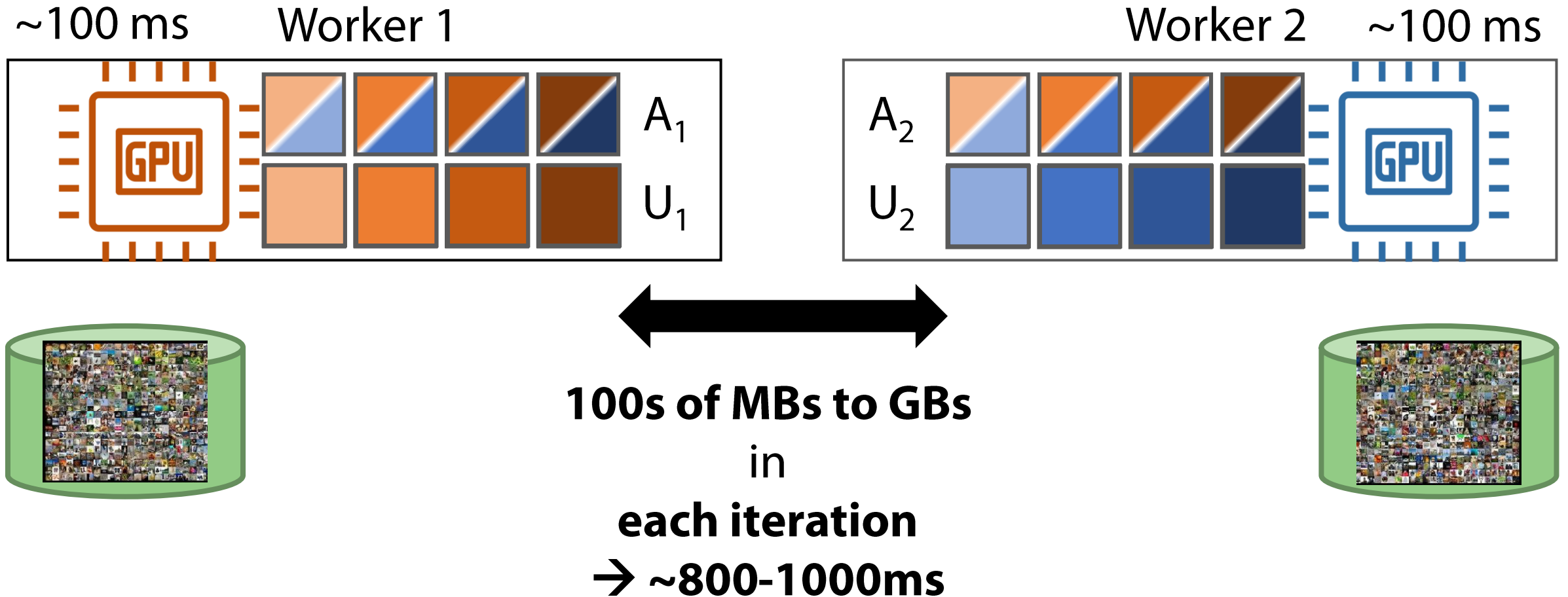


# Distributed ML (data-parallel)

---

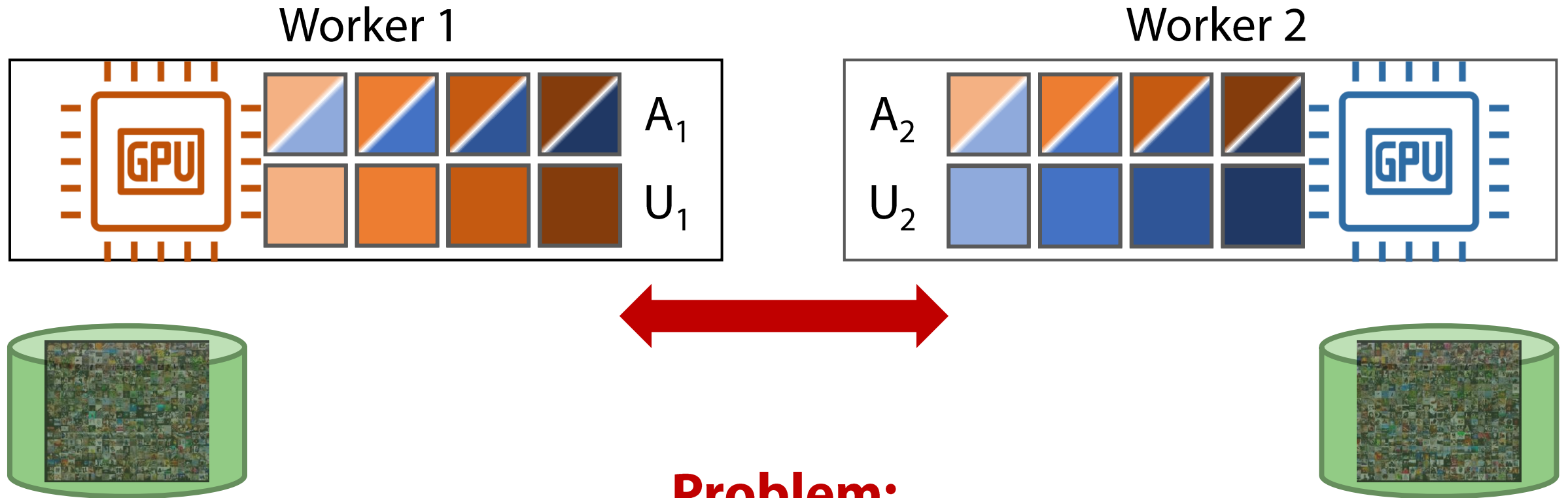


# Distributed ML (data-parallel)



# Distributed ML (data-parallel)

---



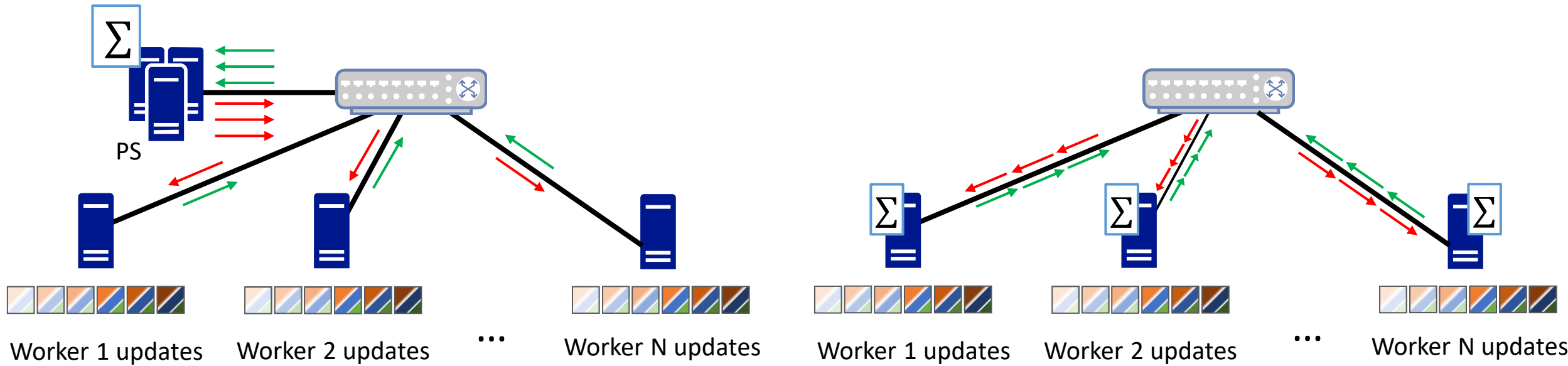
**Problem:**

**Very intensive communication in all-to-all fashion!  
Network increasingly the bottleneck to training speed**

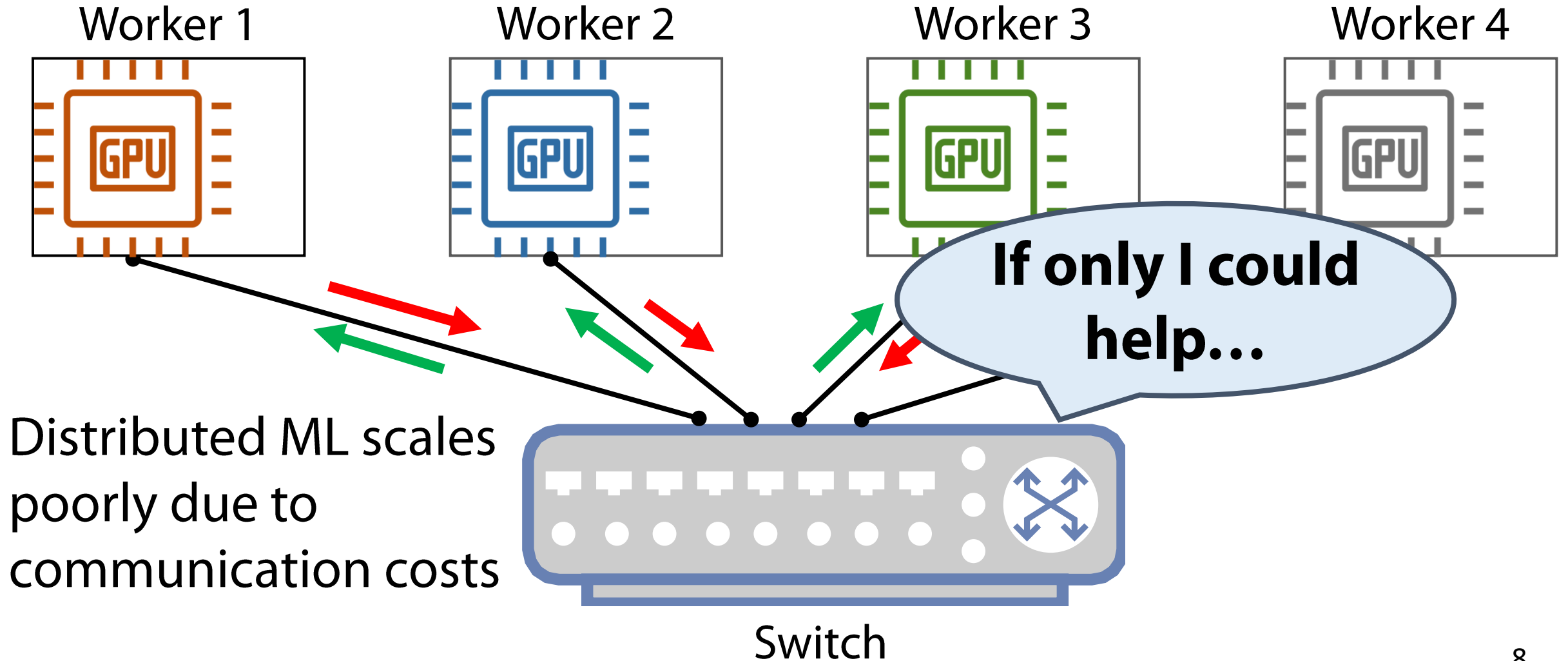


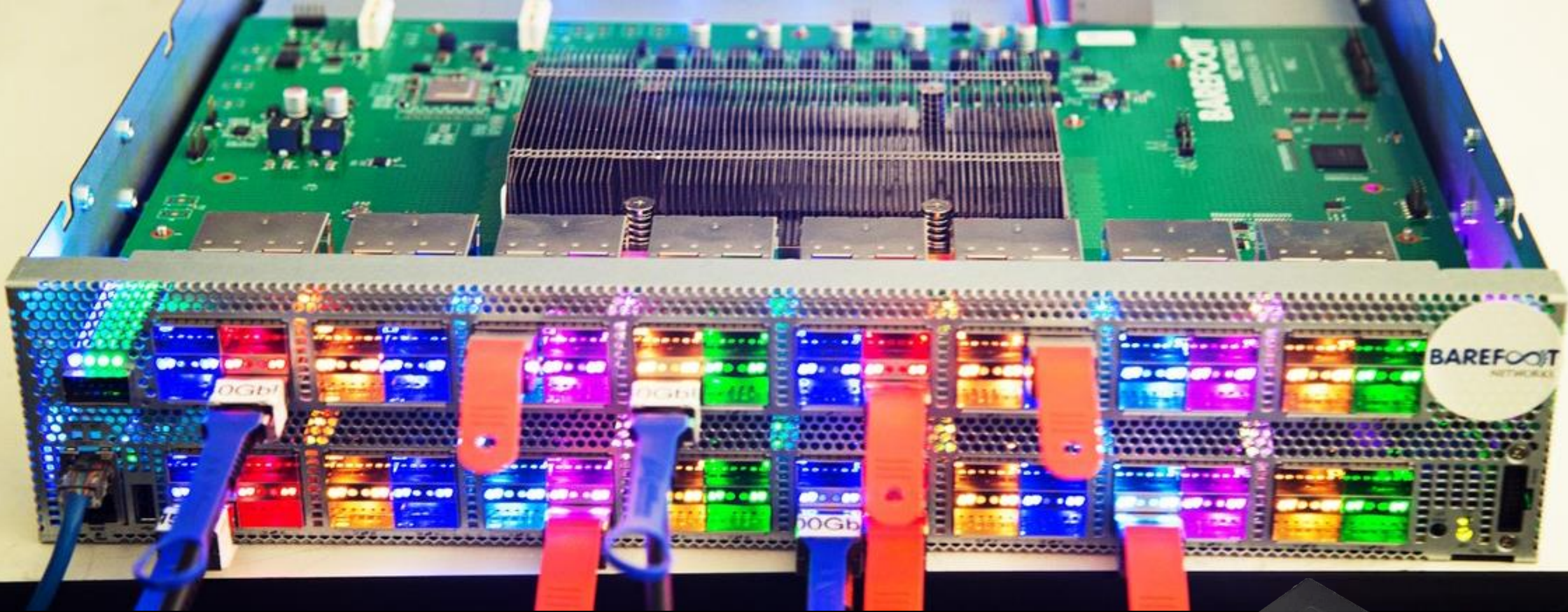
# Parameter Server (PS) and All-Reduce (ring)

---



# A closer look at model synchronization





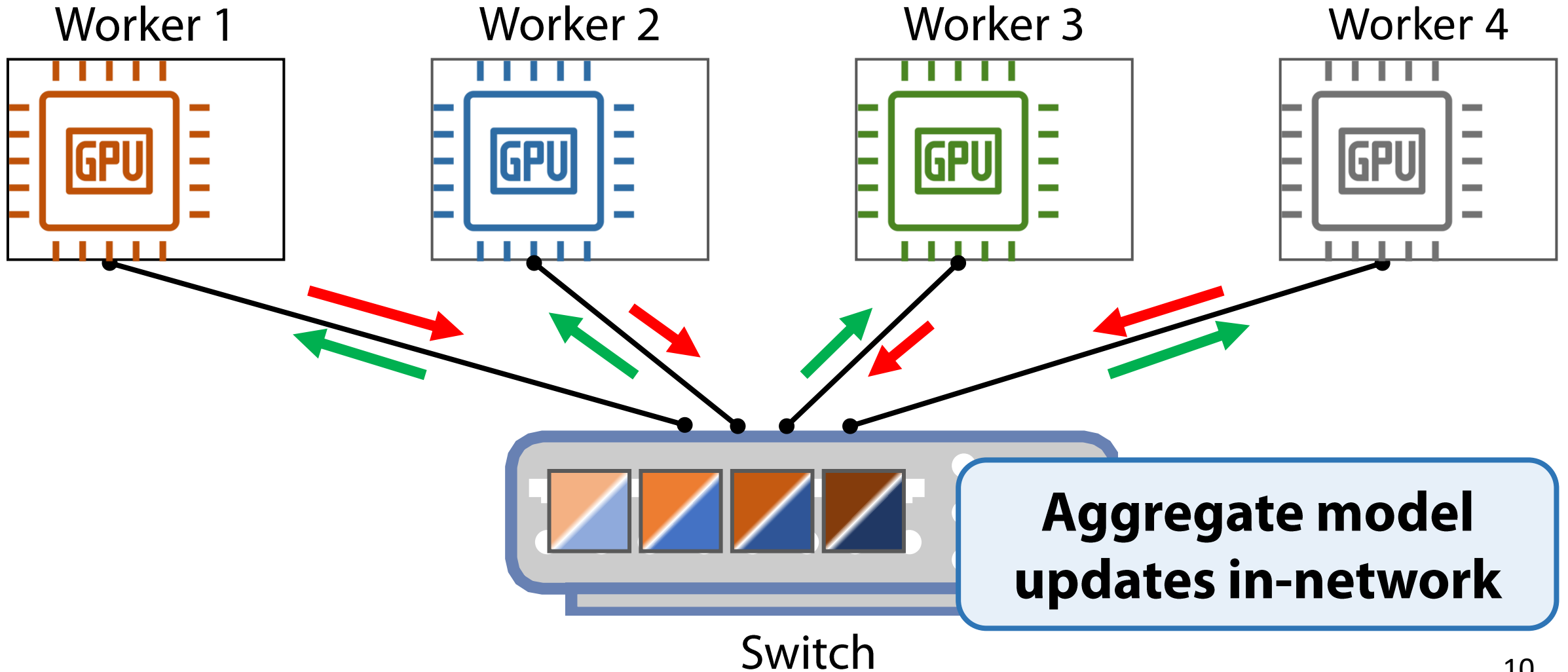
Programmable data plane



6.5 Tbps

100 Gbps  
line rate  
processing

# The network is the ML accelerator



# SwitchML: Co-design ML and networking

---

## Challenges

</> Limited computation

📦 Limited storage

🔌 **No floating points**

☁️ Packet loss



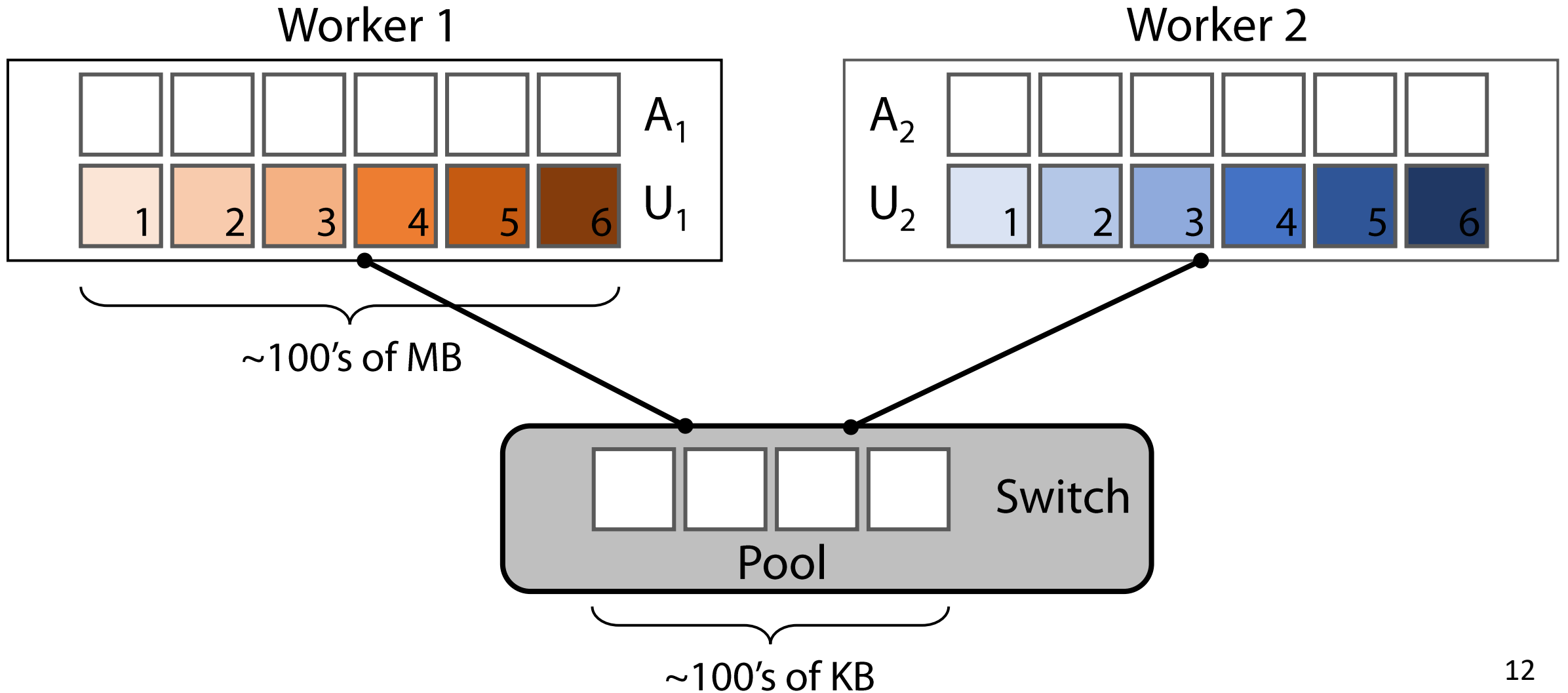
**6.5 Tbps**  
programmable  
data plane

## Design

- Combined switch-host architecture
- Pool-based streaming aggregation
- Quantized integer operations
- Failure-recovery protocol
- In-switch RDMA implementation

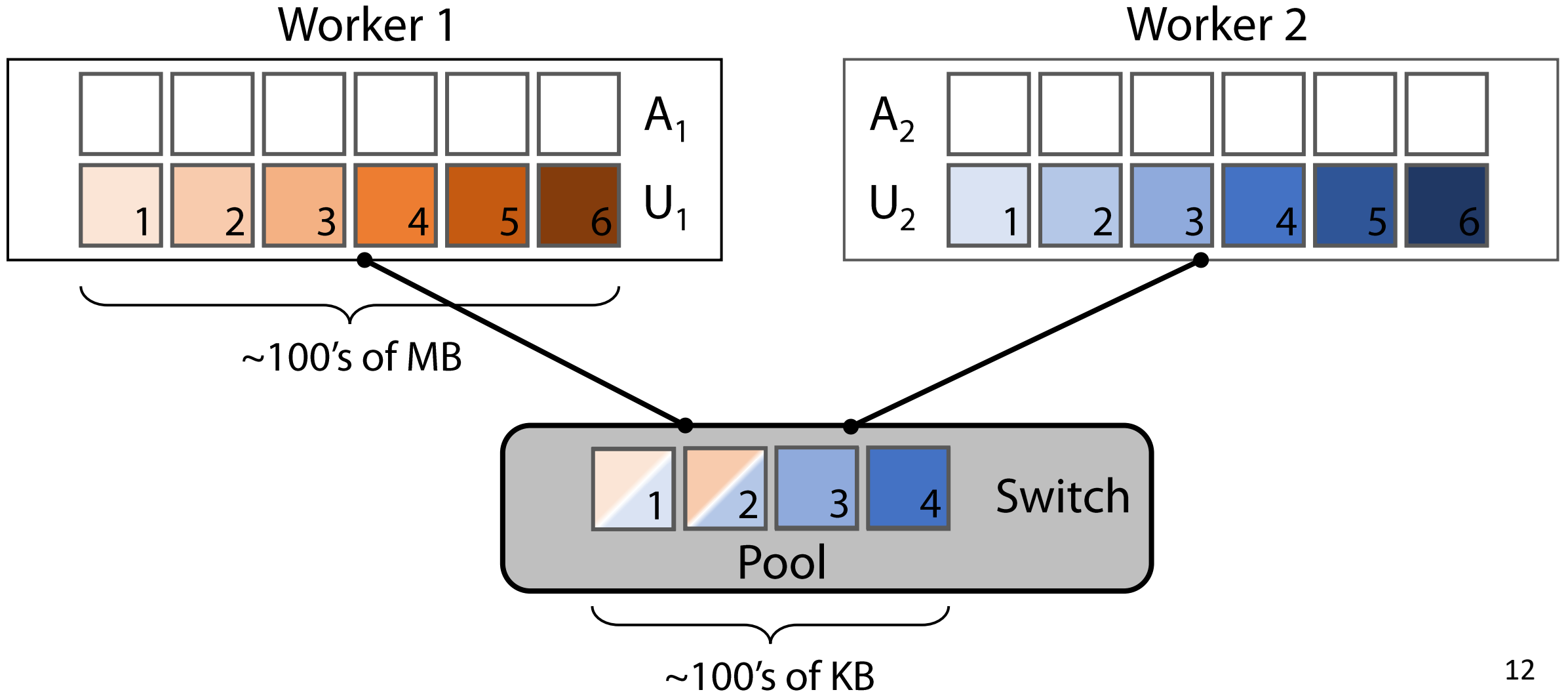
# Streaming aggregation

---



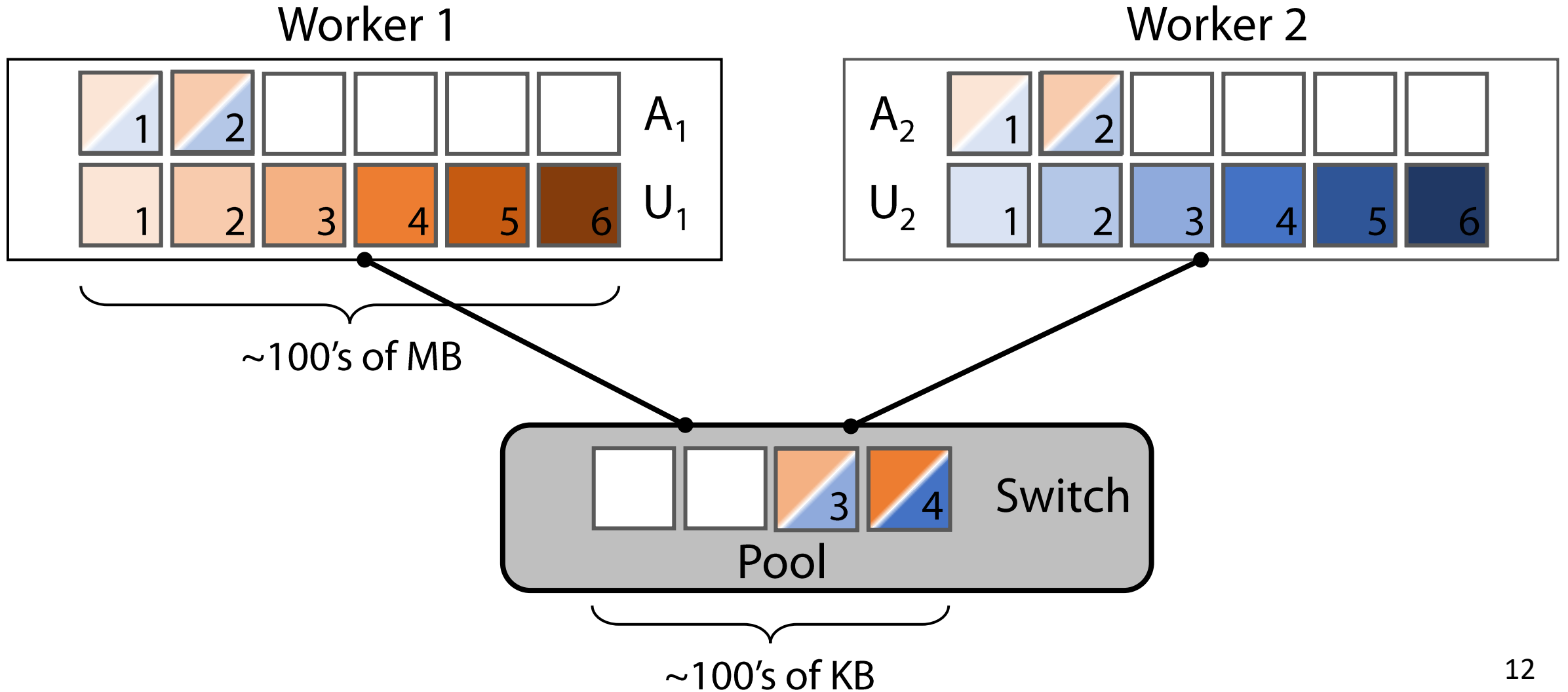
# Streaming aggregation

---



# Streaming aggregation

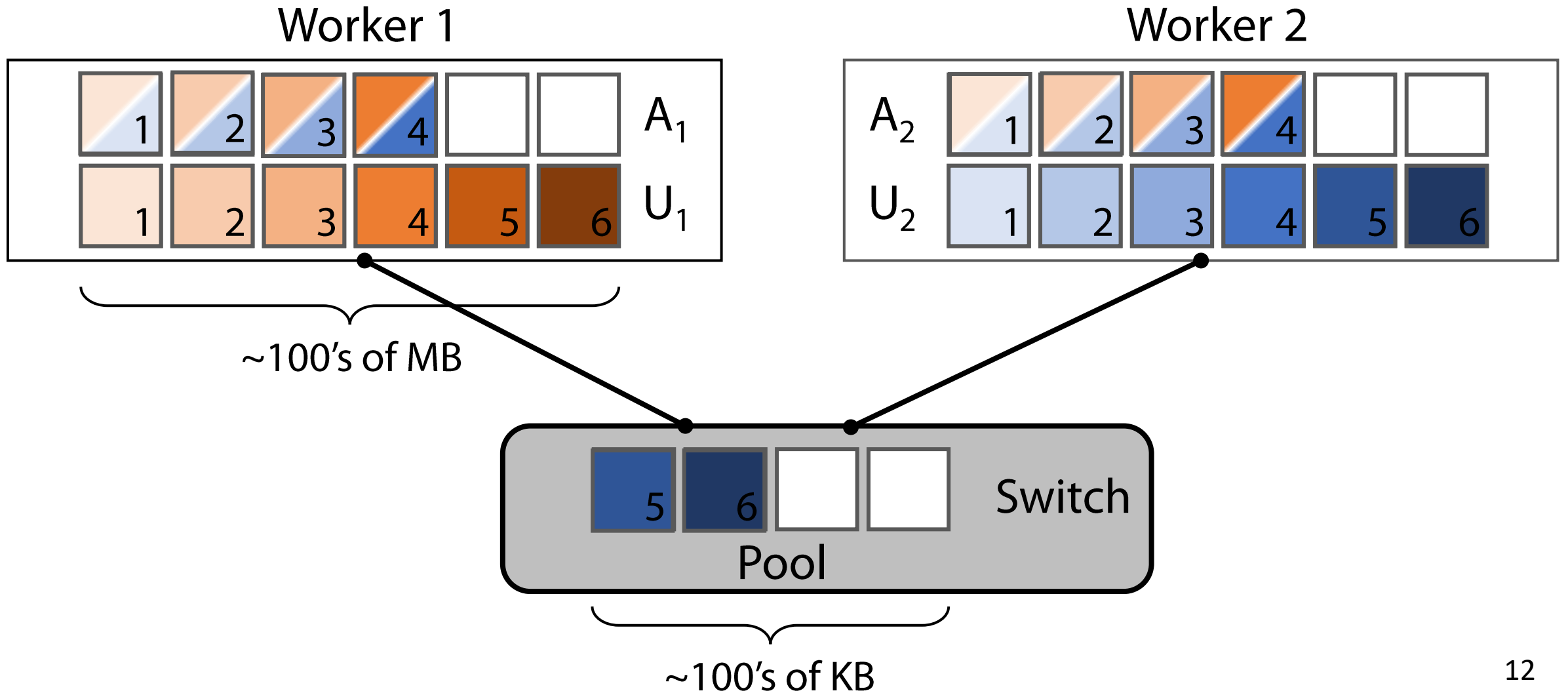
---



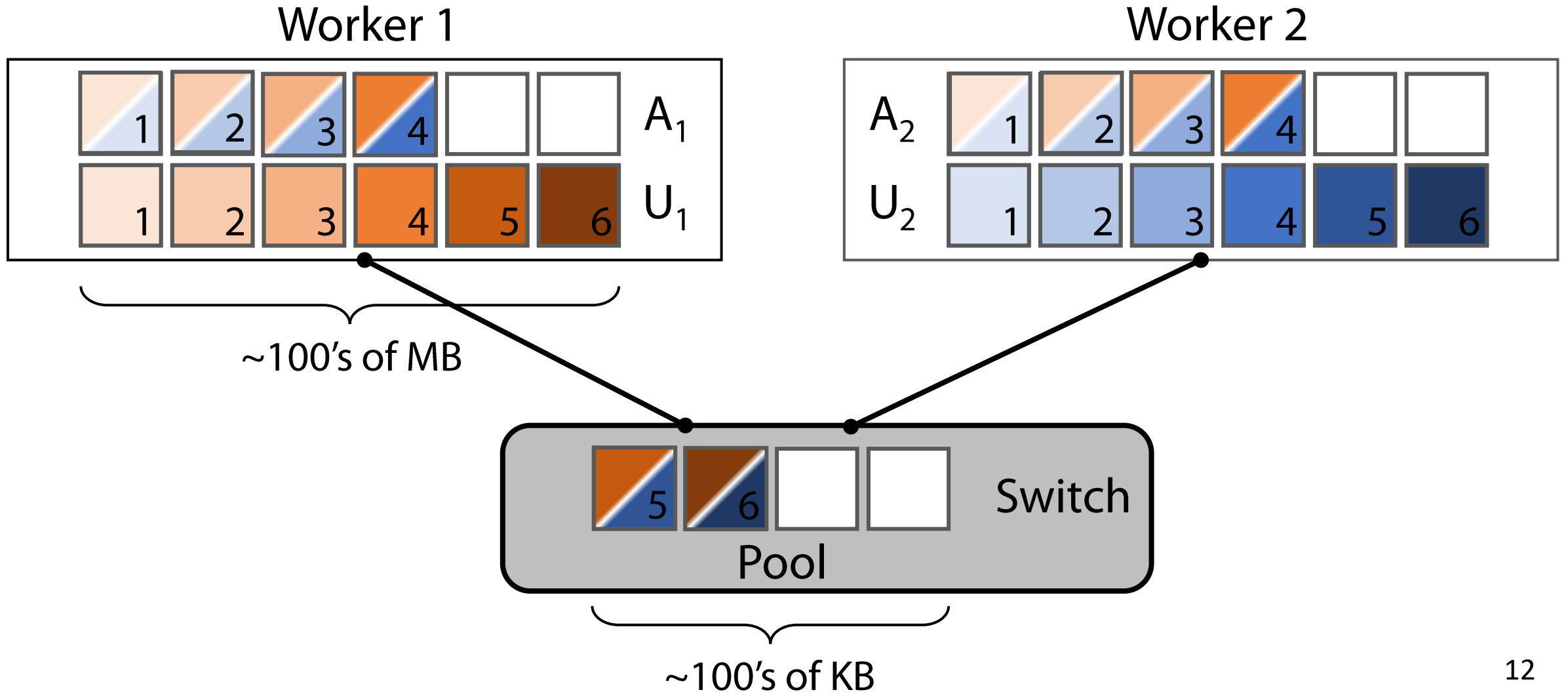


# Streaming aggregation

---

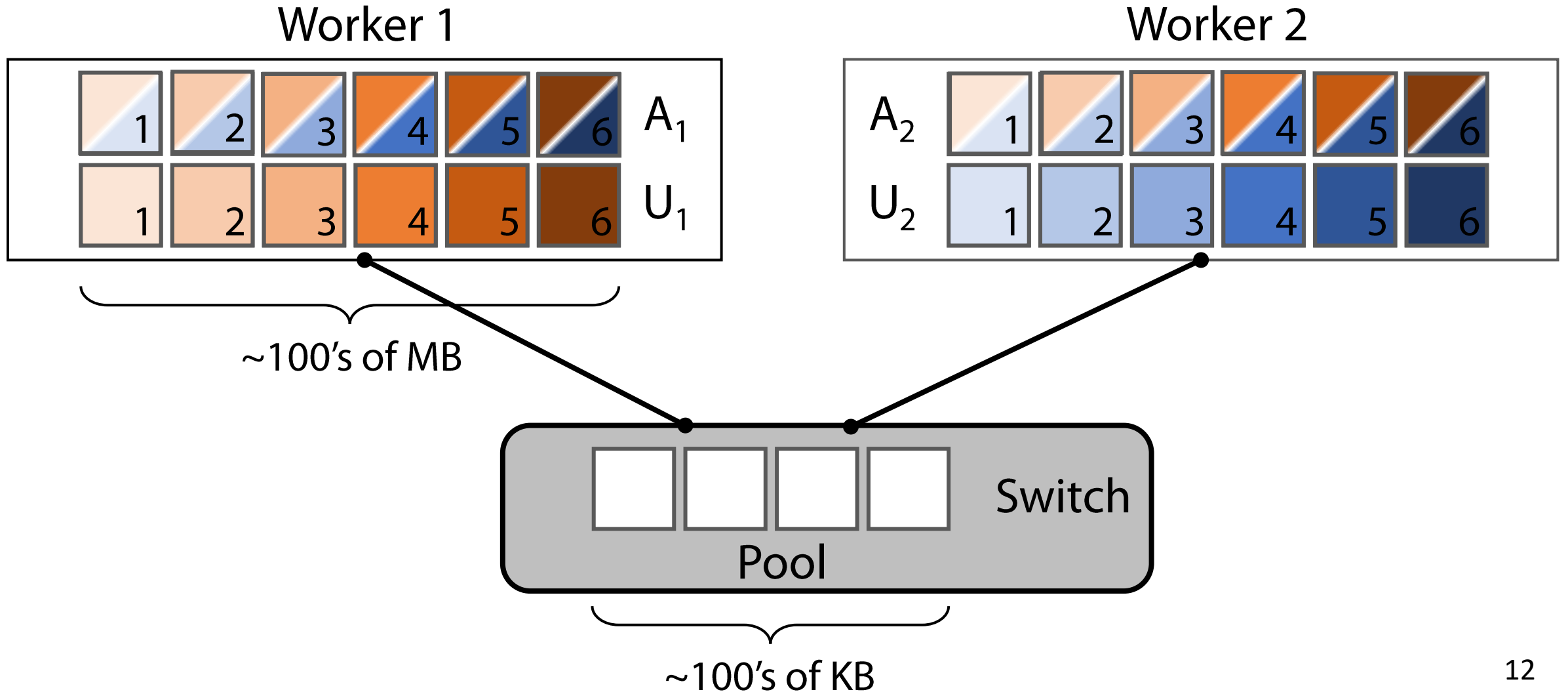


# Streaming aggregation

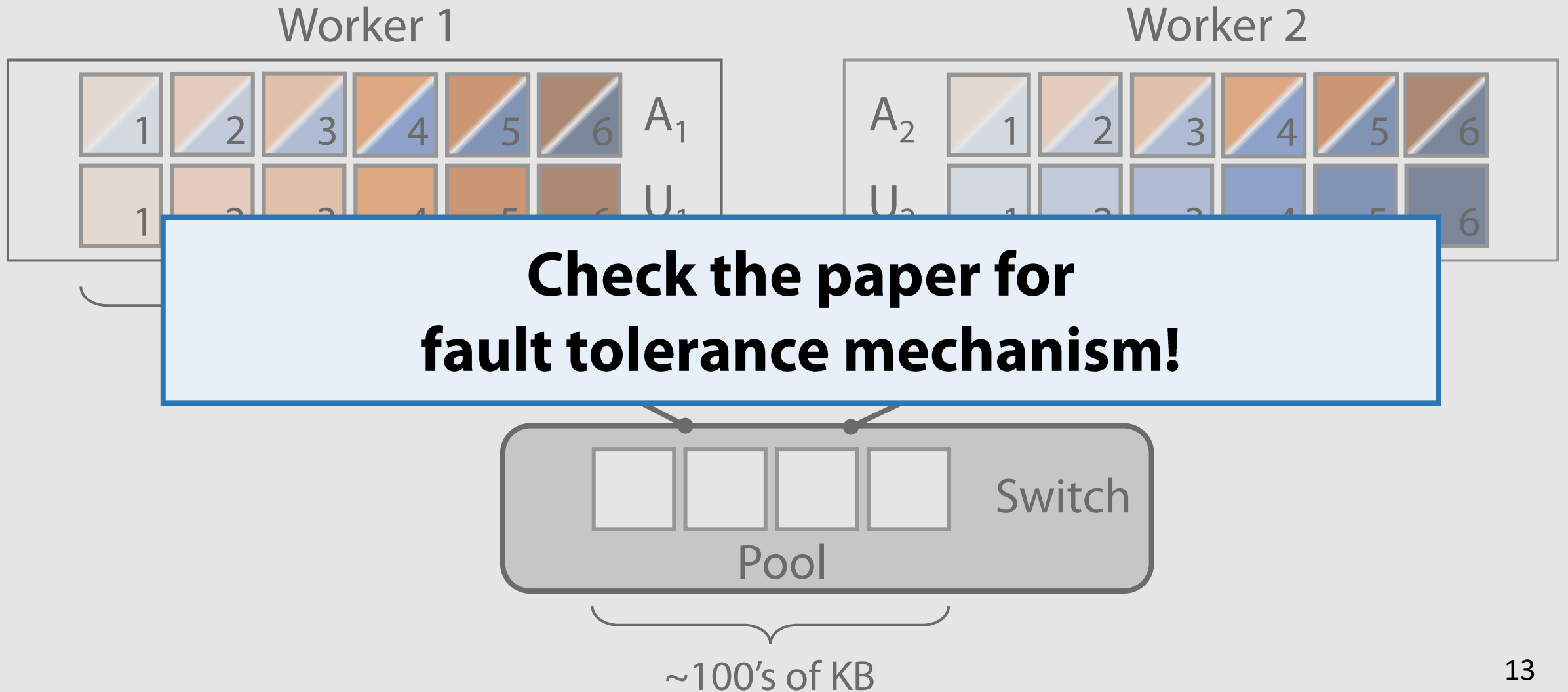


# Streaming aggregation

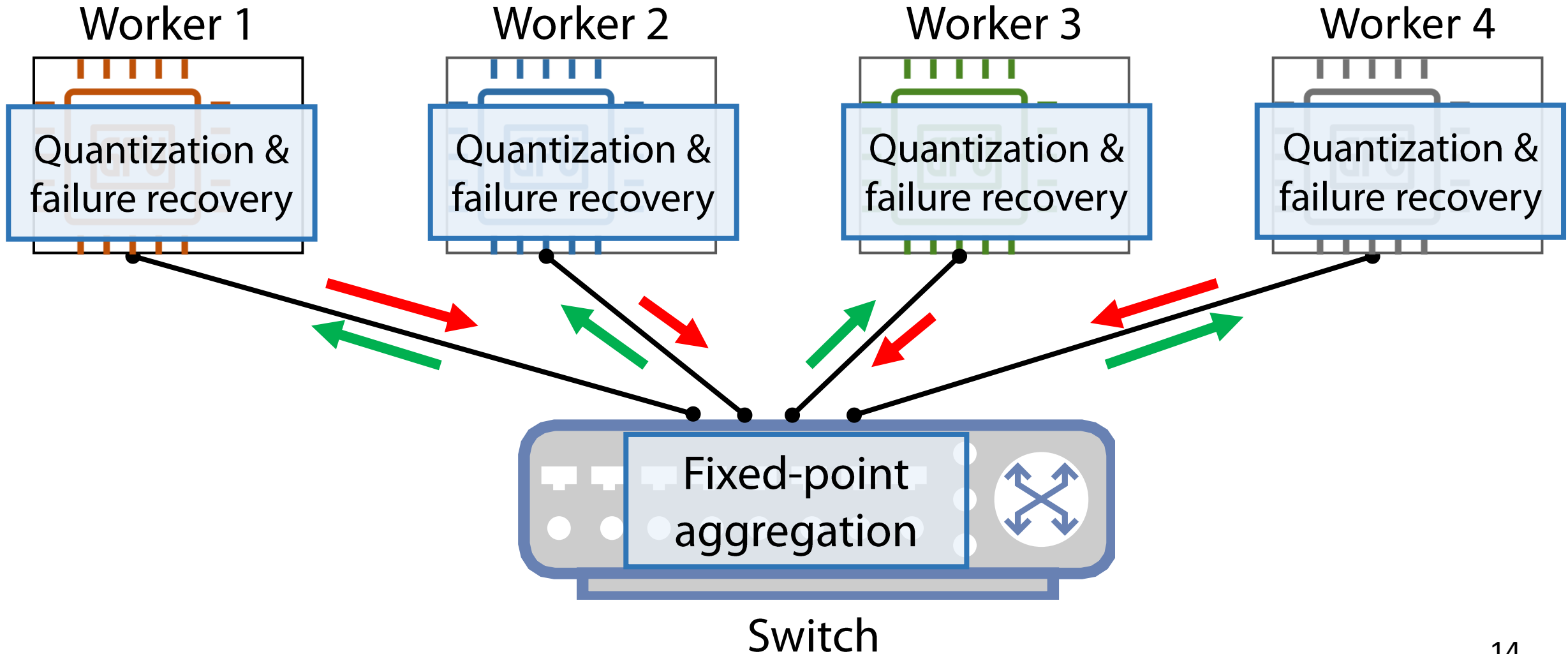
---



# Streaming aggregation



# Combined switch-host architecture



# Combined switch-host architecture

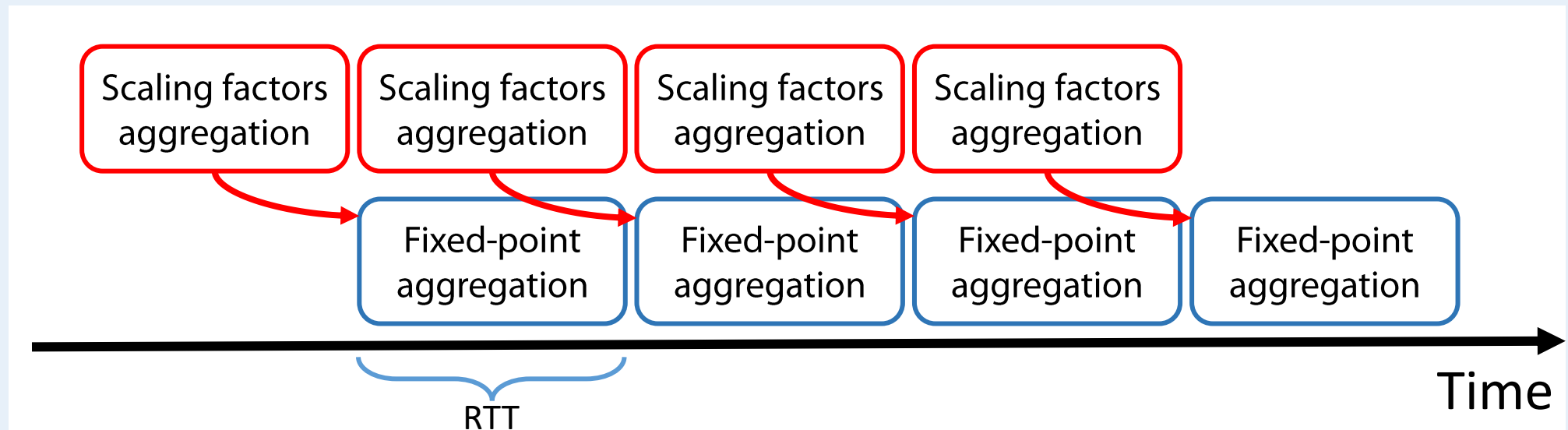
Worker 1

Worker 2

Worker 3

Worker 4

## Block quantization



# Combined switch-host architecture

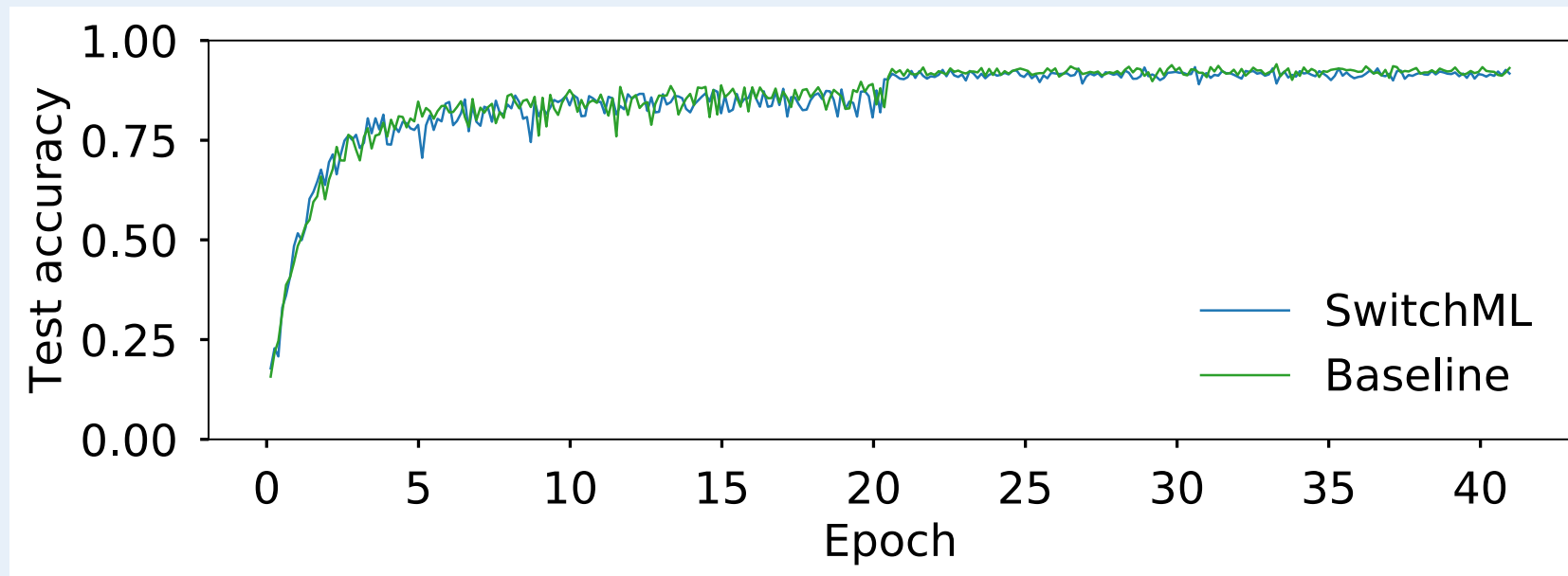
Worker 1

Worker 2

Worker 3

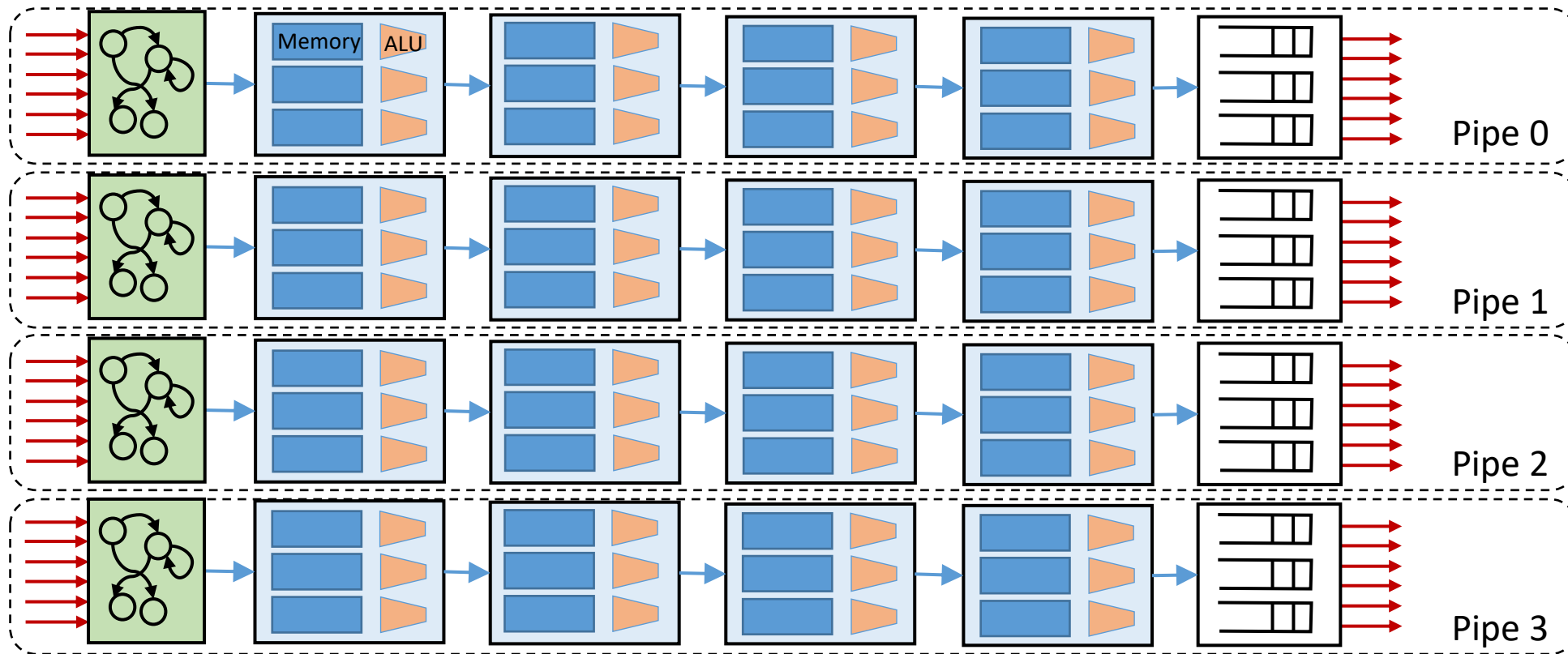
Worker 4

**Quantization allows training to similar accuracy in a similar number of iterations as an unquantized network**



# How large a packet can a switch process?

**256B** of payload per packet → **75.7%** network efficiency

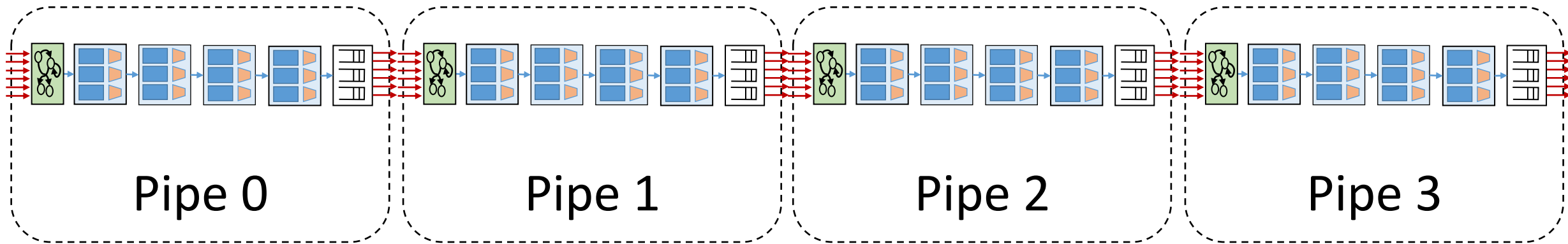




# How large a packet can a switch process?

---

**1024B** of payload per packet → **92.6%** network efficiency




# A glimpse of the evaluation

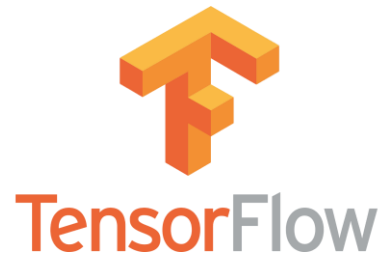
---

Check the paper for an extensive evaluation!

# Implementation and evaluation

---

- Switch program written for Intel Tofino 
- End-host C++ library providing a familiar all-reduce API
- Integrated with ML frameworks

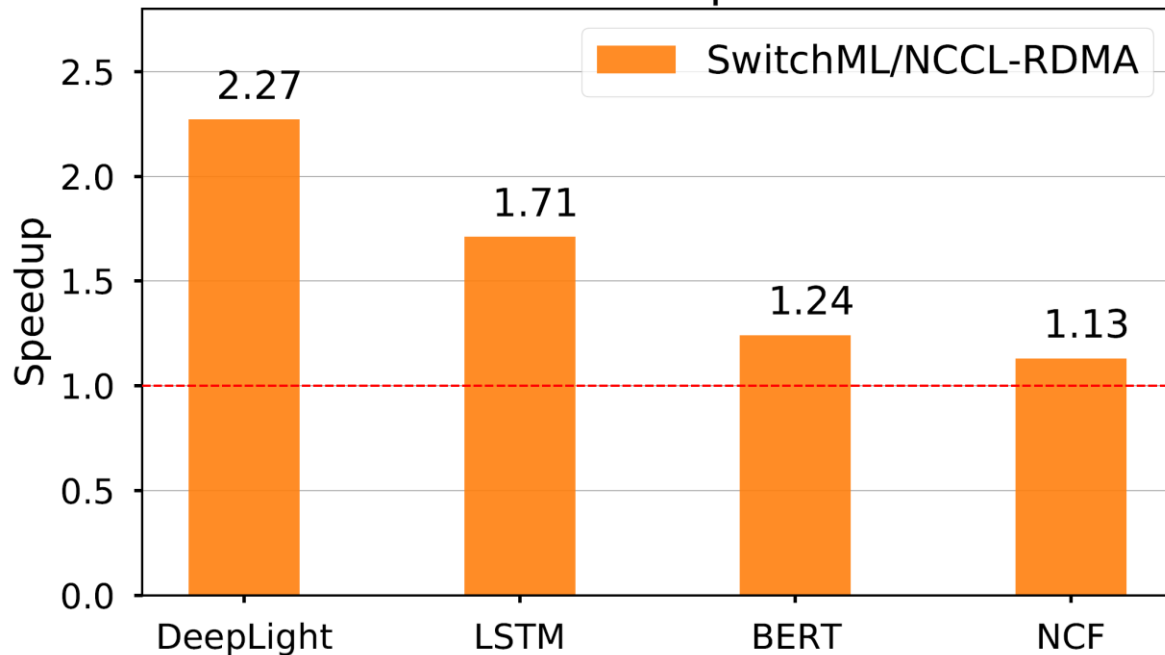


- Standard ML benchmarks
- Microbenchmarks for aggregation performance

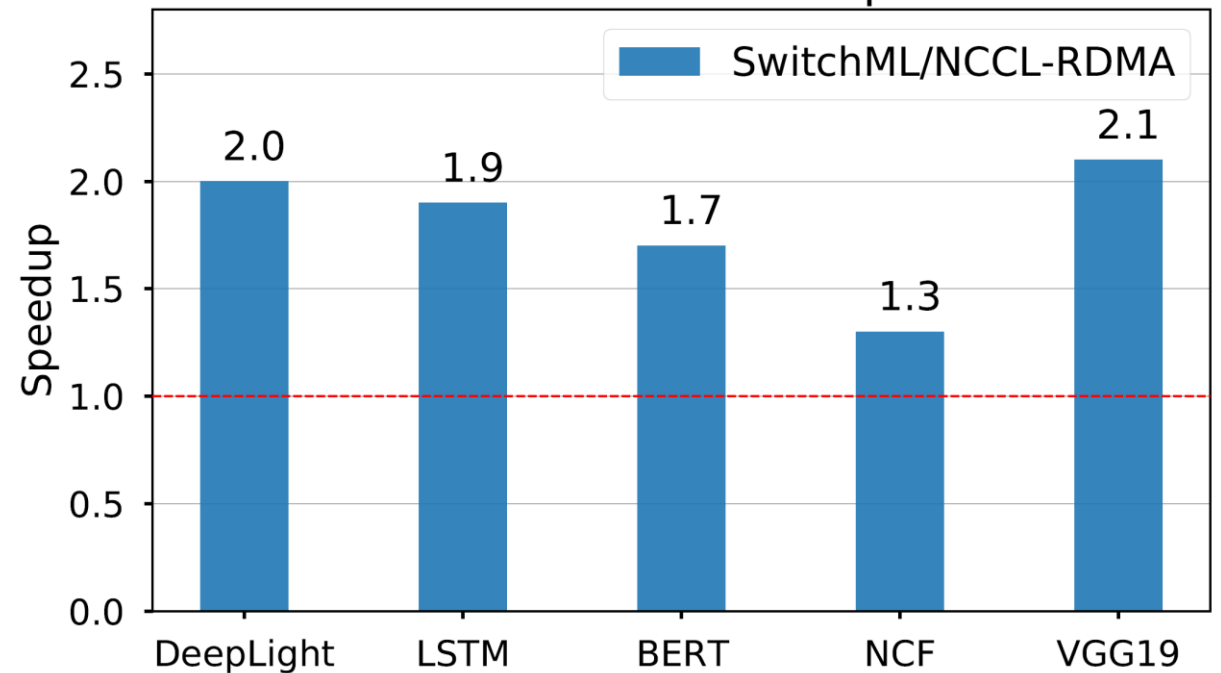
# How much faster is SwitchML?

SwitchML provides a speedup in training throughput up to 2.27x on 100Gbps networks. Speedup is higher with faster GPUs that reduce the computation/communication ratio.

100Gbps

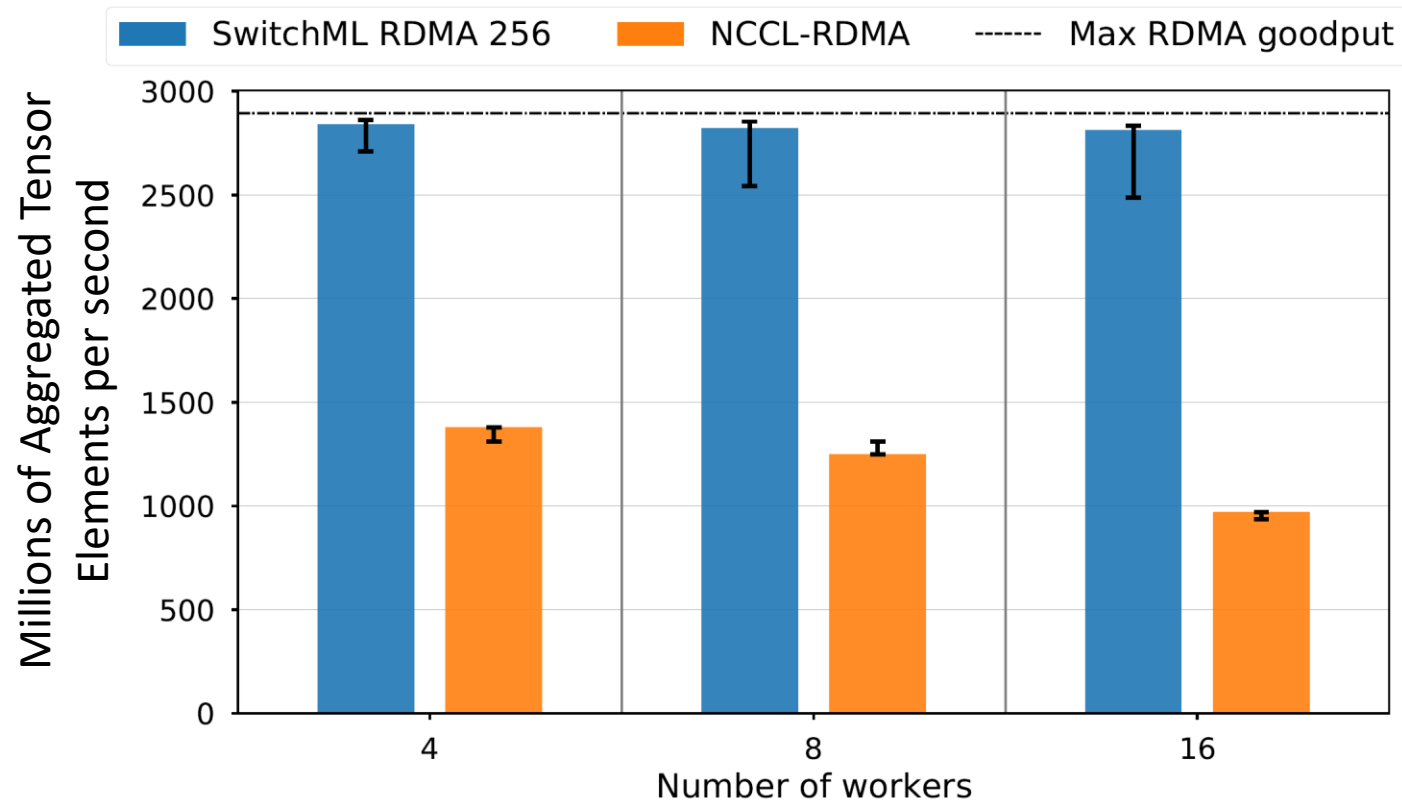


GPU 10x - 100Gbps



# How does SwitchML scale with the number of workers?

SwitchML performance does not depend on the number of workers.



# Summary

- Use **in-network aggregation** to synchronize model parameters updates
  - Reduce network traffic volume and latency
- SwitchML speeds up training up to 2.27x with real-world DNN benchmarks
- Aggregation time does not depend on the number of workers

[github.com/p4lang/p4app-switchML](https://github.com/p4lang/p4app-switchML)

