Towards Accelerating Data Intensive Applications
Shuffle Process Using SmartNICs

Jiaxin Lin, Tao Ji, Xiangpeng Hao, Hokeun Cha, Yanfang Le, Xiangyao Yu, Aditya Akella
Introduction
What’s a SmartNIC?

In simple terms:

- **A network interface card** is a piece of hardware that allows computers to communicate with other devices on a network.
- Adding to this, a SmartNIC also has dedicated CPU and Memory that could be utilized to off-load certain tasks.
Spark Recap
Understanding Spark Phases

- **Map Phase:**
  - Produce intermediate key-value pairs.

- **Shuffle Phase:**
  - **Pre-processing:**
    - **Partitioning:**
      - Intermediate results are partitioned per reducer task.
    - **Processing:**
      - Sort or Aggregate intermediate results before network I/O
  - Involves all-to-all data exchange over the network.

- **Reducer Phase:**
  - Aggregate and process the shuffled data, producing the final output.
Spark: GroupBy/Aggregates flow

Wide Transformation

Narrow Transformation

Machine 1

Machine 2
Issues with Wide Transformations

- Wide transformations involve **Shuffle phase** which is a **performance bottleneck**.

**Why?**

**Setup : 1000 nodes, 50 Map and 50 Reduce tasks per node**

1) The shuffle re-partitioning and pre-processing cost to sort or aggregate spike up the CPU usage.
2) Each Map tasks produces 50K partitions for all the reducer tasks and a total of 2.5M partitions for the overall Spark Job. - **Challenge 1**
   a) ~2.5M network I/O calls from reducer to Map
   b) ~2.5M random disk I/O reads on the Mapper.
Goals

- Develop **SmartShuffle** to accelerate data-intensive applications shuffle process.

Objectives

- Leverage SmartNICs to offload computation tasks related to shuffle processes.
- Address challenges posed by the limited computational resources of SmartNICs.
- Introduce a coordinated offload architecture involving both sender-side and receiver-side SmartNICs.
- Present a liquid offloading approach for dynamically migrating computations between host CPU and SmartNIC at runtime.
Motivation: CPU utilization

The CPU usage during the shuffle phase for TeraSort data is close 100% making it a perfect candidate to off-load application-level computation into the networking layer.
What can be offloaded to SmartNIC

- Data Partitioning
- Stateful operators such as aggregation and sorting

Challenges of SmartNIC offloading

**Limited Memory**: Typical RAM size of on-NIC DRAM is 4GB-16GB
- If intermediate output size from Map task is greater than on-NIC RAM size then it would be difficult to fully offload them to NIC. - **Challenge 2**

**Limited CPU**:  
- SmartNICs today have fewer and slower cores than the host server.  
- SmartNIC can cause a performance bottleneck, and the subsequent reduce task will be hindered by the slow SmartNIC cores. - **Challenge 3**
SmartShuffle Architecture
Shuffle Manager:
- Monitors the execution process of map and reduce tasks and controls the shuffle operation across the cluster.

Shuffle Agent:
- Manages host-NIC communication and enforces the rate-based dynamic migration policy

Shuffle Workers:
- An individual on-NIC thread that runs one or more offloaded stateful/stateless operators.

Traffic Scheduler:
- on-NIC orchestration thread that offloads shuffle’s all-to-all network communication process
SmartShuffle uses the two-level partition, which turns the shuffle process from per-task granularity to per-node granularity.

- The map-side SmartNIC workers merge the output from multiple map tasks and partition data.
- The reduce-side SmartNIC gathers and repartitions the data based on the local reduce task number at the node.
- Effectively reducing the number of I/O call to 1k from previous example

Challenge 1- Network/Disk I/O
Coordinated Offloading

- Both the **map-side** and the **reduce-side** SmartNICs of a shuffle jointly contribute to the shuffle offload and relevant computation.
- Map-Side SmartNIC does partitioning and partial aggregation/sorting on the intermediate output from the that node specific map tasks.
- Reducer side does arrangements on the overall partitions which it receives from multiple Maps tasks.

Spilling

Resolves **Challenge 2**

2 scenarios of spilling

- When the Shuffle worker completes its job and the data is ready.
- When data for stateful operators such as sorting don’t fit into the **limited memory** of NIC: In such a case partial results/Input is spilled to next hop.
Workload Migration: Challenge 3

- To maximize the amount of work offloaded to the SmartNIC while avoiding the typically slow SmartNIC cores becoming a bottleneck:
  - Shuffle Agent monitors the growth of DMA registered buffer as signal to check if SmartNIC is overloaded.
  - Threshold for the buffer occupancy is defined as and until \( R=0 \), thread are launched on host machine.

\[
R = \frac{Mapper\_Produced\_Data\_Over\_TimeWindow}{NIC\_Consumed\_Data\_Over\_TimeWindow} - 1
\]
Evaluation
Spark’s total I/O request count grows quadratically with a job’s parallelism.

SmartShuffle, the I/O request count is not influenced by parallelism, as it does node-level I/O merging in the SmartNIC.
- Additional time taken for smart shuffle w/o migration accounts for less powerful core of SmartNIC.

- As the Spilling threshold increases, the amount of data aggregated on smartNIC increases leading to high aggregation rates.
Questions?
Thank You