Running Spark on a High-Performance Cluster using RDMA Networking and NVMe Flash

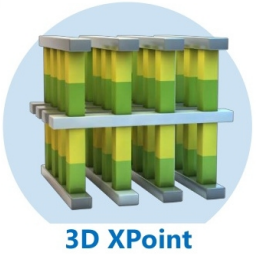
Patrick Stuedi, IBM Research

Hardware Trends

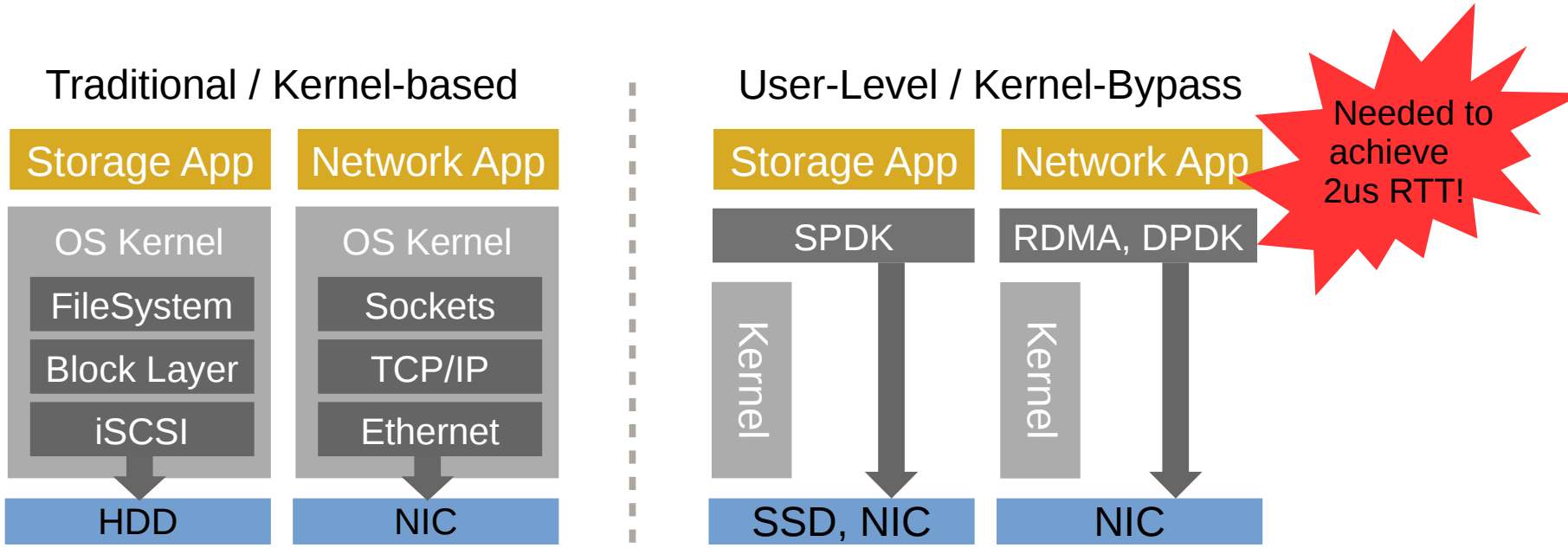
	2010	community target 2017
Storage	100 MB/s 100ms	1000 MB/s 200us
Network	1Gbps 50us	10Gbps 20us
CPU	~3GHz	~3GHz

Hardware Trends

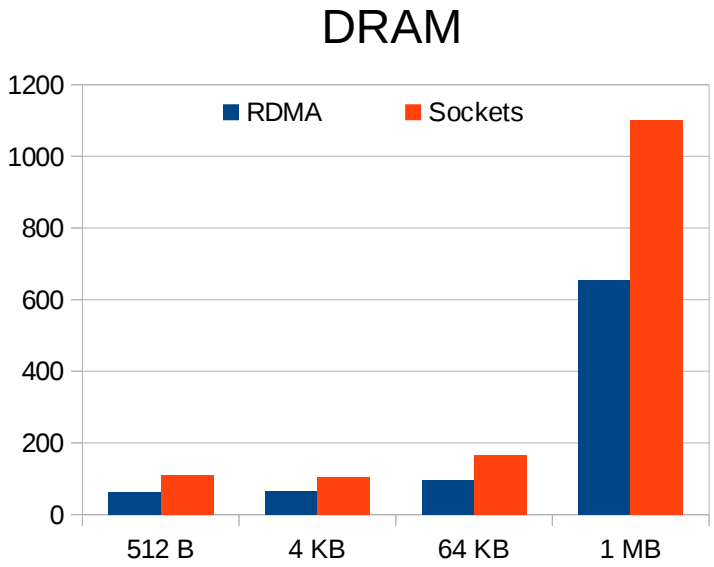
	2010	community target 2017	our target 2017
Storage	100 MB/s 100ms	1000 MB/s 200us	10 GB/s 50us
Network	1Gbps 50us	10Gbps 20us	100Gbps 2us
CPU	~3GHz	~3GHz	☹

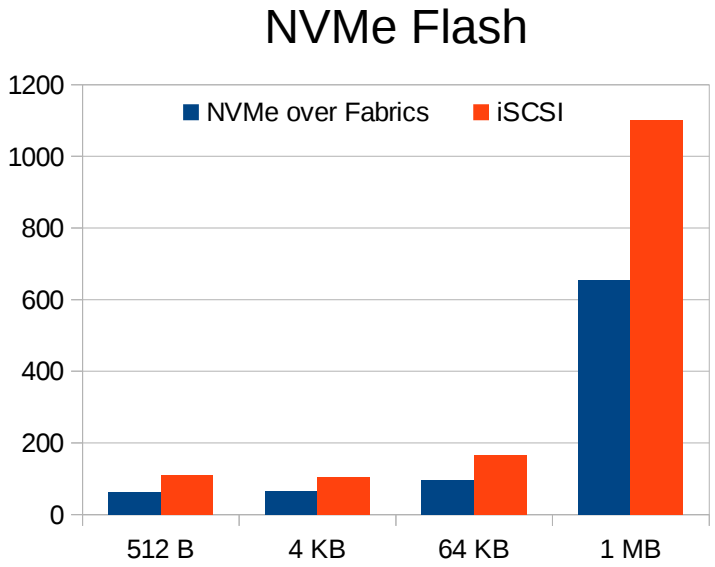


User-Level APIs

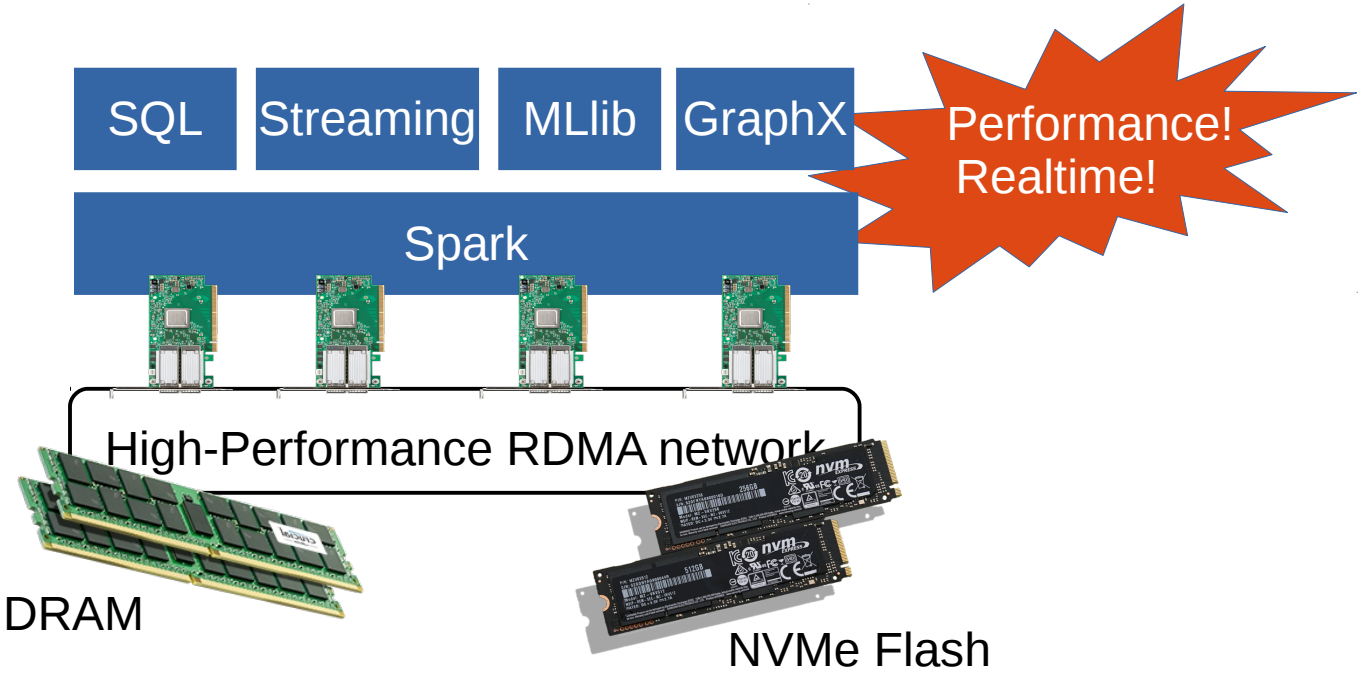


Remote Data Access

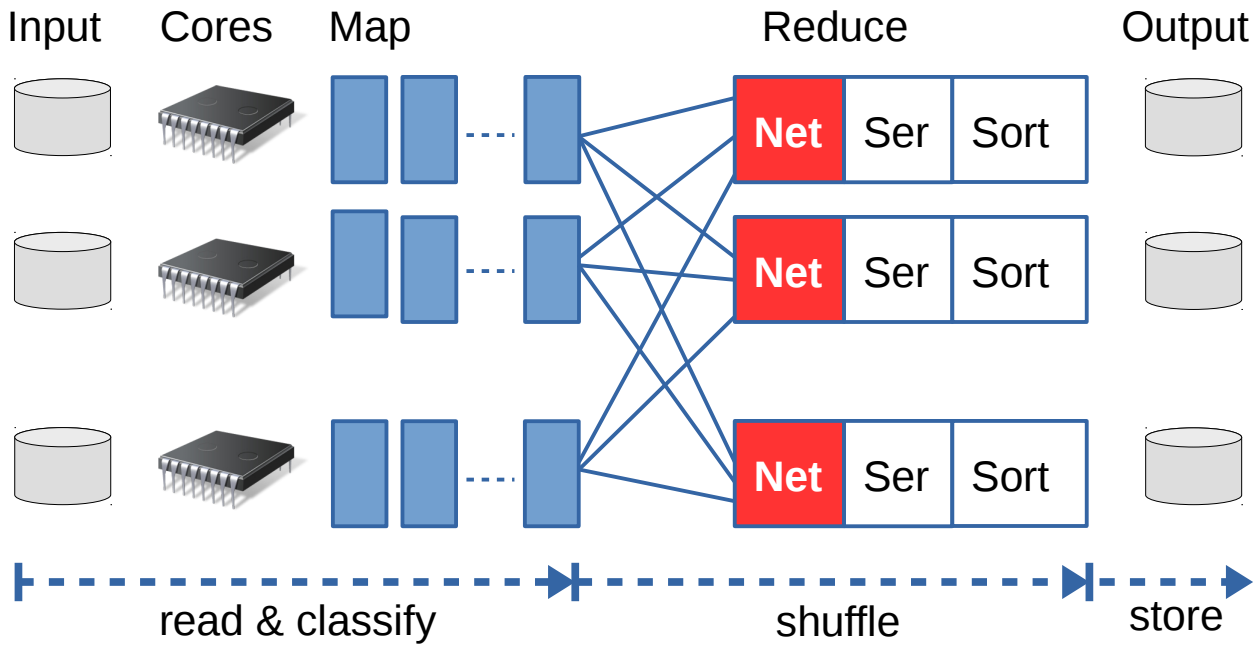




Let's Use it!

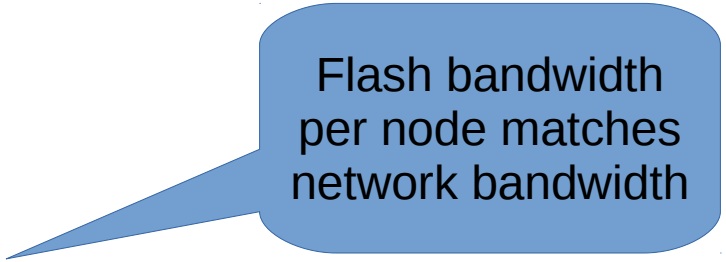

SQL
Streaming
MLlib
GraphX
Performance!
Realtime!
Spark
High-Performance RDMA network
DRAM
NVMe Flash

Case Study: Sorting in Spark

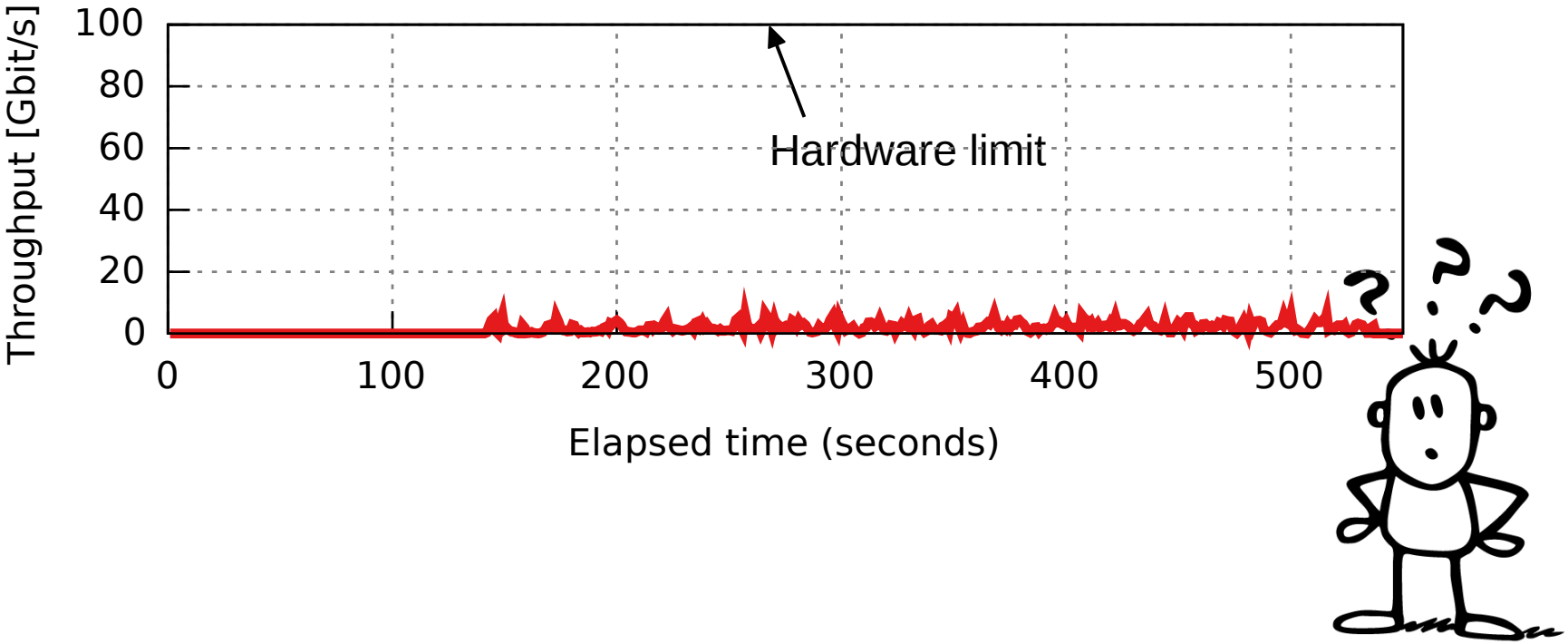


Experiment Setup

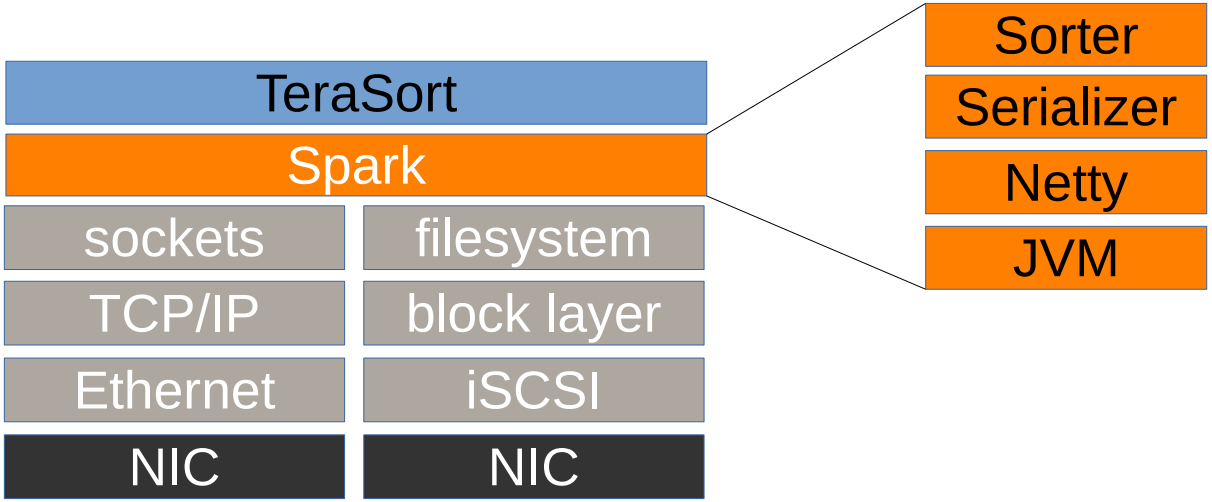
- Total data size: 12.8 TB
- Cluster size: 128 nodes
- Cluster hardware:
  - DRAM: 512 GB DDR 4
  - Storage: 4x 1.2 TB NVMe SSD
  - Network: 100GbE Mellanox RDMA



How is the Network Used?

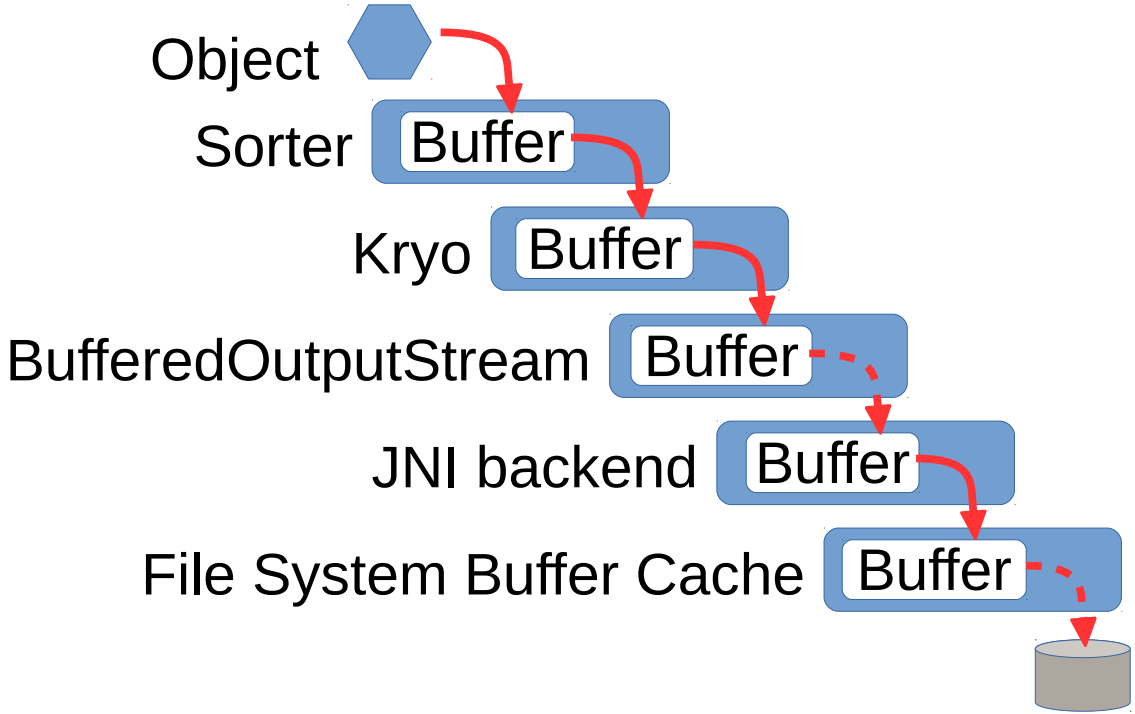


What is the Problem?

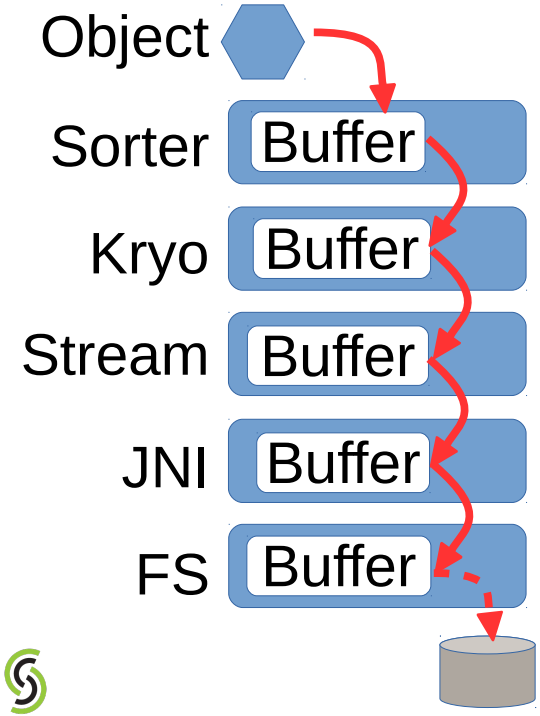


- Spark uses legacy networking and storage APIs: no kernel-bypass
- Spark itself introduces additional I/O layers: Netty, serializer, sorter, etc.

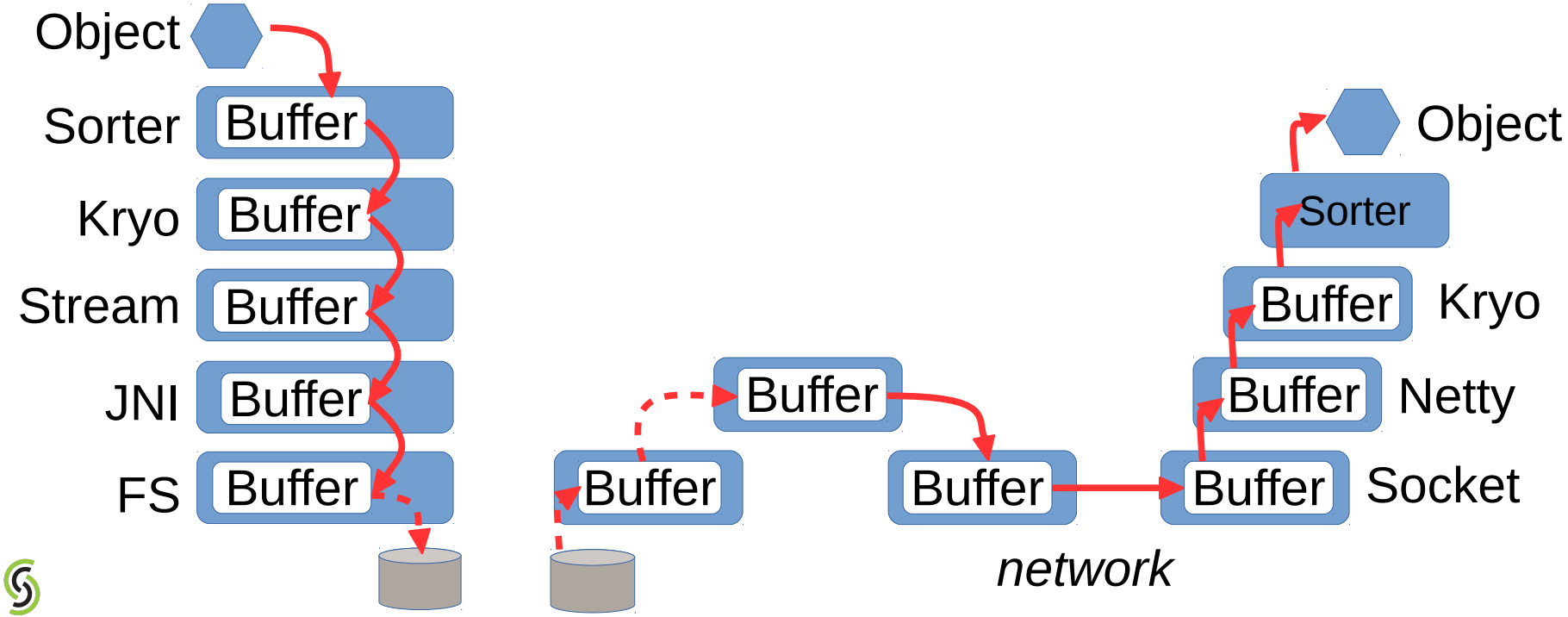
Example: Shuffle (Map)



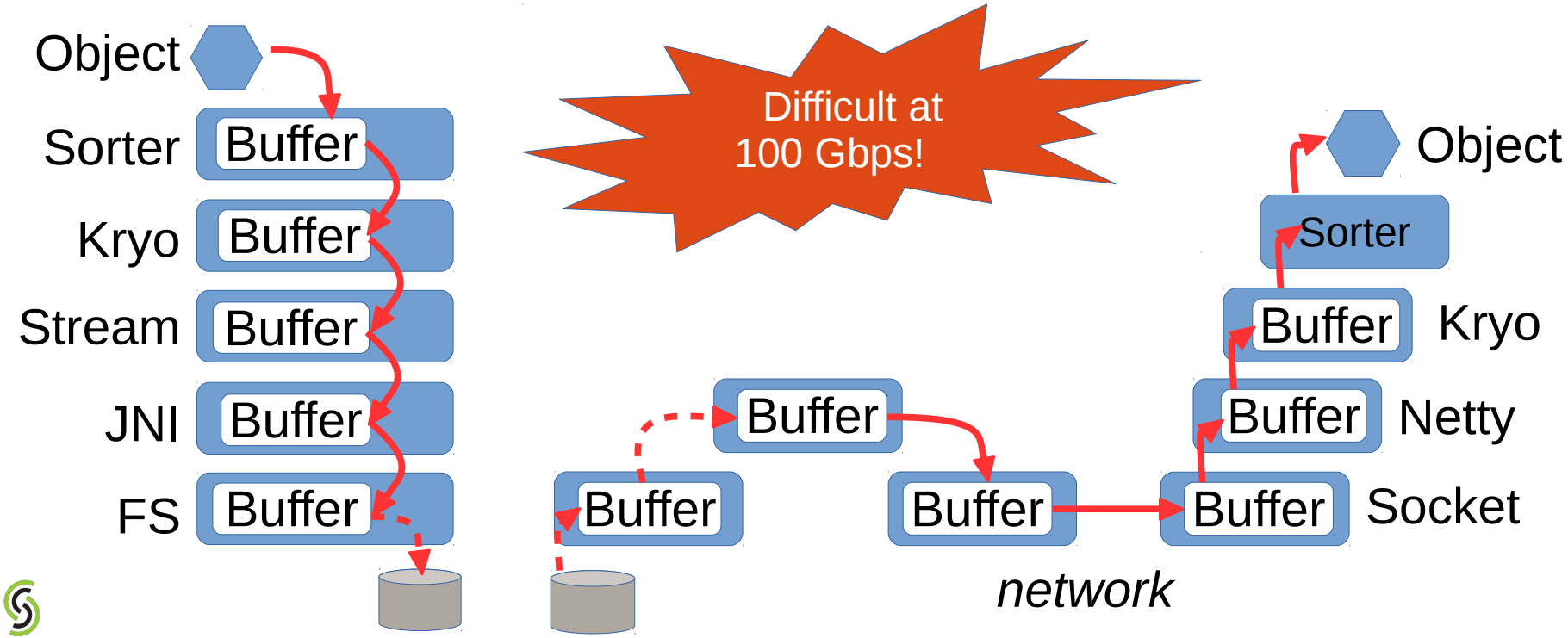
Example: Shuffle (Map)



Example: Shuffle (Map+Reduce)



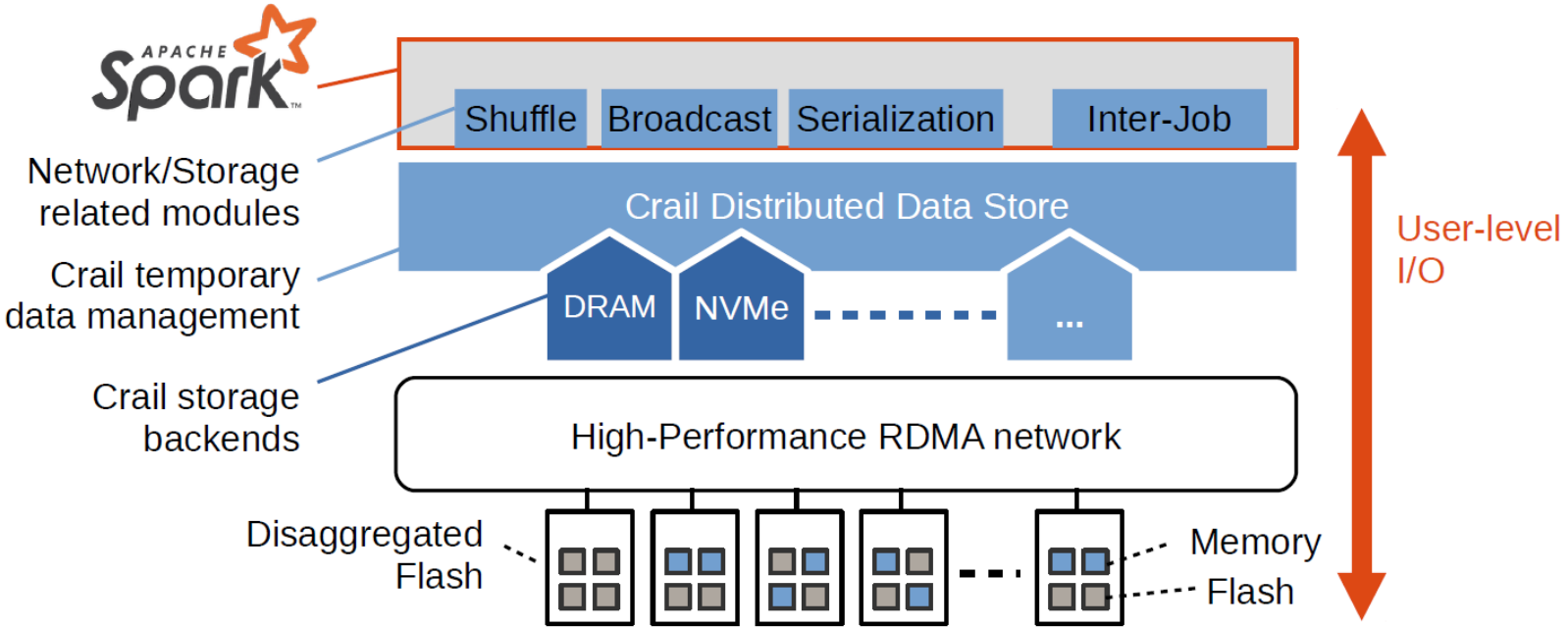
Example: Shuffle (Map+Reduce)



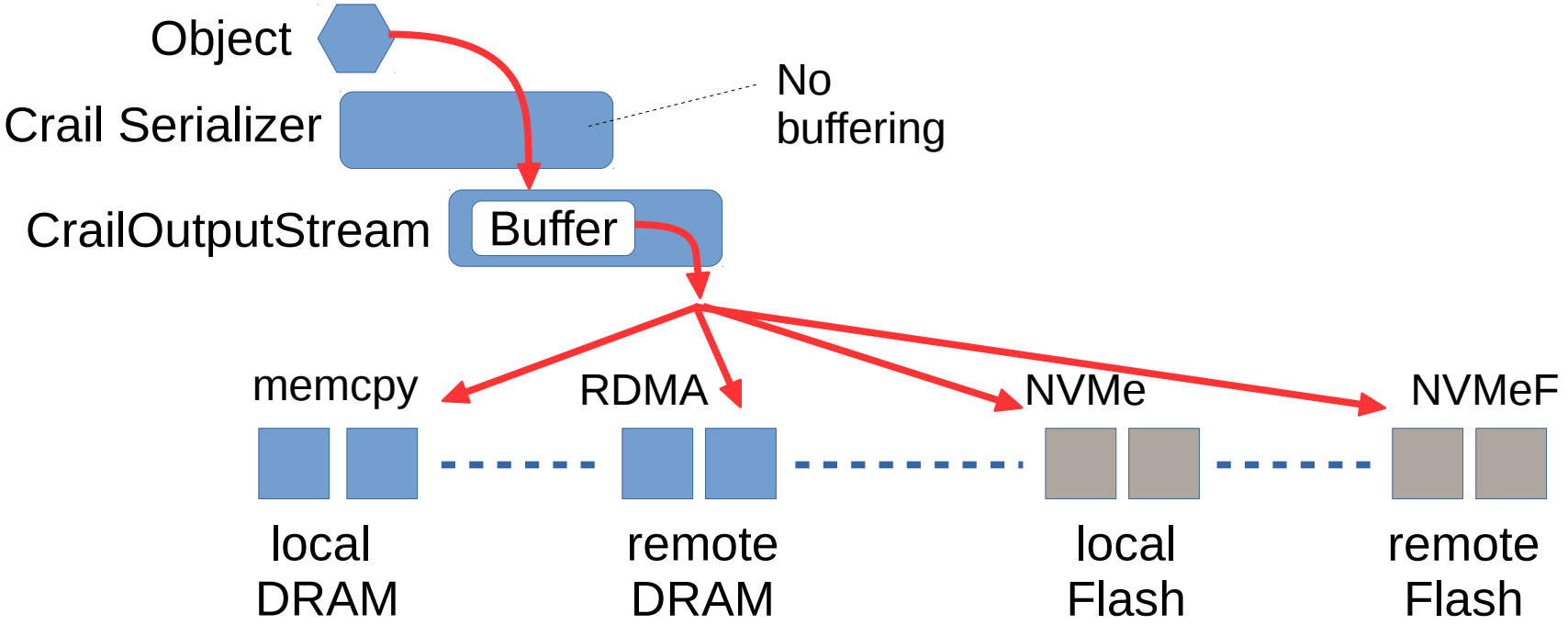
How can we fix this...

- Not just for shuffle
  - Also for broadcast, RDD transport, inter-job sharing, etc.
- Not just for RDMA and NVMe hardware
  - But for any possible future high-performance I/O hardware
- Not just for co-located compute/storage
  - Also for resource disaggregation, heterogeneous resource distribution, etc.
- Not just improve things
  - Make it perform at the hardware limit

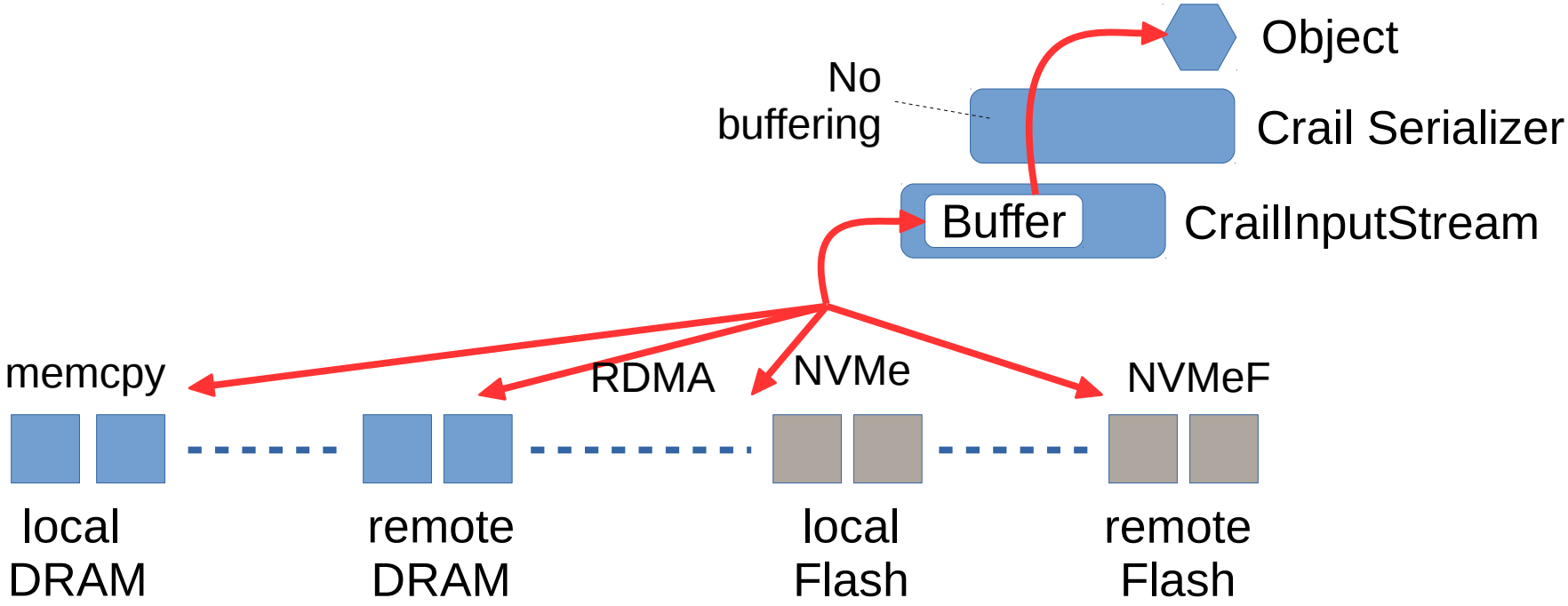
The CRAIL Approach



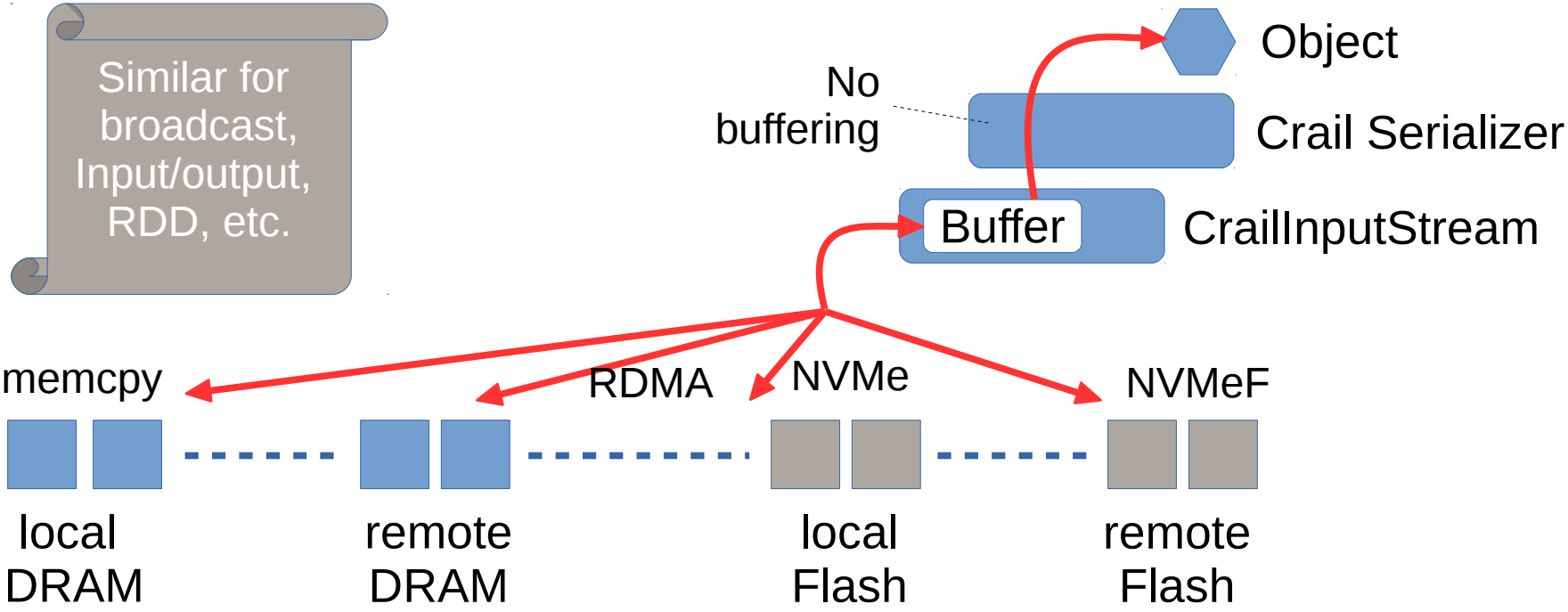
Example: Crail Shuffle (Map)



Example: Crail Shuffle (Reduce)



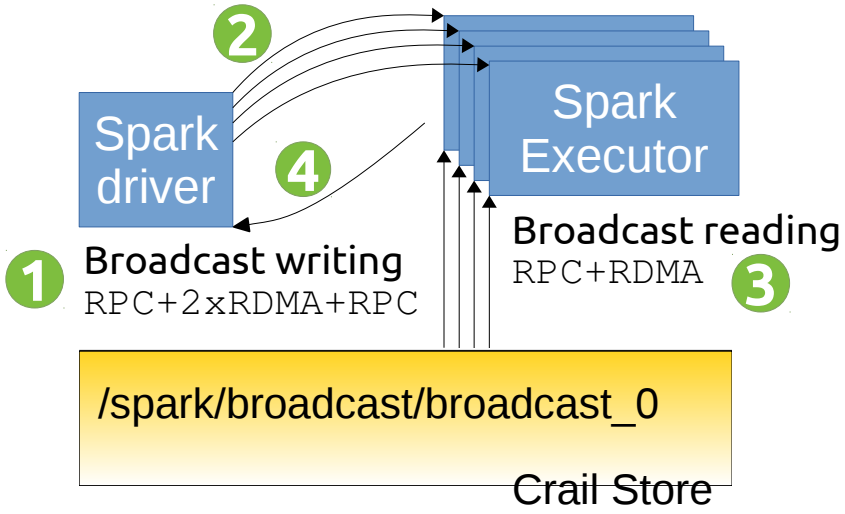
Example: Crail Shuffle (Reduce)

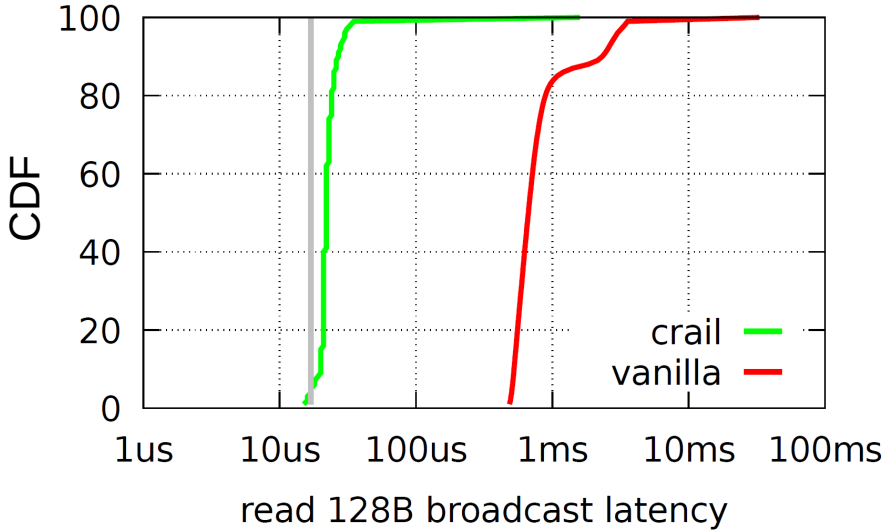


Performance: Configuration

- Experiments
  - Broadcast, GroupBy, TeraSort, SQL
- Cluster size: 8 nodes
- Cluster hardware:
  - DRAM: 512 GB DDR 4
  - Storage: 4x 1.2 TB NVMe SSD
  - Network: 100GbE Mellanox RDMA

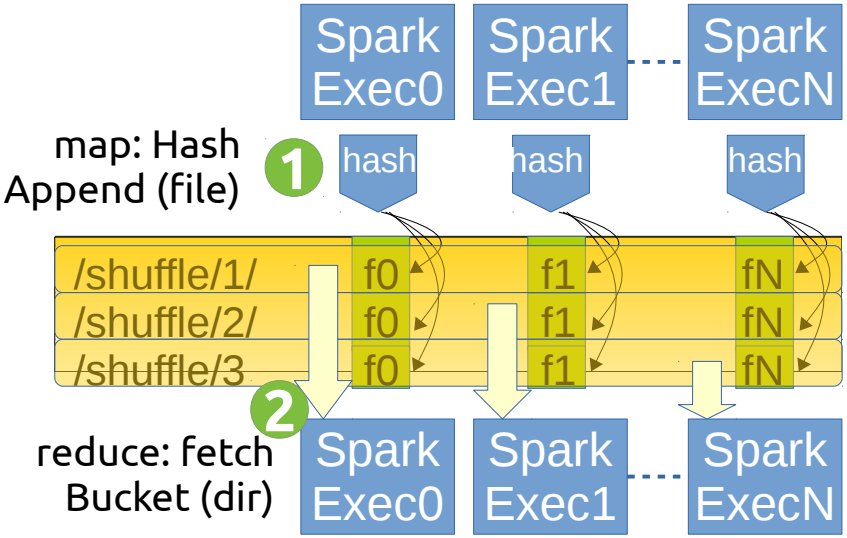
Spark Broadcast

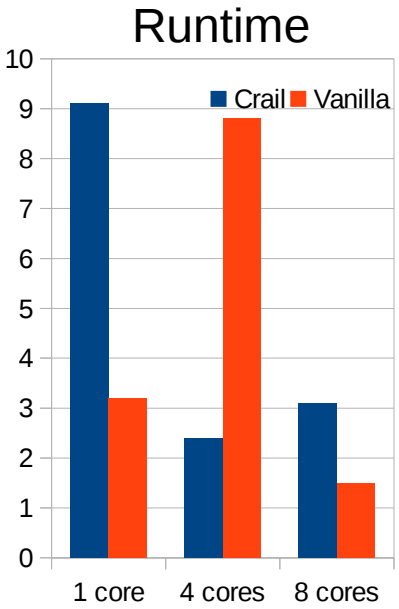


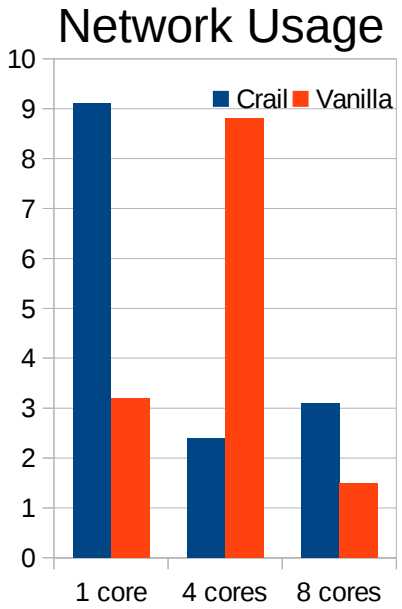



```
val bcVar = sc.Broadcast(new Array[Byte](128))
sc.parallelize(1 to tasks, tasks).map(_ => {
  bcVar.value.length
}).count
```

Spark GroupBy








```
val pairs = sc.parallelize(1 to tasks, tasks).flatmap(_ => {
  var values = new array[(Long,Array[Byte])](numKeys)
  values = initValues(values)
}).cache().groupByKey().map(v => v._1).count()
```

Thank You.

The CRAIL Approach