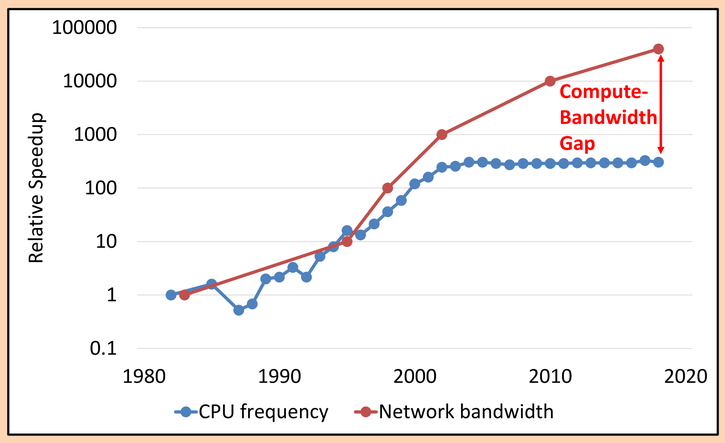
StRoM – Smart Remote Memory

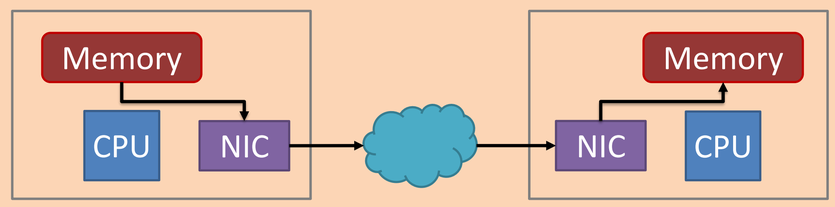
David Sidler*‡, Zeke Wang†‡, Monica Chiosa‡, Amit Kulkarni‡, Gustavo Alonso‡
* Microsoft Corporation
† Collaborative Innovation Center of Artificial Intelligence, Zhejiang University
‡ Systems Group, Department of Computer Science, ETH Zürich

Compute-Bandwidth Gap



- Increase in CPU cycles spent on network processing
- Context switches between OS network stack and application amplify the issue

RDMA (Remote Direct Memory Access)



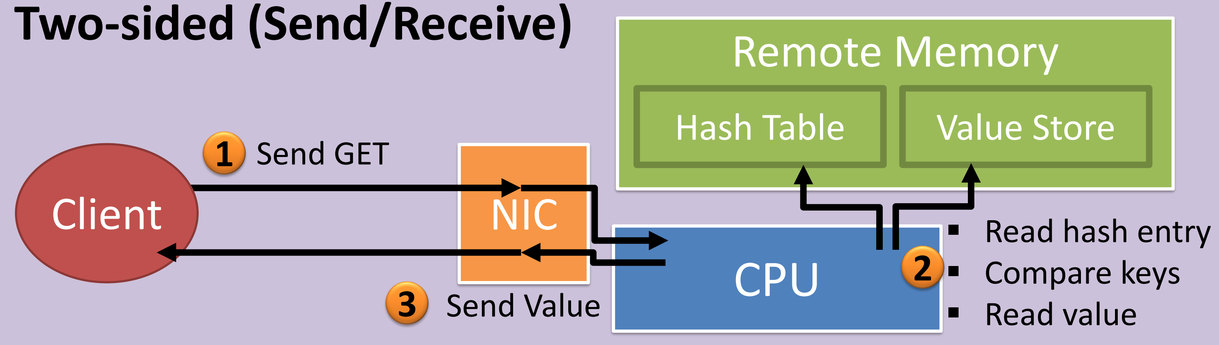
Complete hardware offload
=> Bypasses OS and CPU

Two-sided vs One-sided

Two types of RDMA operations:

- **Two-sided** involves the remote CPU, as in socket-based communication
- **One-sided** directly accesses remote memory, bypassing the remote CPU

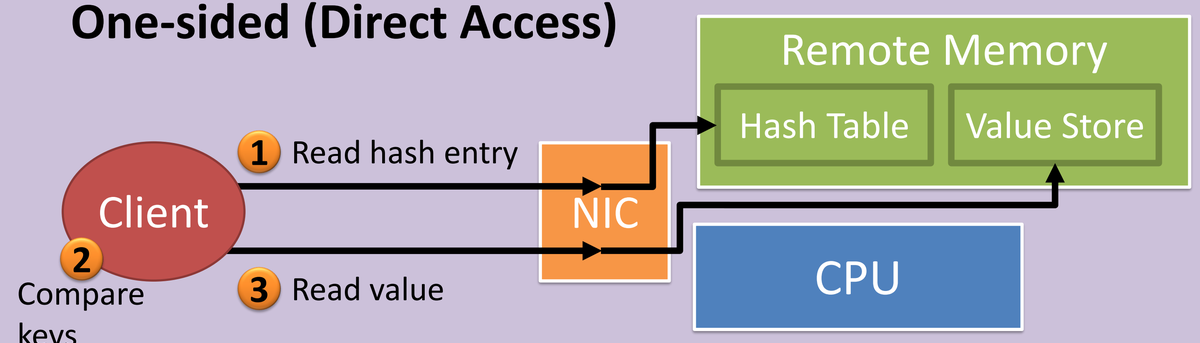
Two-sided (Send/Receive)



- **Single round trip**
- **Simple client-server model**
- **Remote CPU involved**

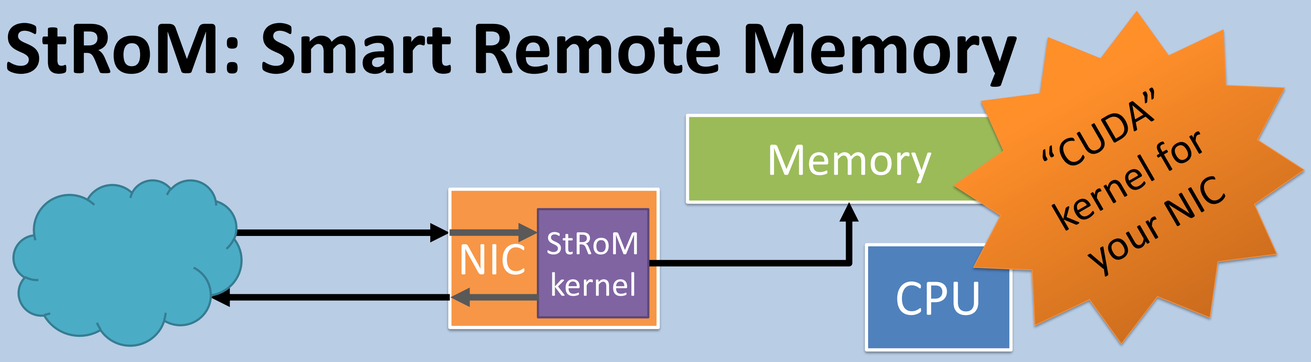
Performance-Complexity trade-off

One-sided (Direct Access)



- **Remote CPU not involved**
- **At least 2 RTs necessary**
- **Handling of misses costly**

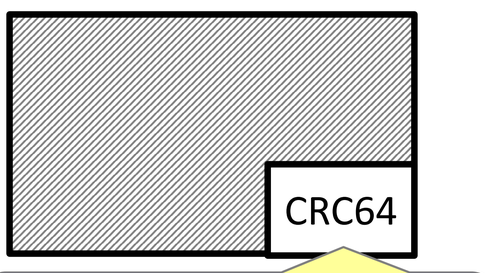
StRoM: Smart Remote Memory



Deployment of acceleration kernels on the NIC

- Invoke one-sided RPCs on the remote NIC
- On-the-fly data processing when transmitting/receiving

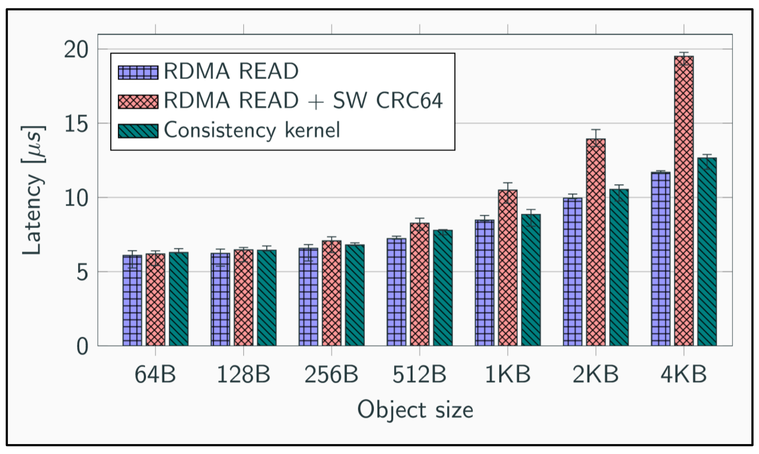
Accelerating Data Access: Consistent Object Retrieval



Calculate CRC64 to verify consistency of data object

Atomictiy on x86 at cache-line granularity. When retrieving large objects over one-sided RDMA, consistency not guaranteed.

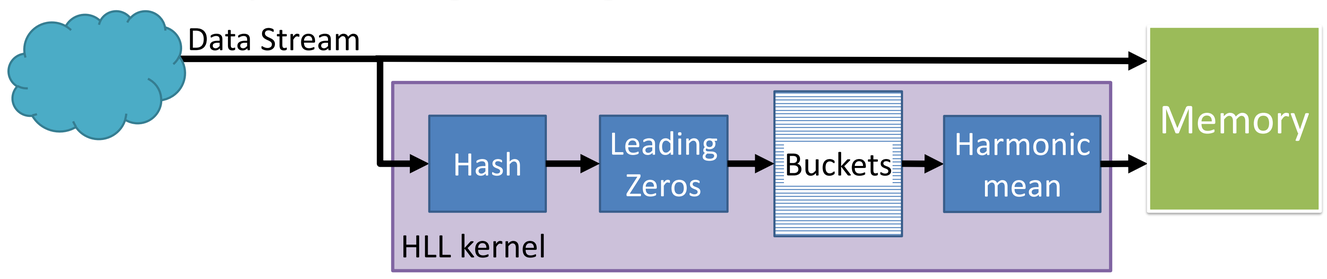
=> Offload consistency check to remote StRoM kernel

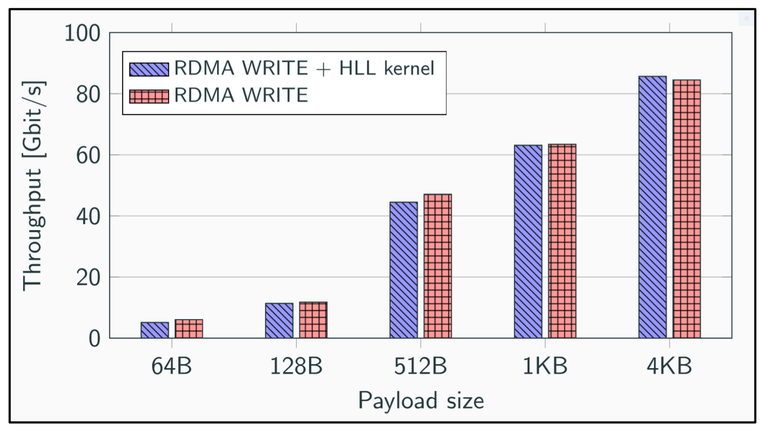


Accelerating Data Processing: Gathering Statistics

HyperLogLog (HLL) kernel to estimate cardinality of the incoming data set

- Cardinality estimation can augment the optimizer in data processing systems
- On-the-fly statistics gathering





Acknowledgements: We would like to thank Xilinx for their generous equipment donation.

github.com/fpgasystems/fpga-network-stack