High-Performance Big Data

EC-Bench: Benchmarking Onload and Offload Erasure Coders on Modern Hardware Architectures

Haiyang Shi, Xiaoyi Lu, and Dhabaleswar K. (DK) Panda

{shi.876, lu.932, panda.2}@osu.edu

The Ohio State University

Bench 2018

Outline

- Introduction
- EC-Bench
- Evaluation
- Conclusion and Future Work

Fault Tolerance & Replication

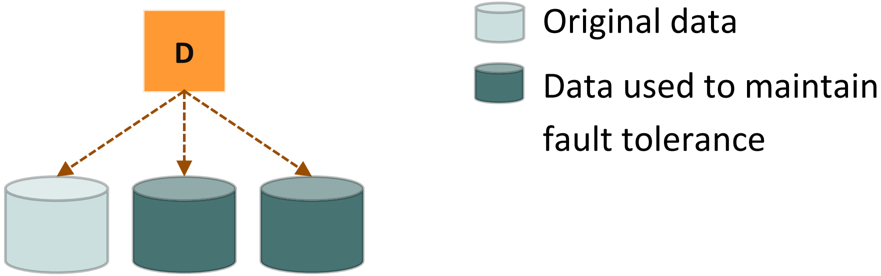
Fault tolerance is a critical requirement for distributed storages

- Availability: data is still accessible under failures
- Durability: no data corruptions under failures

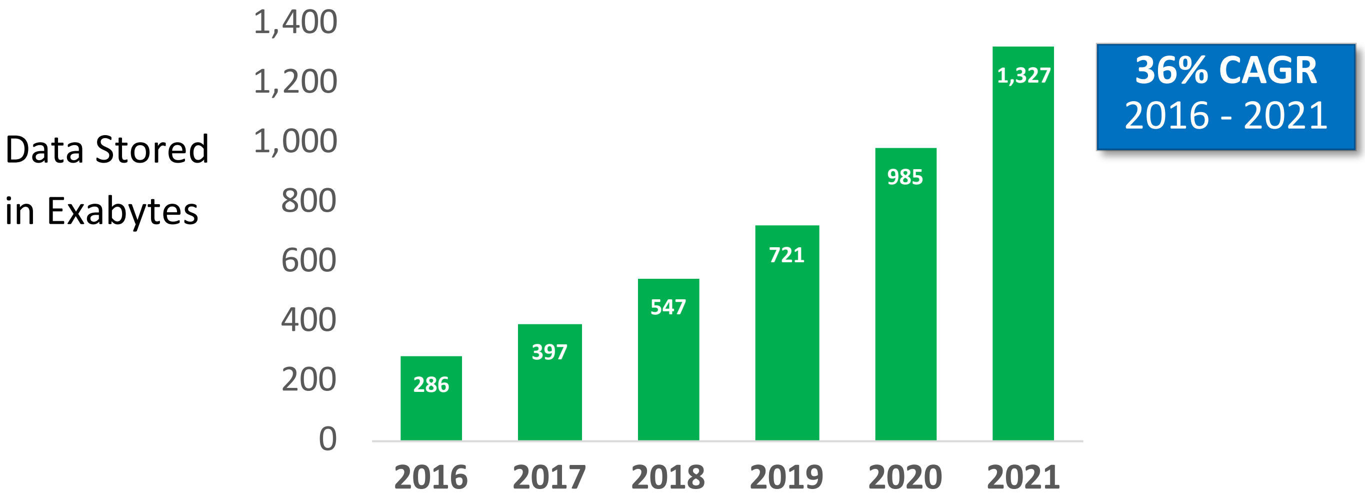
Traditional storage scheme to achieve fault tolerance is replication

- Common replication factors: 3 - 10

Replication factor = 3



Booming of Data



Source: Cisco Global Cloud Index, 2016 - 2021

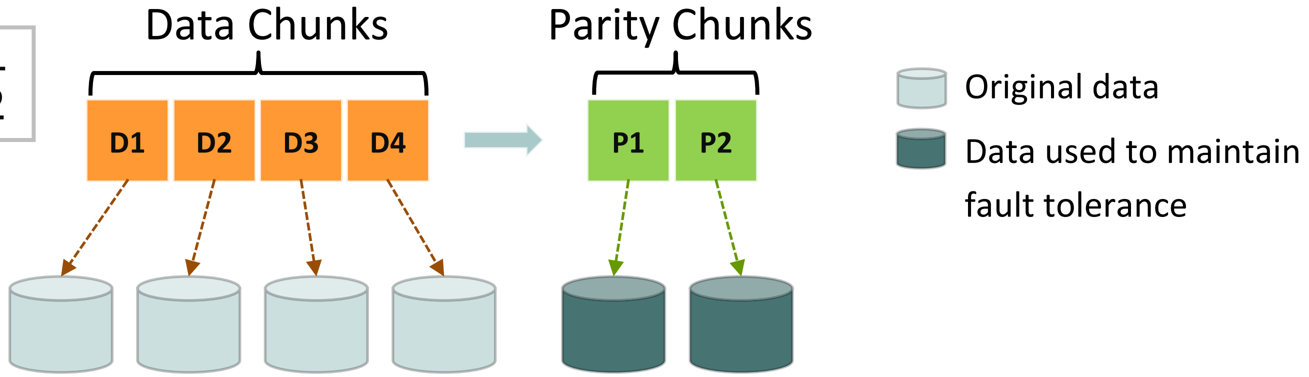
- The booming of data makes storage overhead of replication considerable

Erasure Coding

Erasure Coding is a promising redundancy storage scheme

- Minimize storage overhead by applying erasure encoding on data
- Deliver higher reliability with same storage overhead than replication
- Employed in Google, Microsoft, Facebook, Baidu, etc.
  - ❖ Microsoft Azure reduces storage overhead from 3x (3-way replication) to 1.33x (erasure coding)

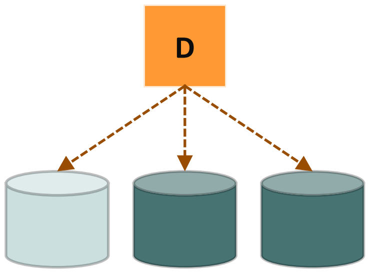
- k = 4
- m = 2



Replication vs. Erasure Coding

Replication

Storage overhead
= 2/1 = 2



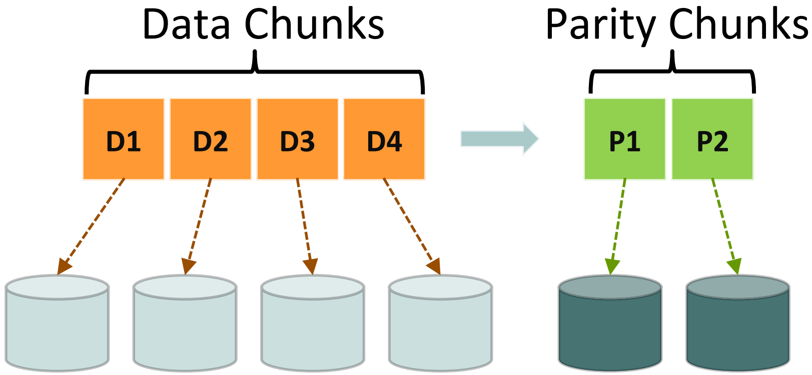
Replication
factor = 3

Original data

Data used to maintain
fault tolerance

Erasure Coding

Storage overhead
= 2/4 = 0.5



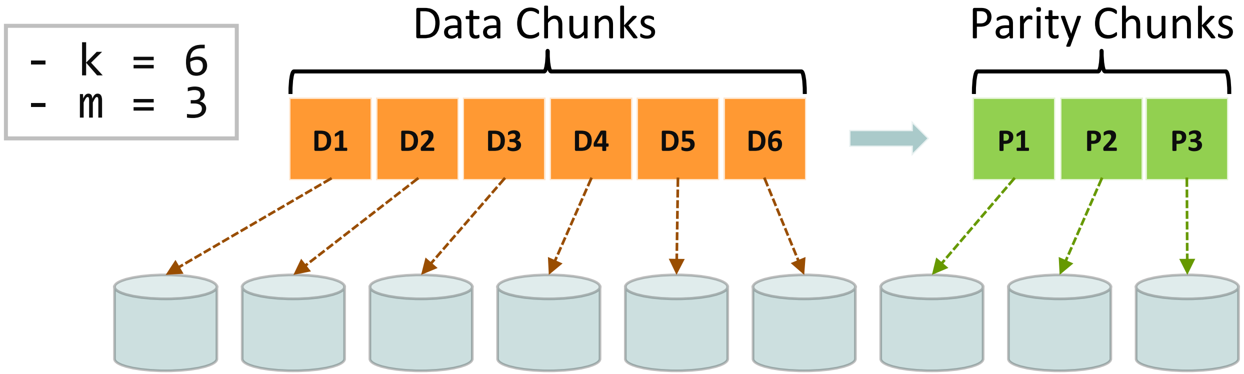

```
- k = 4
- m = 2
```

Original data

Data used to maintain
fault tolerance

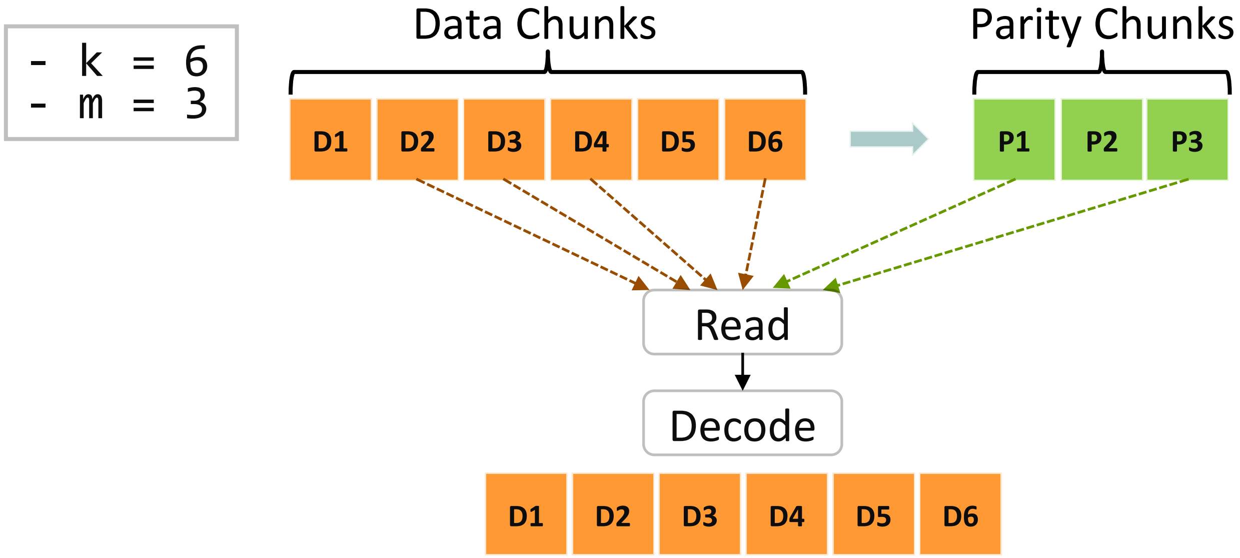
Erasure Coding: Encoding

- Takes in ***k*** data chunks and generates ***m*** parity chunks
- Distributes *(k + m)* chunks to *(k + m)* independent nodes



Erasure Coding: Decoding

- ***Any k*** of *(k + m)* chunks are sufficient to recover the original *k* data chunks



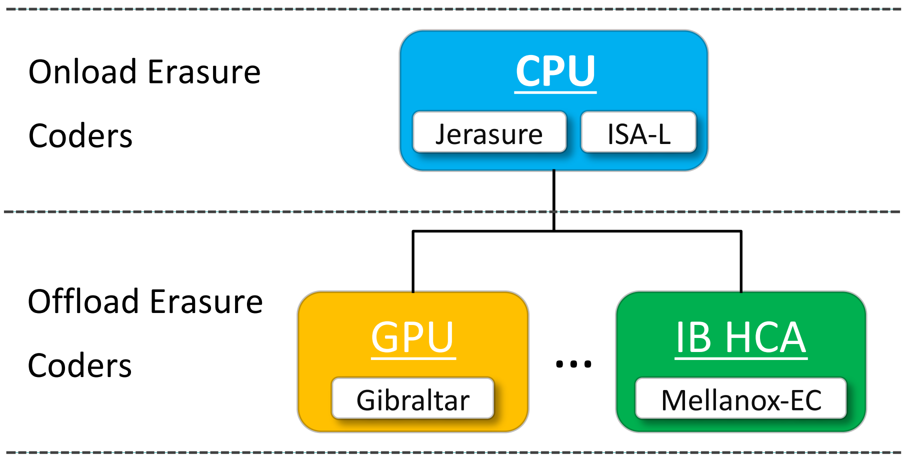
Erasure Coding

- ***Pros:*** Low storage overhead with high fault tolerance
- ***Cons:*** High computation overhead introduced by encode and decode

- High performance hardware-optimized erasure coding libraries to alleviate computation overhead
  - Intel CPUs -> Intel ISA-L
  - Nvidia GPUs -> Gibraltar
  - Mellanox InfiniBand -> Mellanox-EC

Onload and Offload Erasure Coders

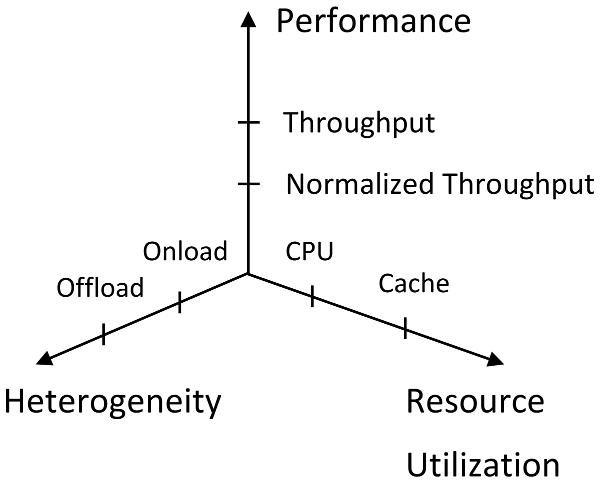
- ***Onload Erasure Coder***
  - Erasure coding operations are performed by host processors
  - Jerasure and Intel ISA-L
- ***Offload Erasure Coder***
  - Erasure coding operations are conducted by accelerators like GPUs and Mellanox InfiniBand HCAs
  - Gibraltar and Mellanox-EC



Our Contributions

- EC-Bench: An unified benchmark suite to benchmark, measure, and characterize onload and offload erasure coders

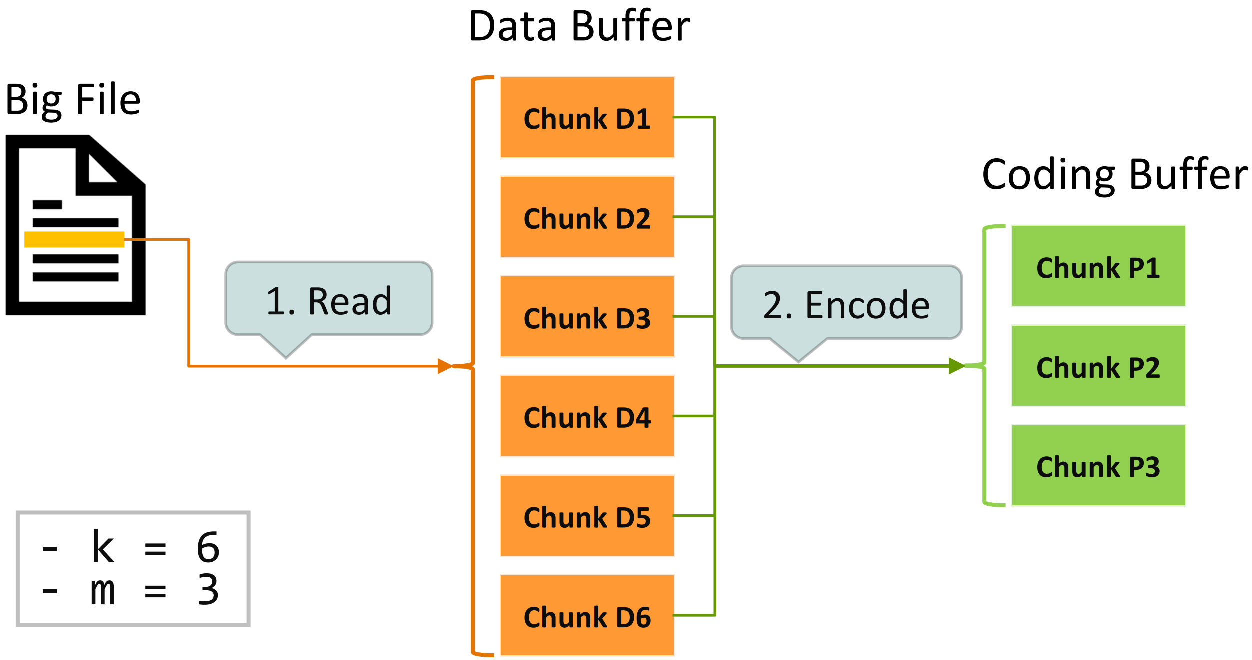
- Evaluations on four popular open source erasure coders with EC-Bench
  - Jerasure
  - Intel ISA-L
  - Gibraltar
  - Mellanox-EC



Outline

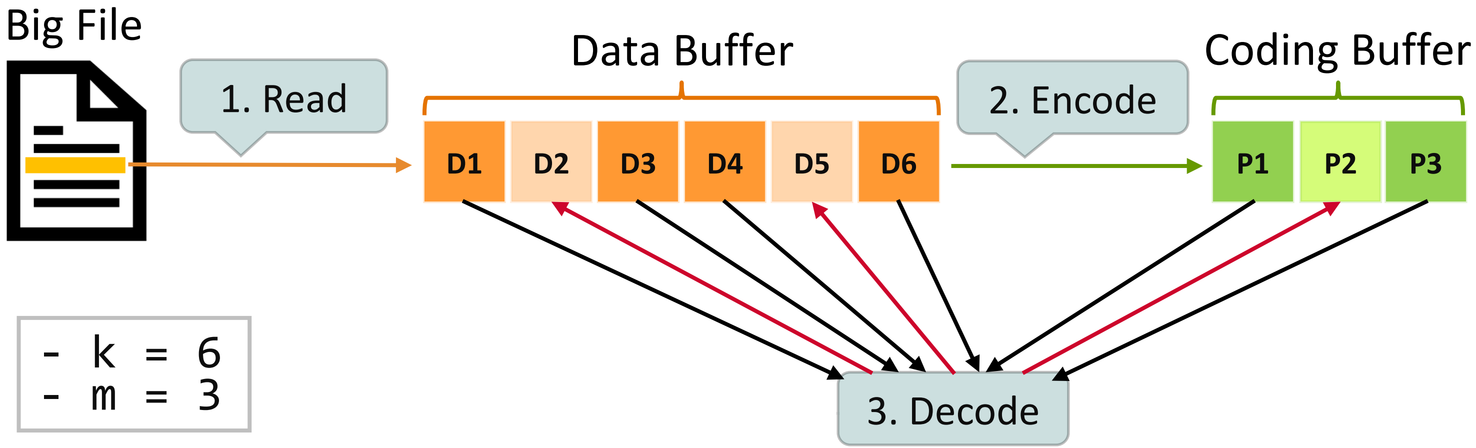
- Introduction
- EC-Bench
- Evaluation
- Conclusion and Future Work

EC-Bench: Encoding Benchmark



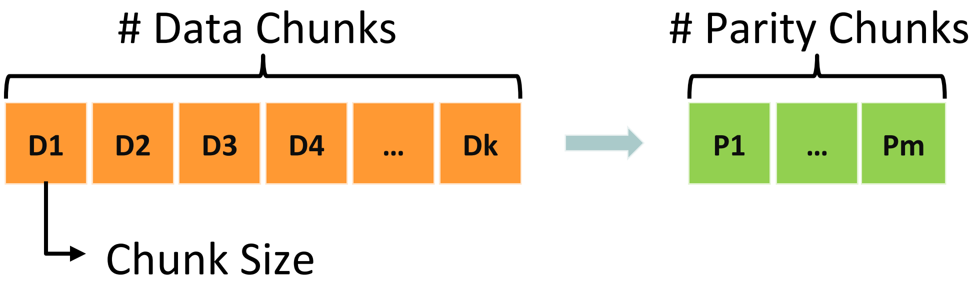
- k = 6
- m = 3

EC-Bench: Decoding Benchmark



- k = 6
- m = 3

EC-Bench: Parameter Space



- k – the number of data chunks
- m – the number of parity chunks
- c – the size of each chunk

EC-Bench: Metrics

- Throughput

$$Thr = \frac{(k+m)\ \cdot\ c}{t}$$

- Normalized Throughput
  - Enable to compare the performance across different configurations
  - Previous studies have demonstrated that optimal erasure codes take k – 1 XOR operations to generate one byte

$$Thr_{\ norm} = \frac{(k\ -1)\ \cdot\ m\ \cdot\ c}{t} = \frac{(k\ -1)\ \cdot\ m}{k+m}\ \cdot\ Thr$$

EC-Bench: Metrics

- CPU Utilization

$$CPU\ Utilization = \frac{CPU\ Cycles}{t}$$

- Cache Pressure

$$Cache\ Pressure = \frac{L1\ Cache\ Misses}{t}$$

Outline

- Introduction
- EC-Bench
- Evaluation
- Conclusion and Future Work

Evaluation: Open Source Libraries

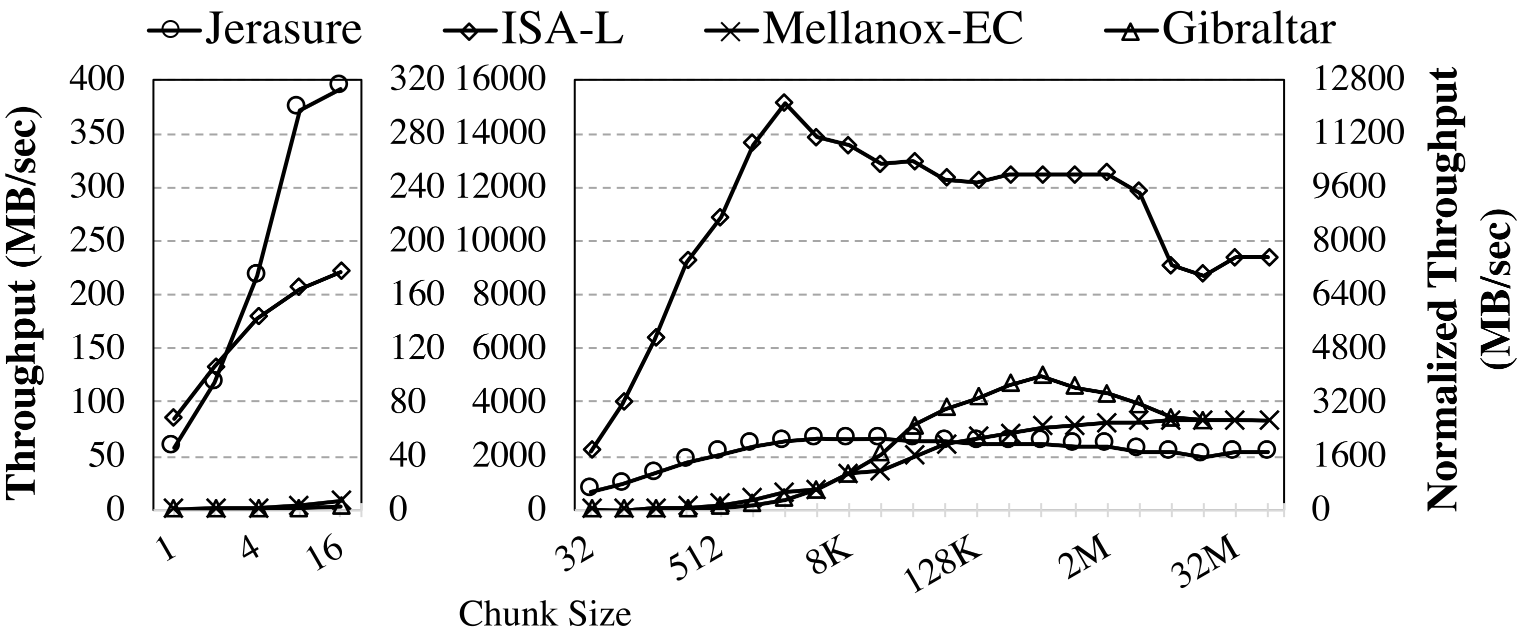
Erasure Coder	Specific Hardware Support	Description
Jerasure	CPU	Jerasure is a CPU-based library released in 2007 that supports a wide variety of erasure codes. Compiled without SSE support.
Intel ISA-L	CPU with SSE/AVX	Intel Intelligent Storage Acceleration Library (ISA-L) is a collection of optimized low-level functions including erasure coding. The erasure coding functions are optimized for Intel instructions, such as Intel SSE, vector and encryption instructions.
Gibraltar	GPU	Gibraltar is a GPU-based library for Reed-Solomon coding.
Mellanox-EC	IB NIC with EC Offload	Mellanox-EC proposed by Mellanox is an HCA-based library for Reed-Solomon coding.

Evaluation: Experimental Setup

- 2.40 GHz Intel(R) Xeon(R) CPU E5-2680 v4
  - 28 cores, 32KB L1 cache, 256KB L2 cache, and 35MB L3 cache
- 128GB DRAM
- Nvidia K80 GPU
- Mellanox ConnectX-5 IB-EDR (100 Gbps) NIC

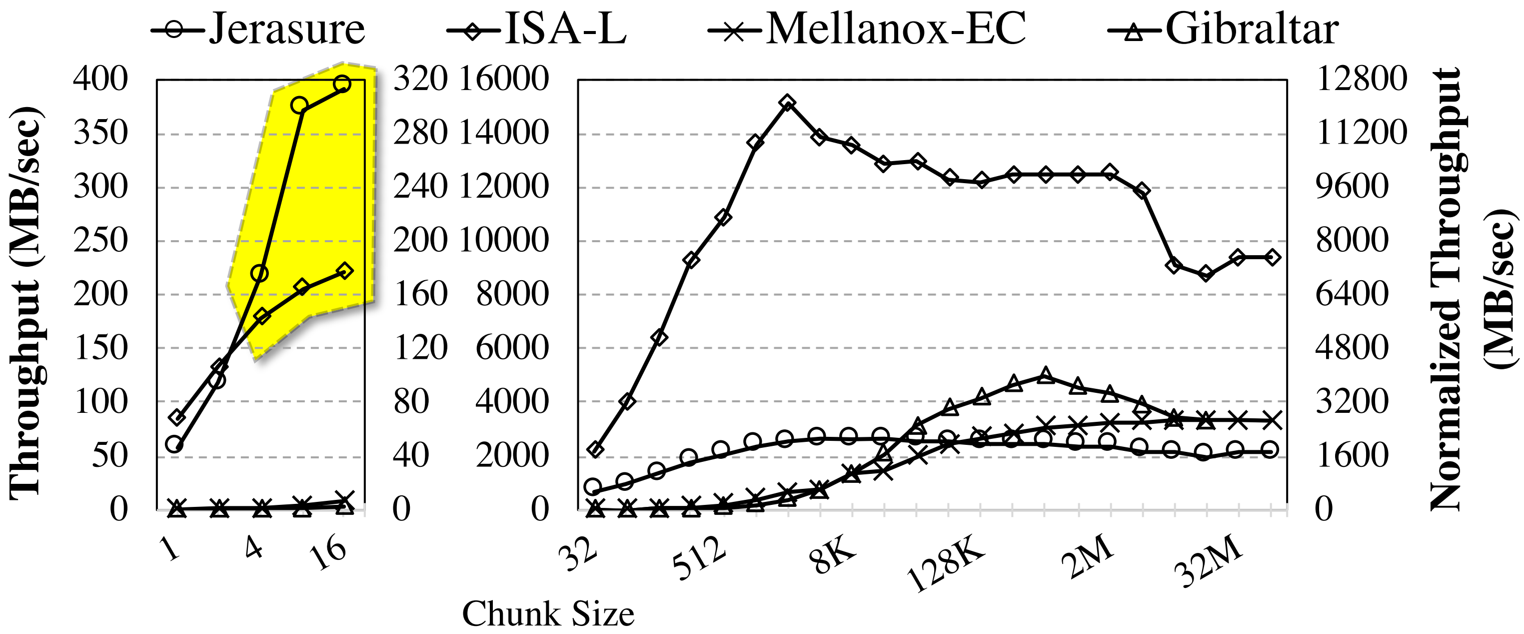
- Explored Configurations (RS(k, m))
  - *RS*(3,2), *RS*(6,3), *RS*(10,4) and *RS*(17,3)

Evaluation: Experimental Results



Throughput Performance with Varied Chunk Sizes for RS(3, 2)

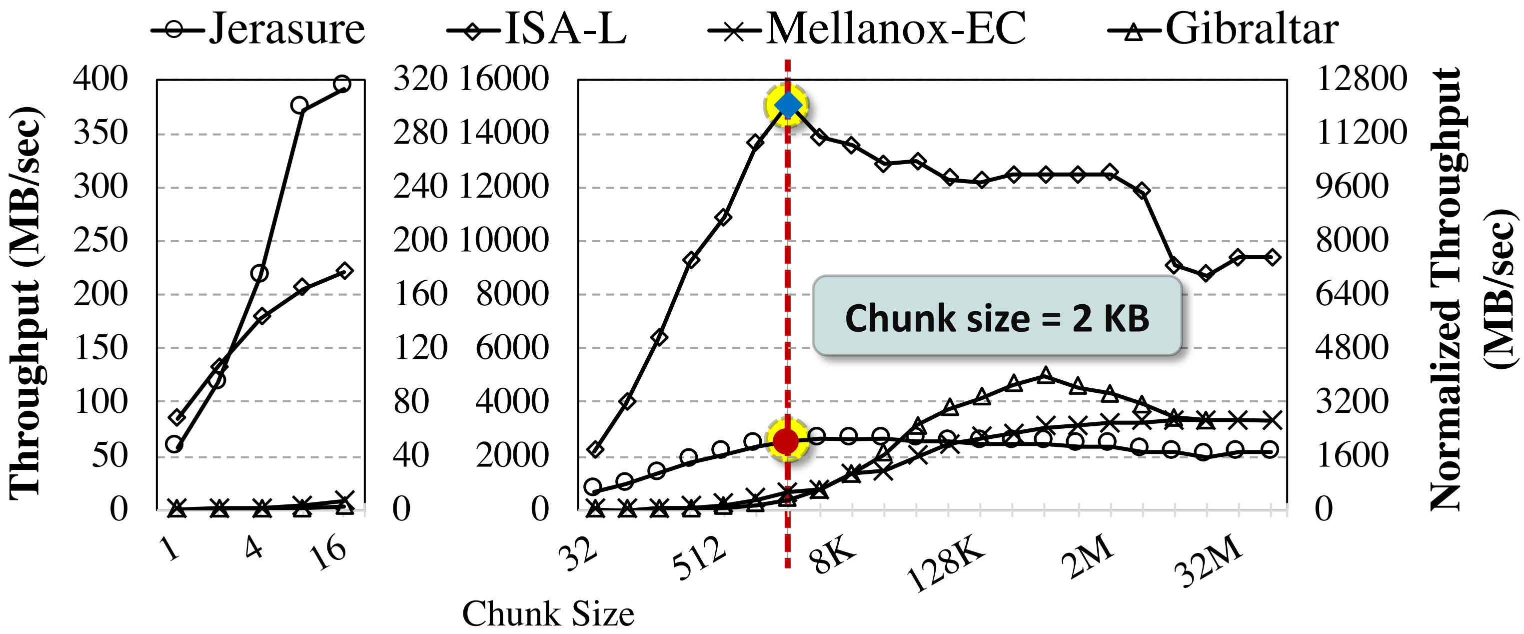
Evaluation: Experimental Results



Throughput Performance with Varied Chunk Sizes for RS(3, 2)

- For small chunk sizes (< 32B), Jerasure performs better than Intel ISA-L

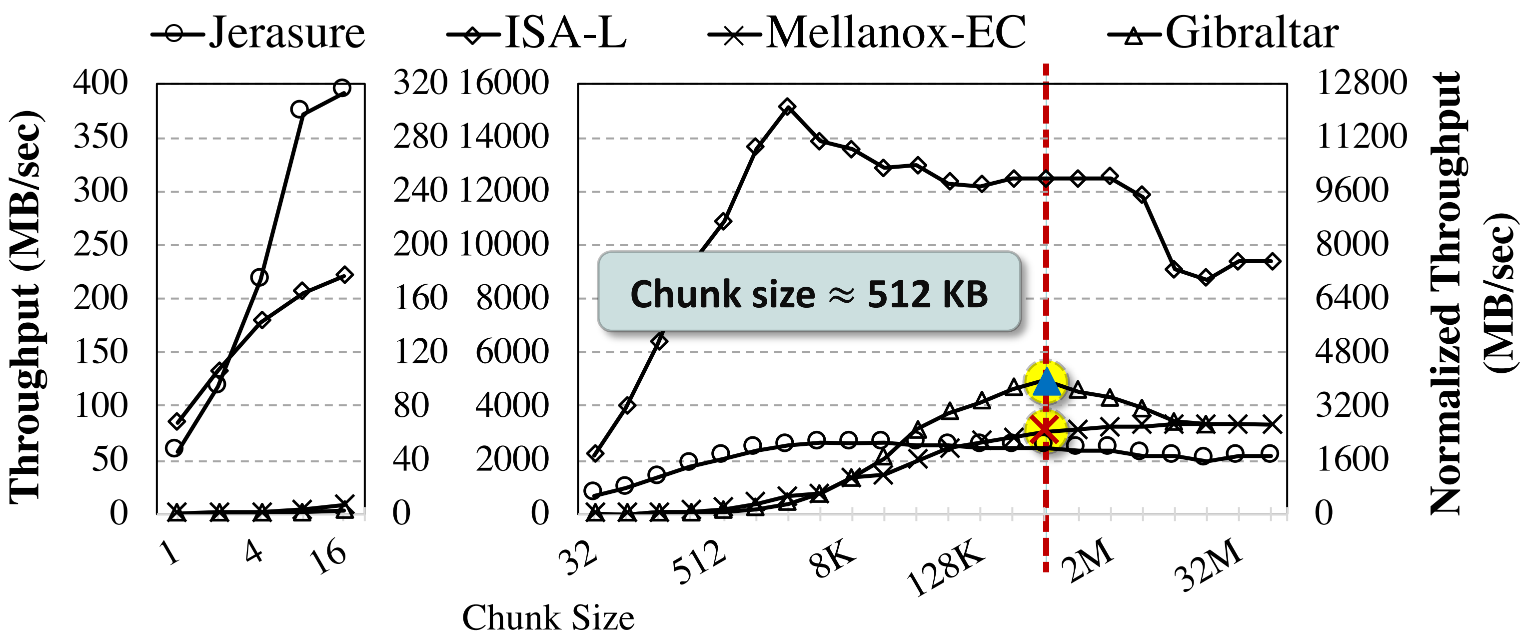
Evaluation: Experimental Results



Throughput Performance with Varied Chunk Sizes for RS(3, 2)

- For both onload coders, the best chunk size to carry out is ***2 KB***

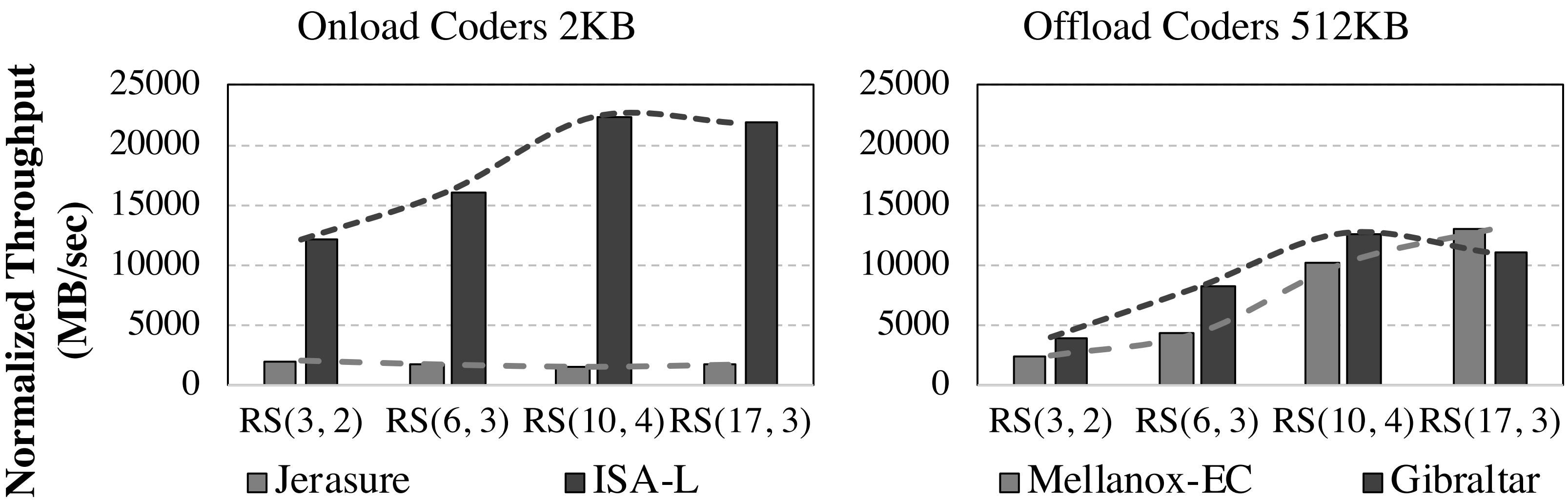
Evaluation: Experimental Results



Throughput Performance with Varied Chunk Sizes for RS(3, 2)

- For both offload coders, the best chunk size to carry out is ***512 KB***
- Because of data transformation overhead

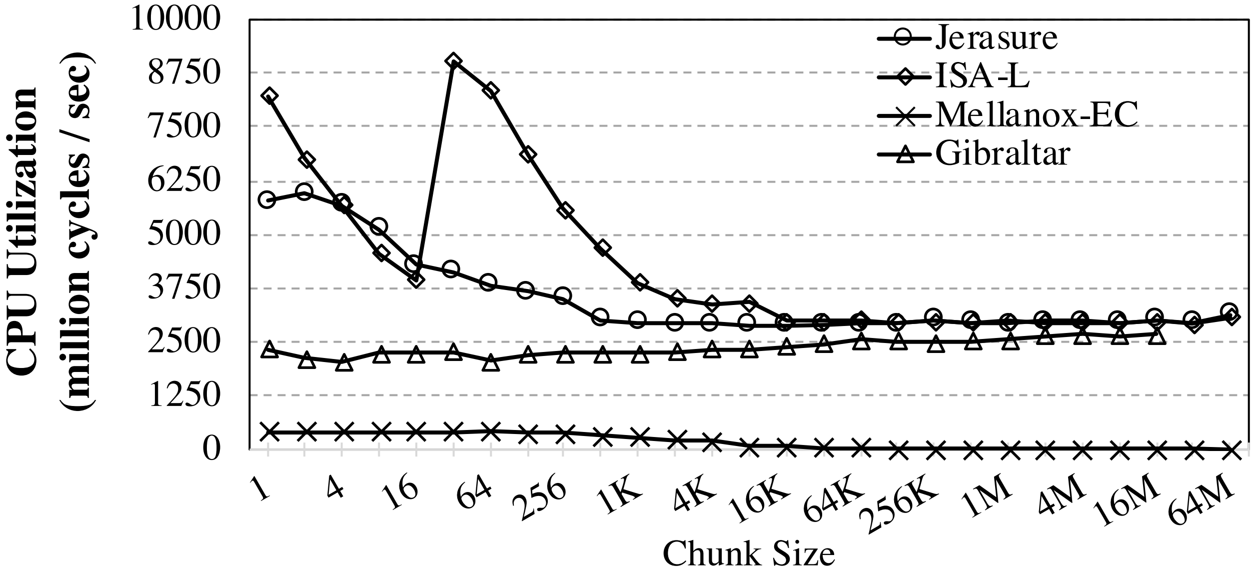
Evaluation: Experimental Results



Normalized Throughput Performance of Onload and Offload Coders

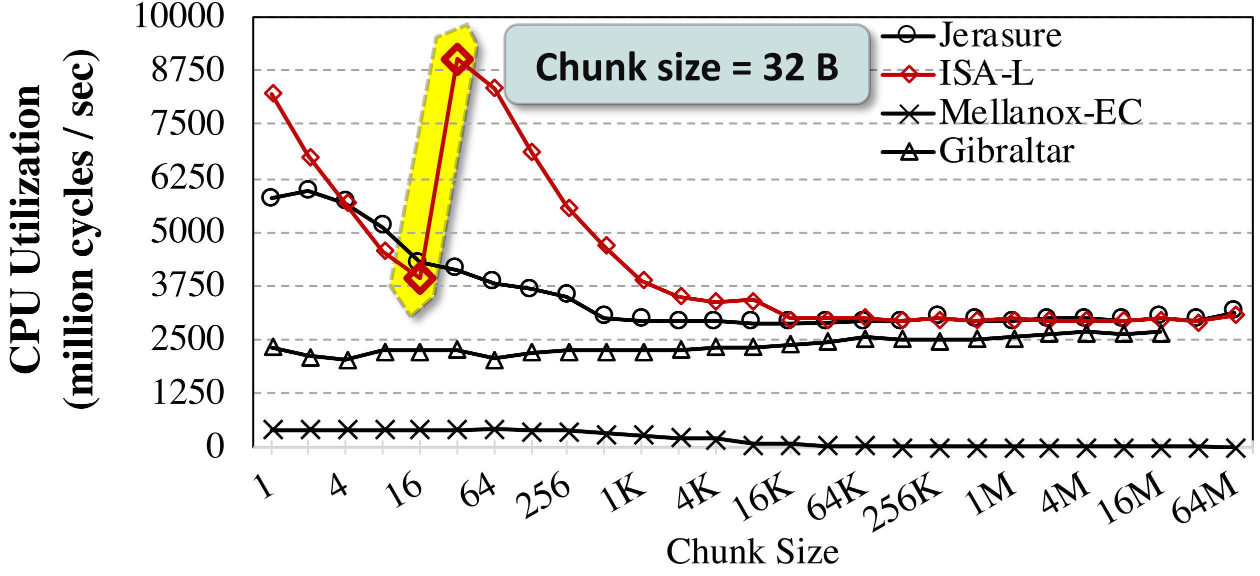
- RS(10, 4) is the best configuration for Intel ISA-L and Gibraltar
- Mellanox-EC performs the best with configuration RS(17, 3)
- Jerasure with RS(3, 2) outperforms other configurations slightly

Evaluation: Experimental Results



CPU Utilization with Varied Chunk Sizes for RS(6, 3)

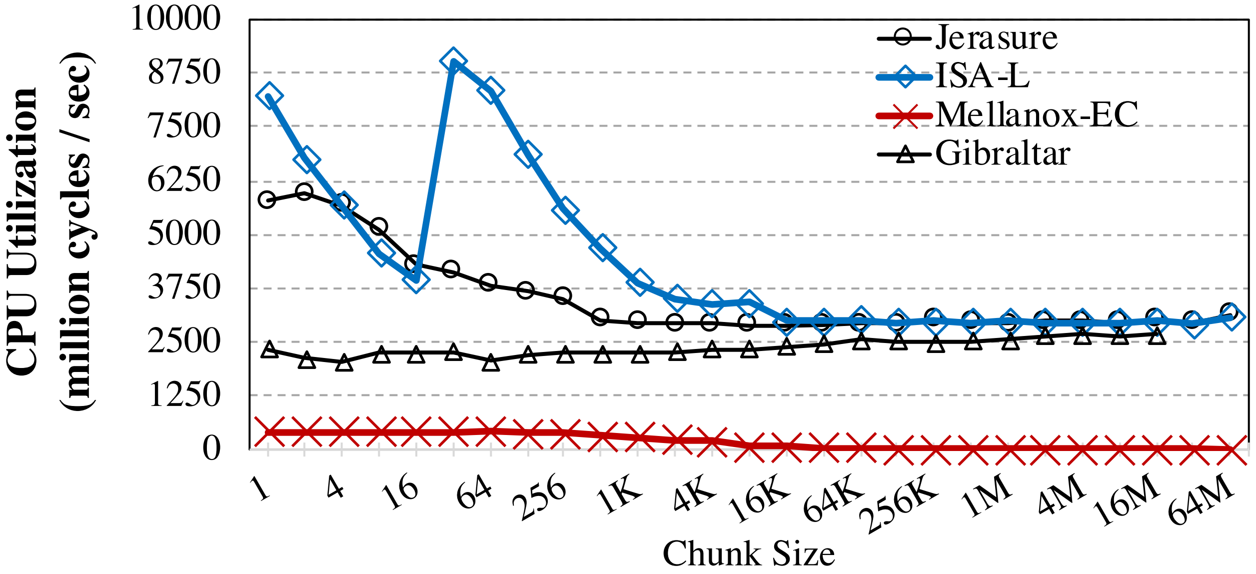
Evaluation: Experimental Results



CPU Utilization with Varied Chunk Sizes for RS(6, 3)

- In Intel ISA-L, different approaches for < 32 bytes and ≥ 32 bytes (details in the function *ec_encode_data_avx2*)

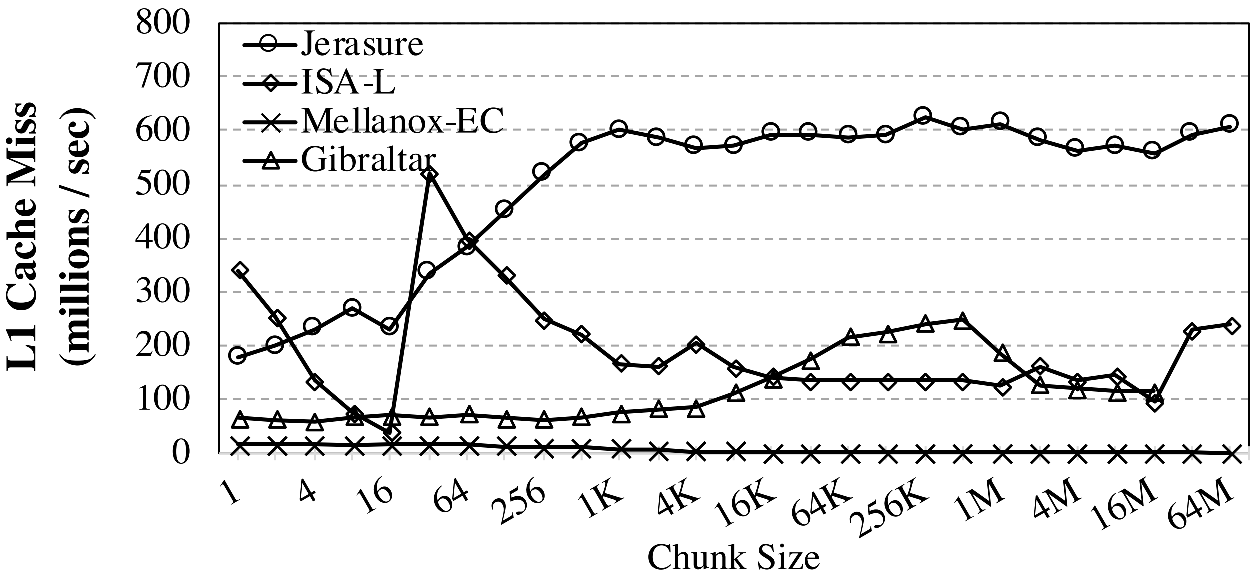
Evaluation: Experimental Results



CPU Utilization with Varied Chunk Sizes for RS(6, 3)

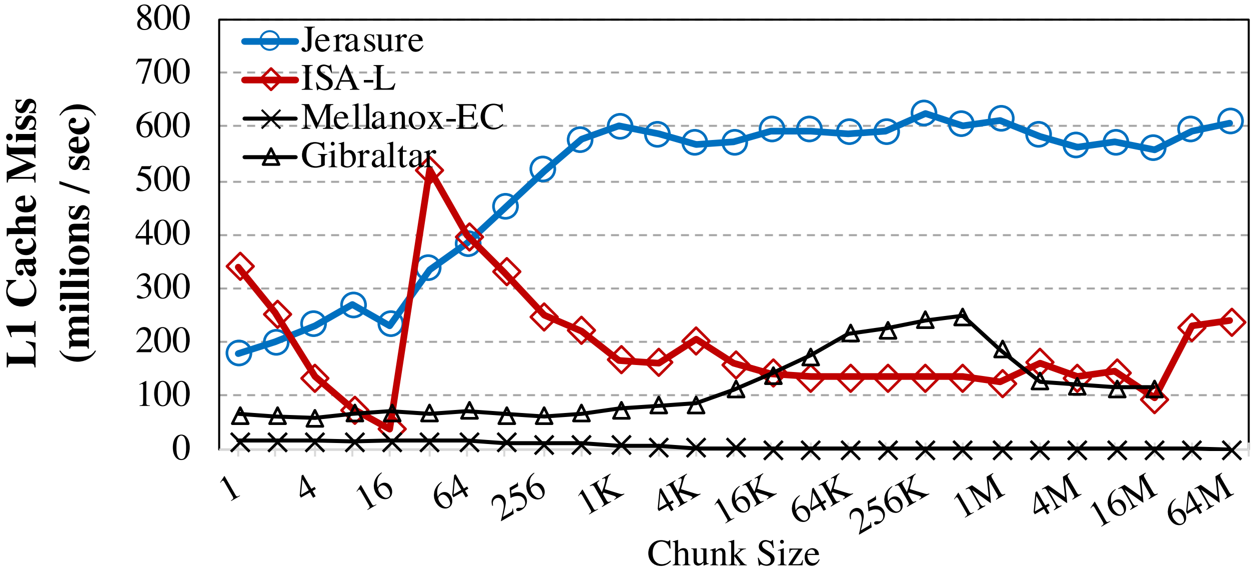
- Offload coders make better use of CPU cycles, thus have less impact on applications' computation
  - With chunk size = 64 MB, 0.41 million cycles by Mellanox-EC vs. 2932.23 million cycles by ISA-L

Evaluation: Experimental Results



Cache Pressure with Varied Chunk Sizes for RS(10, 4)

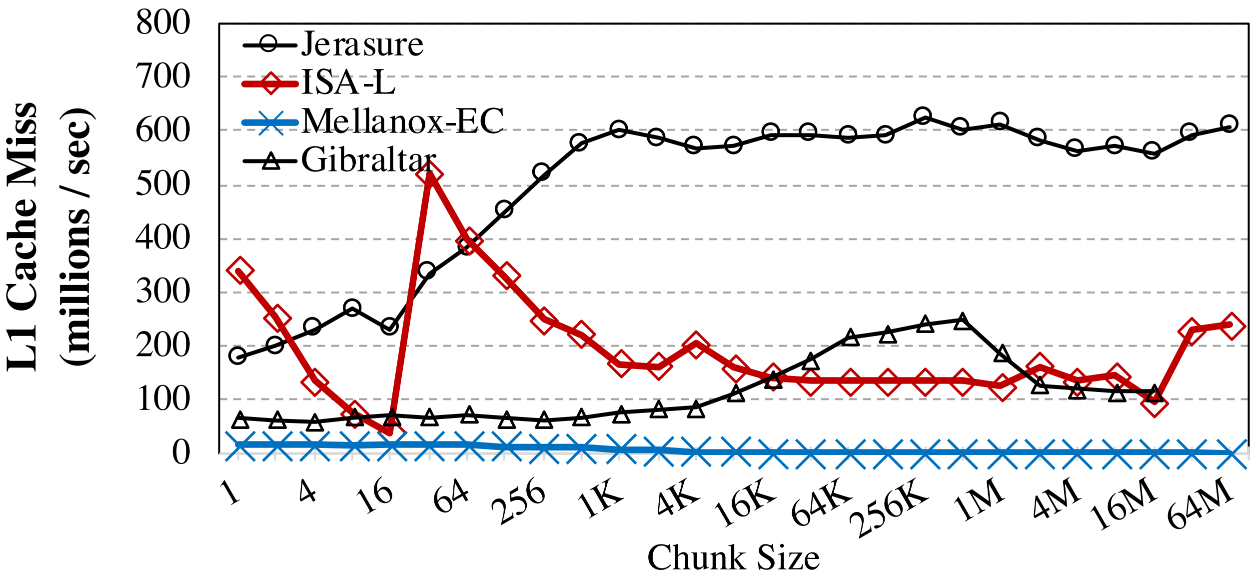
Evaluation: Experimental Results



Cache Pressure with Varied Chunk Sizes for RS(10, 4)

- Within onload coders, Intel ISA-L makes better use of L1 cache compared to Jerasure

Evaluation: Experimental Results



Cache Pressure with Varied Chunk Sizes for RS(10, 4)

- In general, offload coders make less pressure on L1 cache, thus have less impact on applications' cache usage

Outline

- Introduction
- EC-Bench
- Evaluation
- Conclusion and Future Work

Conclusion and Future Work

- EC-Bench: An unified benchmark suite to benchmark, measure, and characterize onload and offload erasure coders

- Evaluations on four popular open source erasure coders with EC-Bench
  - Advanced onload coders outperform offload coders
  - Offload coders make better use of CPU cycles and cache

- Future work
  - Support more hardware-optimized erasure coders, e.g., FPGA-optimized erasure coders

Thank You!

{shi.876, lu.932, panda.2}@osu.edu

Network-Based Computing Laboratory

http://nowlab.cse.ohio-state.edu/

The High-Performance Big Data Project

http://hibd.cse.ohio-state.edu/