Evaluation of Energy-efficient Real-time Data Compression in Edge Computing

Yuta Shimizu †, Van An Le§, Yusuke Tanimura§, Yiyu Tan†
†Faculty of Science and Engineering, Iwate University, Japan
§ National Institute of Advanced Industrial Science and Technology, Japan

Introduction

Edge computing has become important and enabled applications such as real-time data processing in IoT. In Edge computing, a big challenge is the **huge amounts of time-series data**, such as audio and images, produced by IoT devices, which will **increase storage and power consumption in edge servers**. Nowadays, data compression is a critical technique to address this problem. Different from prior research, which mainly focused on raw throughput and overlooked critical metrics such as energy efficiency and real-time responsiveness in multi-task edge environments, this research quantitatively **evaluates the performance and limitations of data compression in an edge server**, and demonstrates that multi-core processors may not meet contemporary requirements for real-time data compression in edge computing.

Methods

- Modern **data compression algorithms**, including deflate, gzip, snappy, zstd, and lz4 [1], were implemented and their performance was evaluated on an edge node with an Intel i7-6950X (20 threads) processor, and their performance, including compression ratio, power efficiency, and latency, was evaluated.
- **Datasets**: CIFAR-10 (image) [2] and ESC-50 (audio) [3]
- **Experiments:**
  - **Core scalability** to explore the relation between compression performance and the number of cores.
  - **Trade-off at compression level** to study the trade-off between compression ratio and energy efficiency.
  - **Real-time performance under load contention** to study the latency of real-time data compression in the case of multiple tasks being executed simultaneously.

Evaluation Results

- **Core scalability.** The compression throughput of the algorithms snappy and zstd (level 3) in the case of different threads is evaluated (Figure 1).
  - The parallelism of CPU is saturated as the core count is increased.
  - The ultra-fast algorithm snappy exhibits a severe bottleneck, with throughput increasing by only about 2.1% (from 1442.43 MB/s to 1472.76 MB/s) in the audio and dropping by 6.2% in the image when scaling from 1 thread to 20 threads, which indicates that the saturation is limited by I/O and memory bandwidth rather than core count.
- **Compression level trade-off.** The compression performance of the algorithm zstd at different compression levels is measured (Table 1).
  - As the compression is increased from level 1 to level 15 (higher level is, more optimization are employed), although compression ratio is improved slightly (3.6% in audio and 2.3% in image), **the energy efficiency is dropped by 83% (from 9.097 MB/s/W to 1.55 MB/s/W) and 76% (from 2.569 MB/s/W to 0.61 MB/s/W)** in the audio and image, respectively.
- **Real-time performance under load contention.** Table 2 shows real-time latency of the compression task in real-world environments, where data compression and other multiple parallel tasks occupied half of CPU resources, respectively.
  - Under load contention, even the fastest algorithm snappy, the P99 latency is degraded by **about 6.5%** (from 1.265ms with full CPU resources to 1.353ms with 50% of CPU resources) in the audio, which is a critical failure for time-sensitive edge applications.

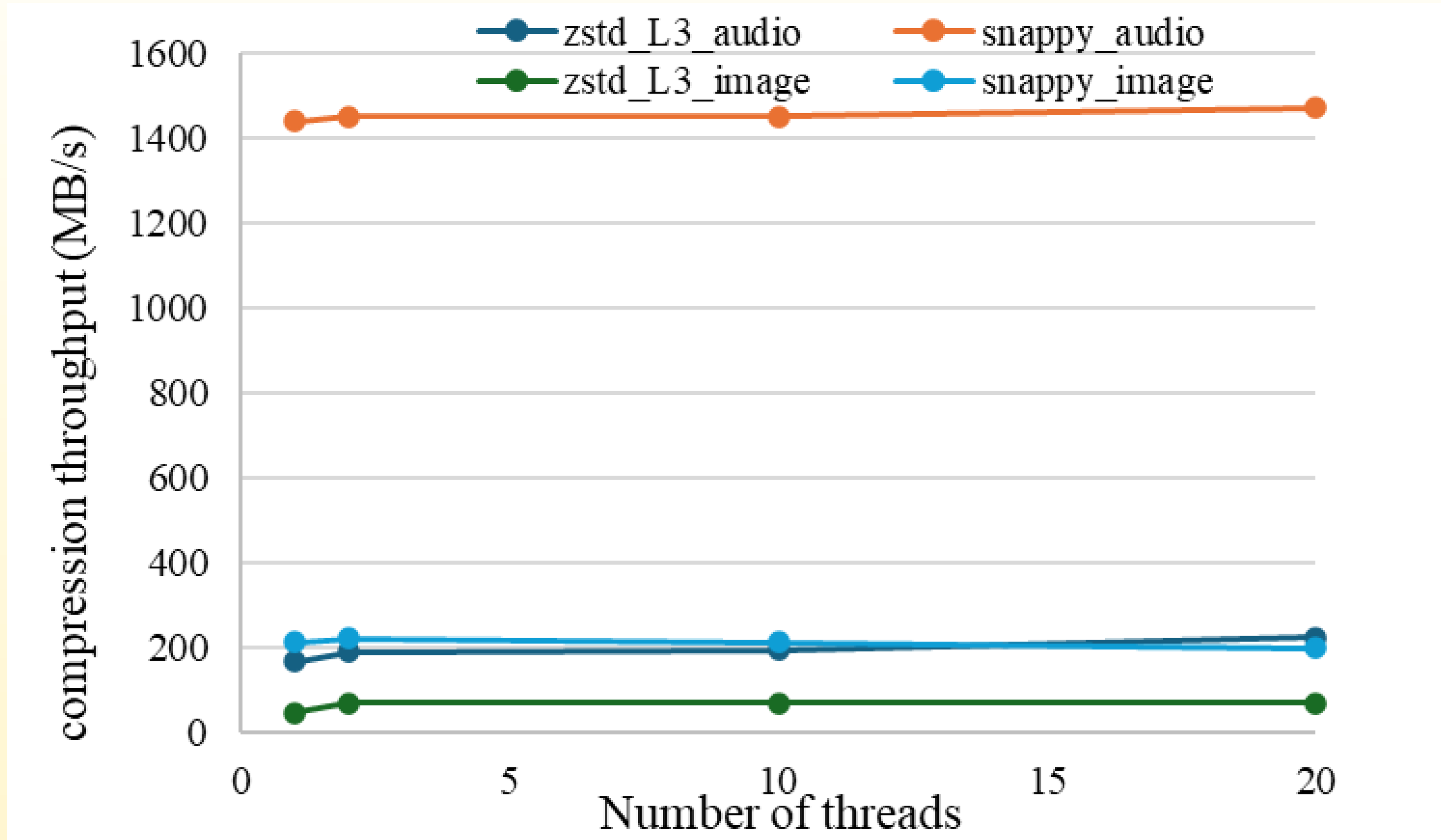

Figure 1: Compression throughput in the zstd and snappy

Table 1: Energy efficiency in the zstd

algorithm mode	audio		image	
	compression ratio	energy efficiency (MB/s/W)	compression ratio	energy efficiency (MB/s/W)
zstd_L1	0.749	9.087	0.870	2.569
zstd_L3	0.735	7.812	0.868	2.68
zstd_L7	0.736	5.225	0.866	1.806
zstd_L10	0.736	4.584	0.866	1.377
zstd_L15	0.713	1.55	0.847	0.61

Table 2 Performance under load contention

Algorithm	Compression ratio		Throughput (MB/s)		p50_latency (ms)		p99_latency (ms)		Energy efficiency (MB/s/W)	
	audio	image	audio	image	audio	image	audio	image	audio	image
deflate_zlib	0.697	0.835	22.973	22.327	4.446	0.054	24.219	0.167	0.895	1.023
gzip	0.724	0.866	23.244	23.495	4.381	0.039	23.937	0.091	0.882	0.88
snappy	0.808	0.864	1035.61	195.8	0.113	0.005	1.353	0.011	40.13	7.378
zstd	0.726	0.868	223.026	72.732	0.326	0.015	4.747	0.035	8.094	2.762
lz4	0.856	0.923	968.381	357.35	0.1	0.003	1.263	0.006	37.641	13.412

Conclusion

Edge servers with multi-core processors have reached their practical limit for real-time data compression in edge computing due to **poor scaling**, **severe efficiency loss in the case of high compression ratio, and latency increase in real-world environment**. To address this, dedicated hardware accelerators like FPGAs and GPUs may be required to achieve high energy efficiency and highly scalable data processing in edge servers.

Reference:
[1] https://github.com/NVIDIA/CUDALibrarySamples/tree/main/nvCOMP
[2] https://www.cs.toronto.edu/~kriz/cifar.html
[3] https://github.com/karolpiczak/ESC-50