Contrasting the Performance of Compression Algorithms on Genomic Data

Cornel Constantinescu, IBM Research Almaden

Outline of the Talk:

- Introduction / Motivation
- Data used in experiments
- General purpose compressors comparison
- Simple Improvements
- Special purpose compression
- Transparent compression – working on compressed data (prototype)
- Parallelism / Multithreading
- Conclusion

Introduction / Motivation

- Despite the large number of research papers and compression algorithms proposed for compressing genomic data generated by sequencing machines, by far the most commonly used compression algorithm in the industry for FASTQ data is *gzip.*

- The main drawbacks of the proposed alternative special-purpose compression algorithms are:
  - *slow speed* of either compression or decompression or both, and also their
  - brittleness by *making various limiting assum*ptions about the input FASTQ format (for example, the structure of the headers or fixed lengths of the records [1]) in order to further improve their specialized compression.

1. Ibrahim Numanagic, James K Bonfield, Faraz Hach, Jan Voges, Jorn Ostermann, Claudio Alberti, Marco Mattavelli, and S Cenk Sahinalp. Comparison of high-throughput sequencing data compression tools. Nature Methods, 13(12):1005–1008, October 2016.

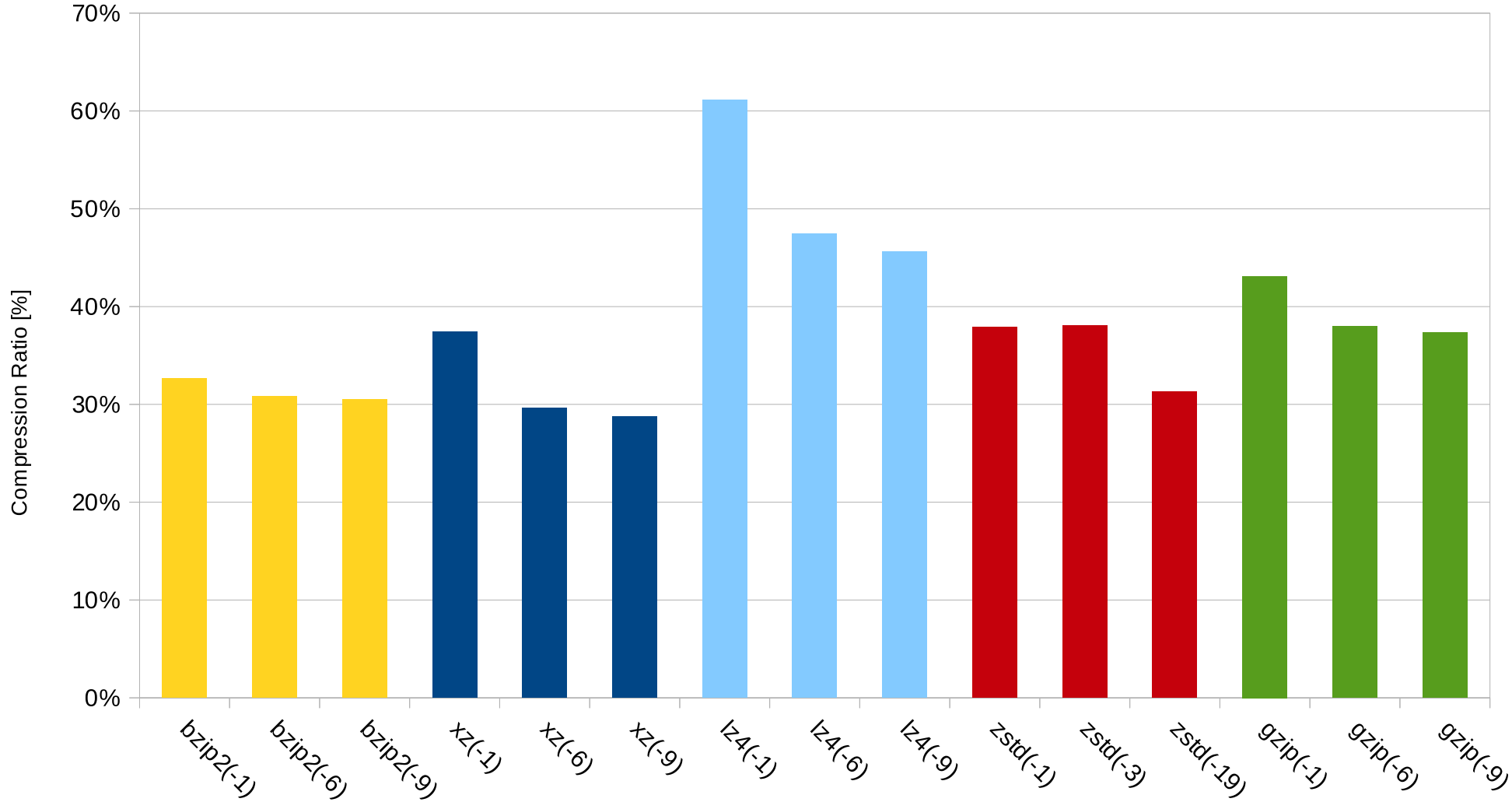
Genomic data used in experiments:

FASTQ/FASTA Data	Source (ftp://ftp.ncbi.nih.gov/1000genomes/ftp/)	Raw Size
ERR317482_1.fastq	https://www.ebi.ac.uk/ena/data/view/ERR317482	6.8 GB
ERR317482_2.fastq	https://www.ebi.ac.uk/ena/data/view/ERR317482	6.8 GB
ERR034549_1.filt.fastq	data/NA12777/sequence_read/ERR034549_1.filt.fastq.gz	17 GB
ERR034549_2.filt.fastq	data/NA12777/sequence_read/ERR034549_2.filt.fastq.gz	17 GB
ERR125594_1.filt.fastq	data/NA21122/sequence_read/ERR125594_1.filt.fastq.gz	13 GB
ERR125594_2.filt.fastq	data/NA21122/sequence_read/ERR125594_2.filt.fastq.gz	13 GB
SRR099453_1.filt.fastq	data/NA11920/sequence_read/SRR099453_1.filt.fastq.gz	22 GB
SRR099453_2.filt.fastq	data/NA11920/sequence_read/SRR099453_2.filt.fastq.gz	22 GB
SRR764769_1.filt.fastq	data/NA21122/sequence_read/SRR764769_1.filt.fastq.gz	8.6 GB
SRR764769_2.filt.fastq	data/NA21122/sequence_read/SRR764769_2.filt.fastq.gz	8.6 GB
human_g1k_v37.fasta	ftp://ftp.broadinstitute.org/bundle/b37/human_g1k_v37.fasta.gz	3.0 GB

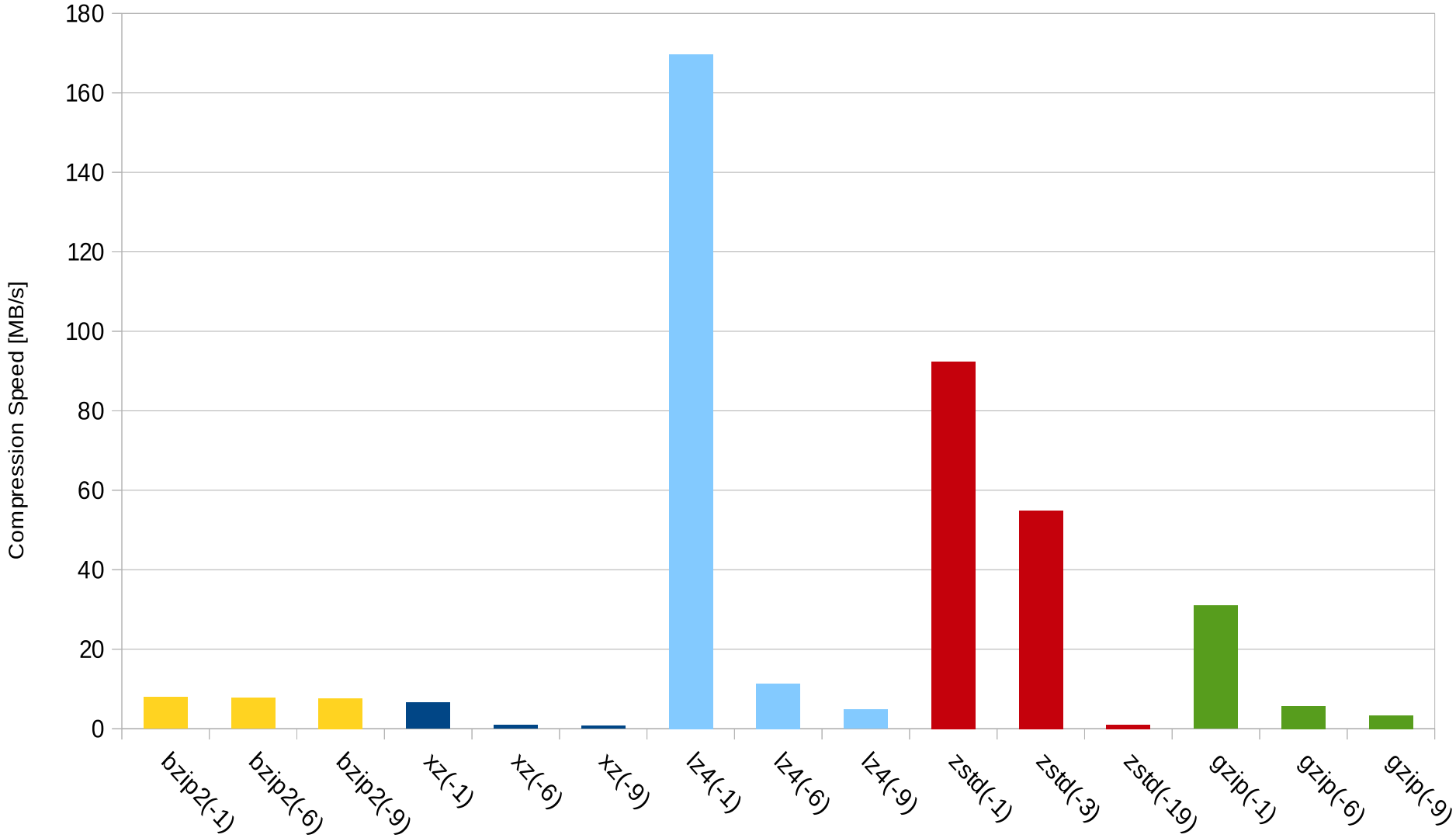
Outline of the Talk:

- Introduction / Motivation
- Data used in experiments
- General purpose compressors comparison
- Simple Improvements
- Special purpose compression
- Transparent compression – working on compressed data
- Parallelism / Multithreading
- Conclusion

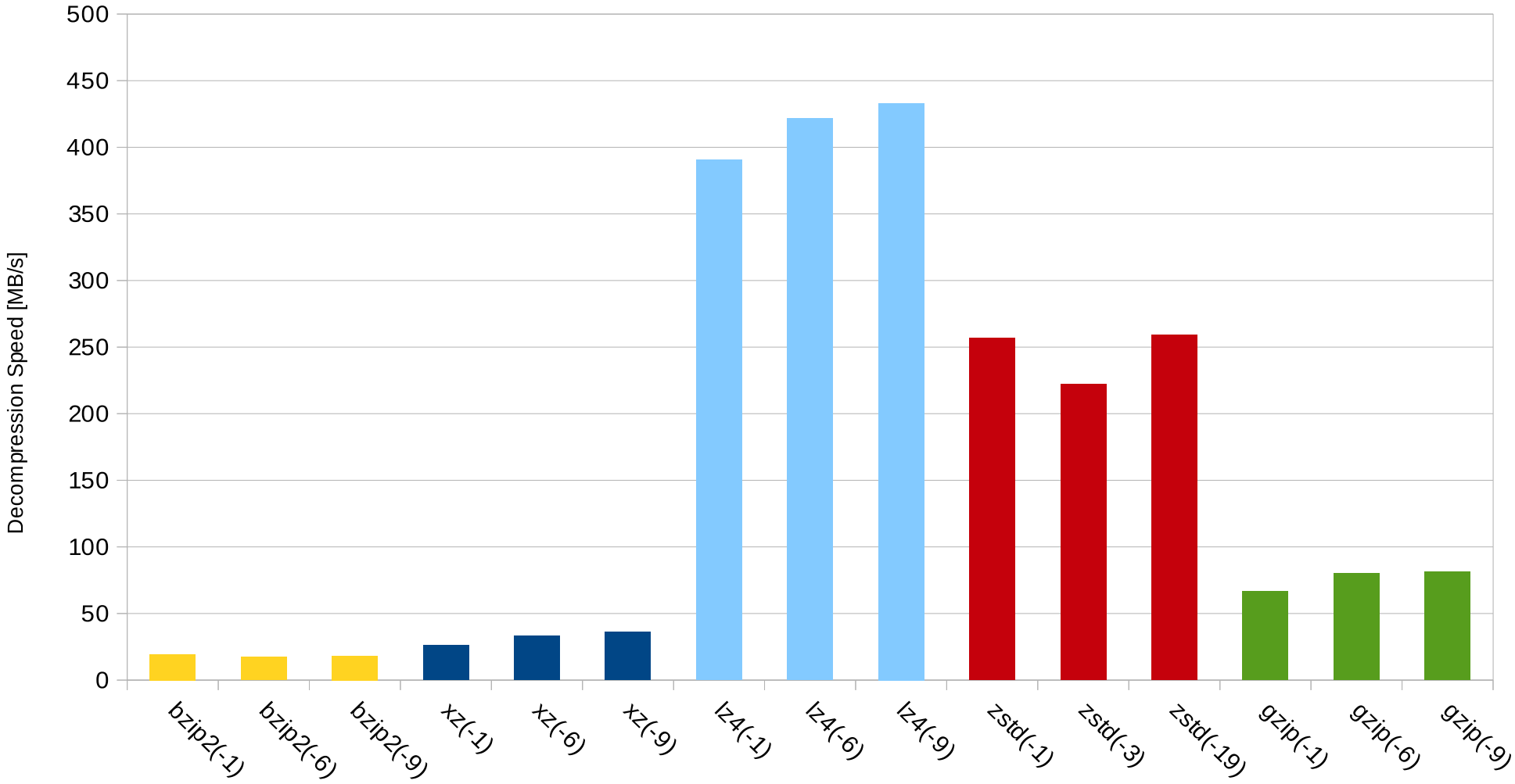
Compression Ratios on ERR317482_1 (6.8GB)



Compression Speeds (MB/s) on ERR317482_1



Decompression Speeds (MB/s) on ERR317482_1

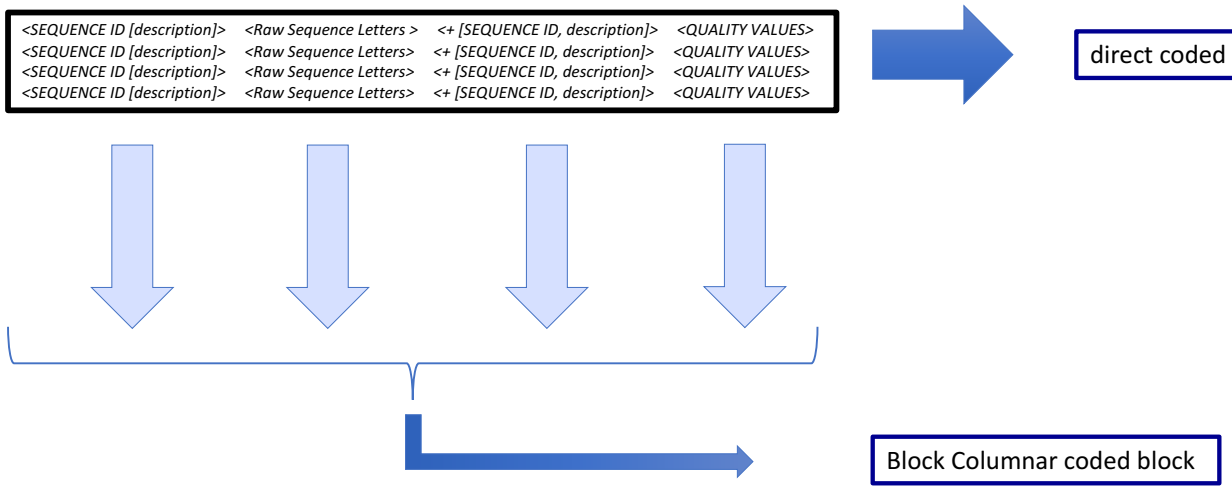


Outline of the Talk:

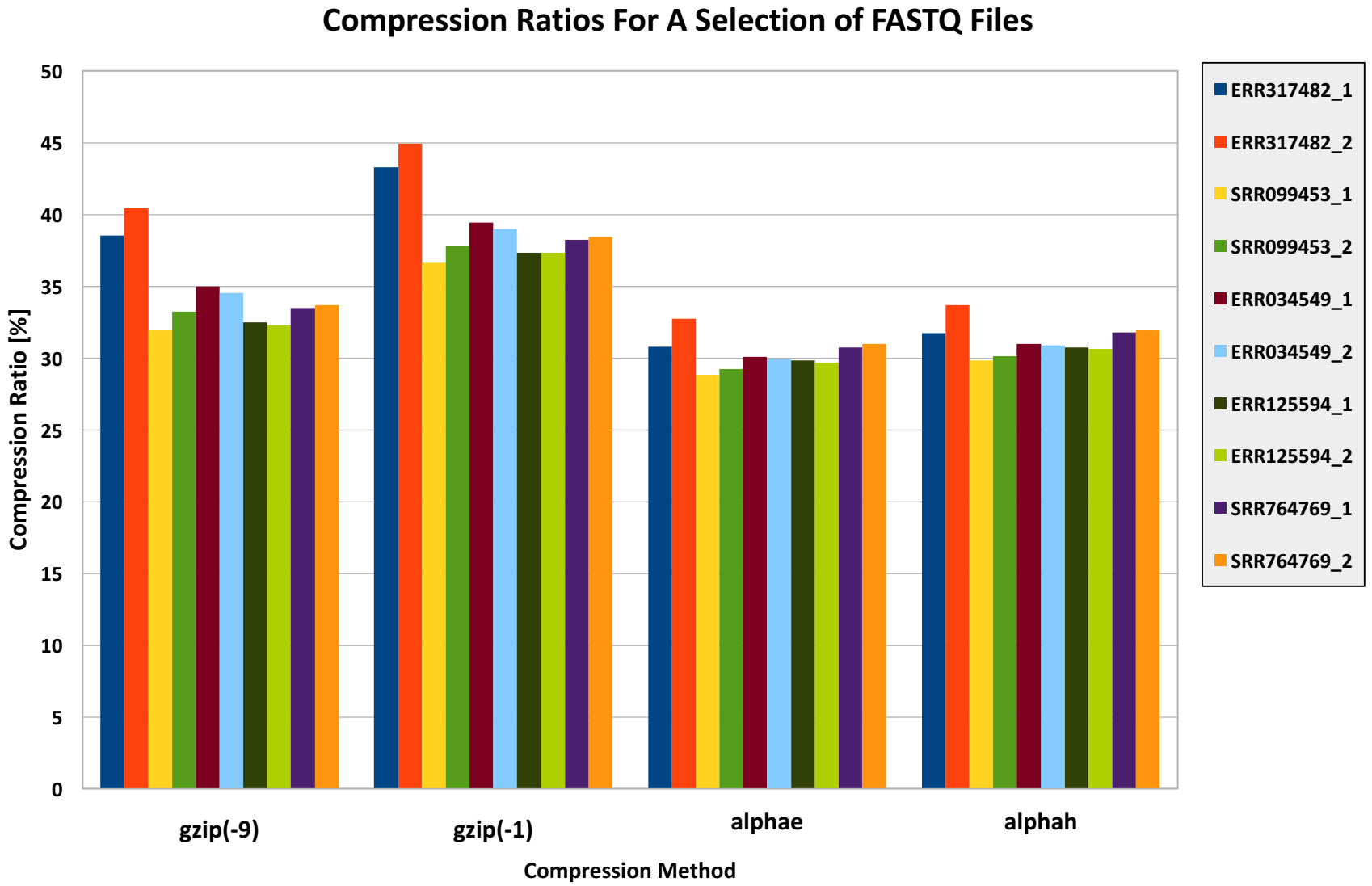
- Introduction / Motivation
- Data used in experiments
- General purpose compressors comparison
- Simple Improvements
- Special purpose compression
- Transparent compression – working on compressed data
- Parallelism / Multithreading
- Conclusion

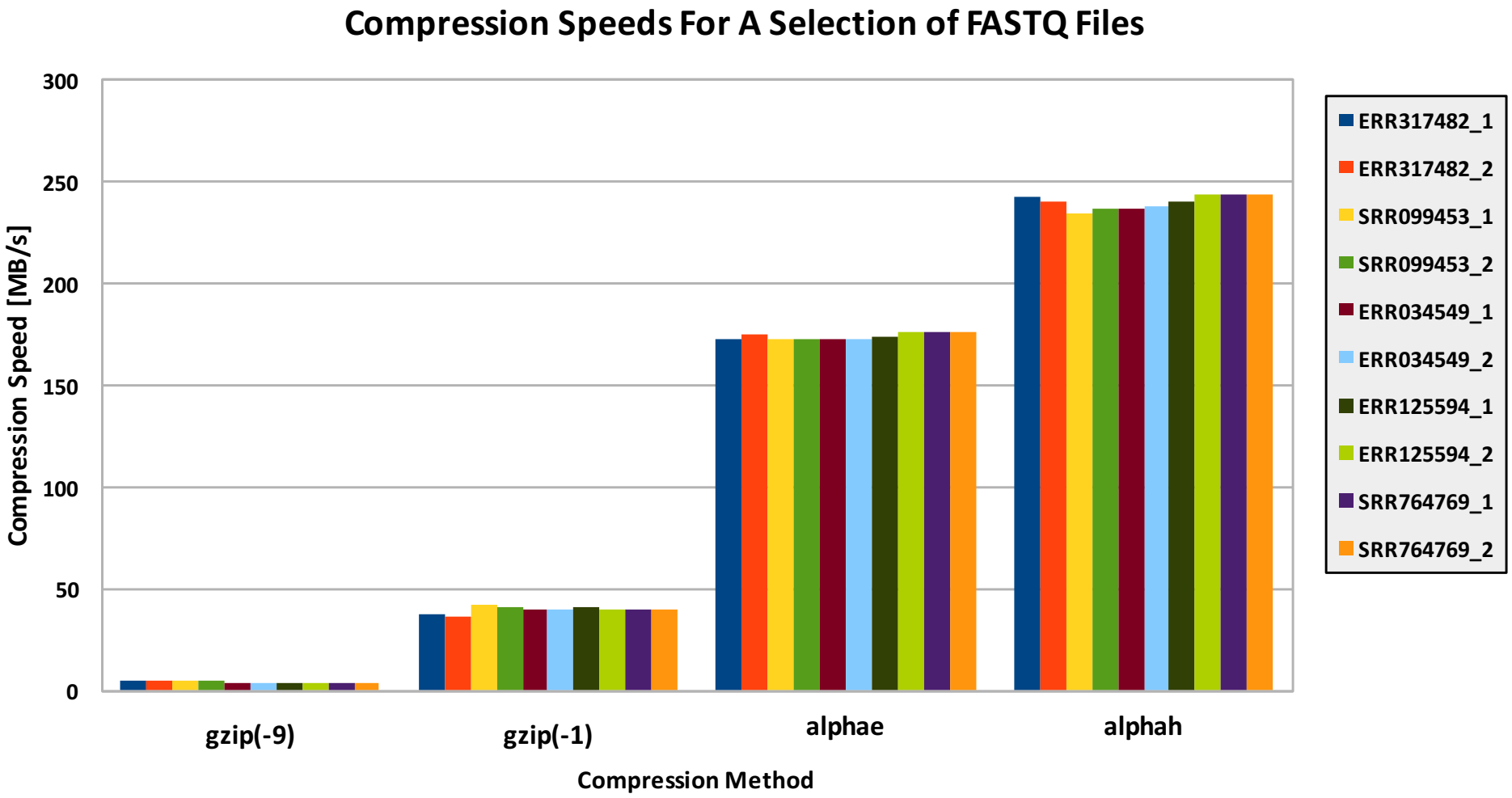
Modeling Improvement:

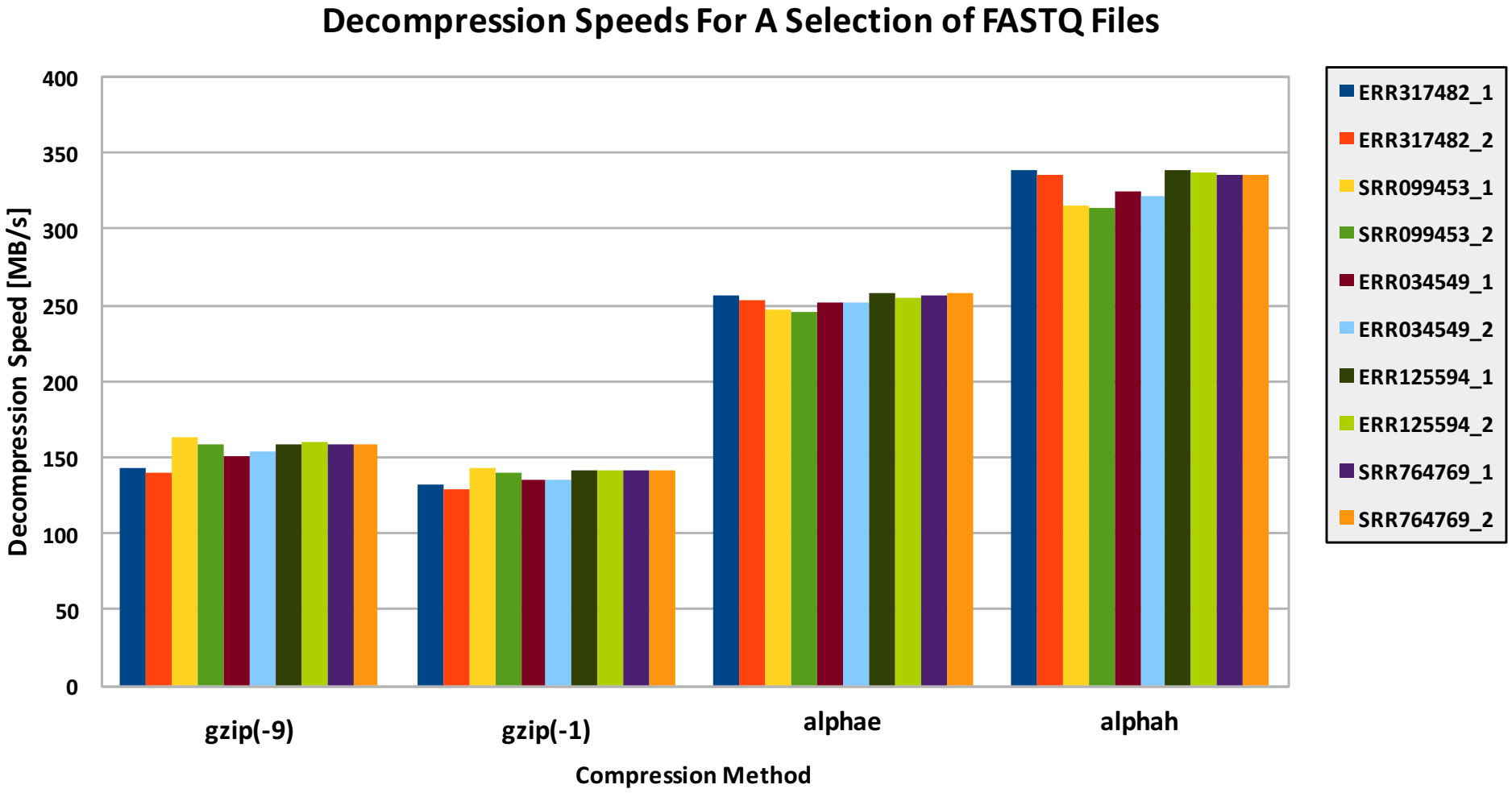
- Arrange data before compression to have the components of the same type, across reads, grouped; so, all headers grouped together, all sequences grouped, and so on.



- Compress each component or group of components (eg. sequences and quality values) using a specific coder (eg. Lempel-Ziv for headers and and Huffman or arithmetic encoder for sequences and quality values).


Compression Ratios For A Selection of FASTQ Files
Compression Ratio [%]
50
45
40
35
30
25
20
15
10
5
0
gzip(-9)
gzip(-1)
alphae
alphah
Compression Method
ERR317482_1
ERR317482_2
SRR099453_1
SRR099453_2
ERR034549_1
ERR034549_2
ERR125594_1
ERR125594_2
SRR764769_1
SRR764769_2

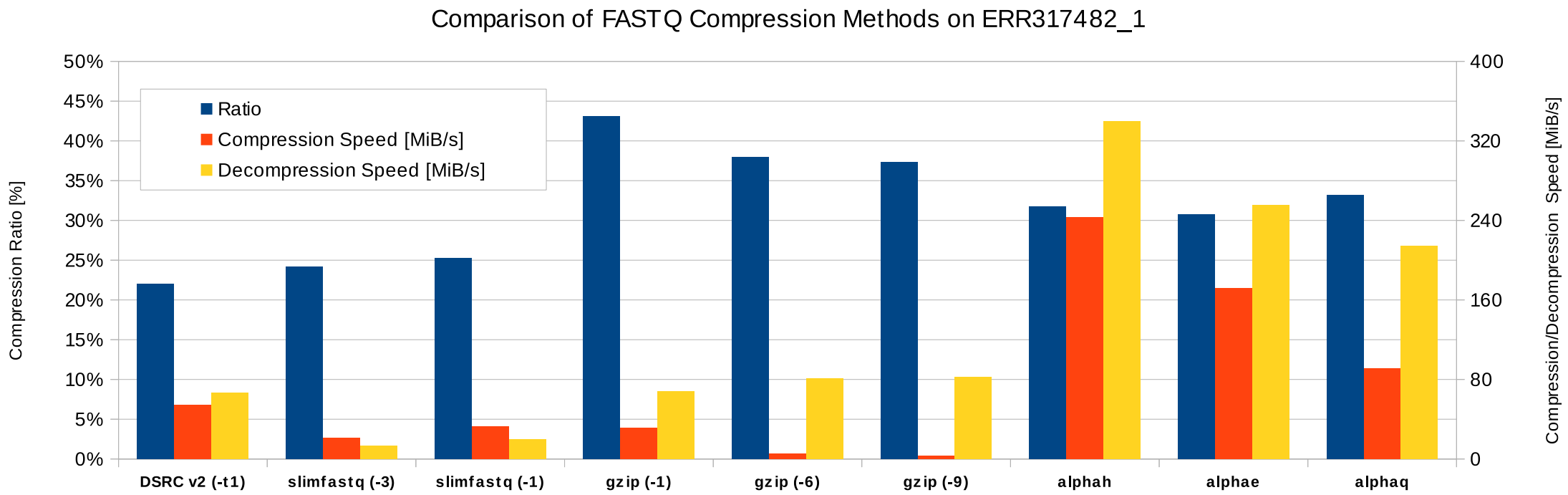

Compression Speeds For A Selection of FASTQ Files
Compression Speed [MB/s]
300
250
200
150
100
50
0
gzip(-9)
gzip(-1)
alphae
alphah
Compression Method
ERR317482_1
ERR317482_2
SRR099453_1
SRR099453_2
ERR034549_1
ERR034549_2
ERR125594_1
ERR125594_2
SRR764769_1
SRR764769_2


Decompression Speeds For A Selection of FASTQ Files
Decompression Speed [MB/s]
400
350
300
250
200
150
100
50
0
gzip(-9)
gzip(-1)
alphae
alphah
Compression Method
ERR317482_1
ERR317482_2
SRR099453_1
SRR099453_2
ERR034549_1
ERR034549_2
ERR125594_1
ERR125594_2
SRR764769_1
SRR764769_2

Outline of the Talk:

- Introduction / Motivation
- Data used in experiments
- General purpose compressors comparison
- Simple Improvements
- Special purpose compression
- Transparent compression – working on compressed data
- Parallelism / Multithreading
- Conclusion

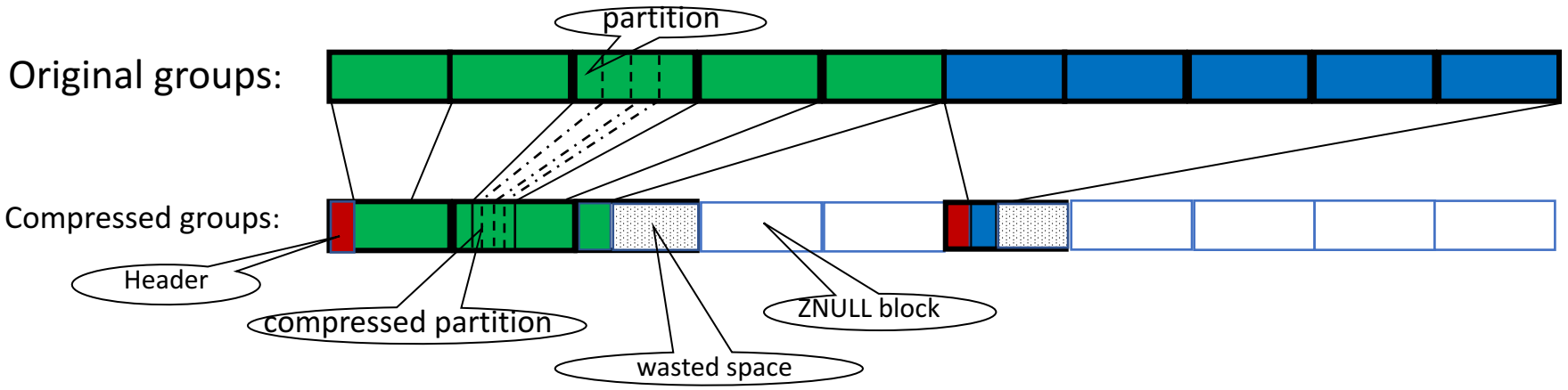
Comparison with specialized FASTQ compressors:



Outline of the Talk:

- Introduction / Motivation
- Data used in experiments
- General purpose compressors comparison
- Simple Improvements
- Special purpose compression
- Transparent compression – working on compressed data (prototype)
- Parallelism / Multithreading
- Conclusion

Compressed Data Layout in Spectrum Scale (GPFS):



Compression Group:

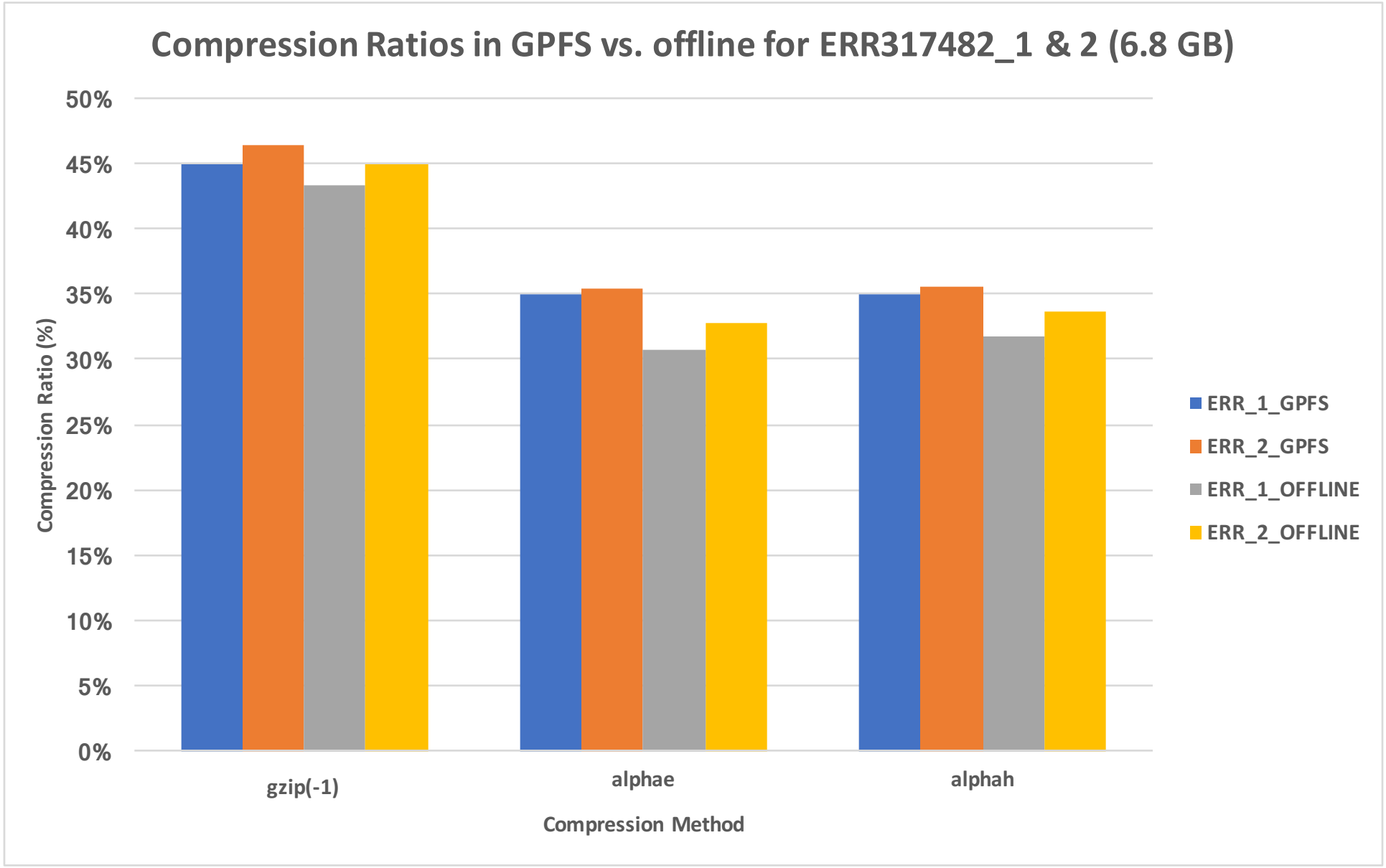
- File divided into fixed size "*compression groups*" (example 5 blocks per group).
- Each group compressed into smaller set of of blocks (--> sparse).
- Header in each group maps original to compressed data location.

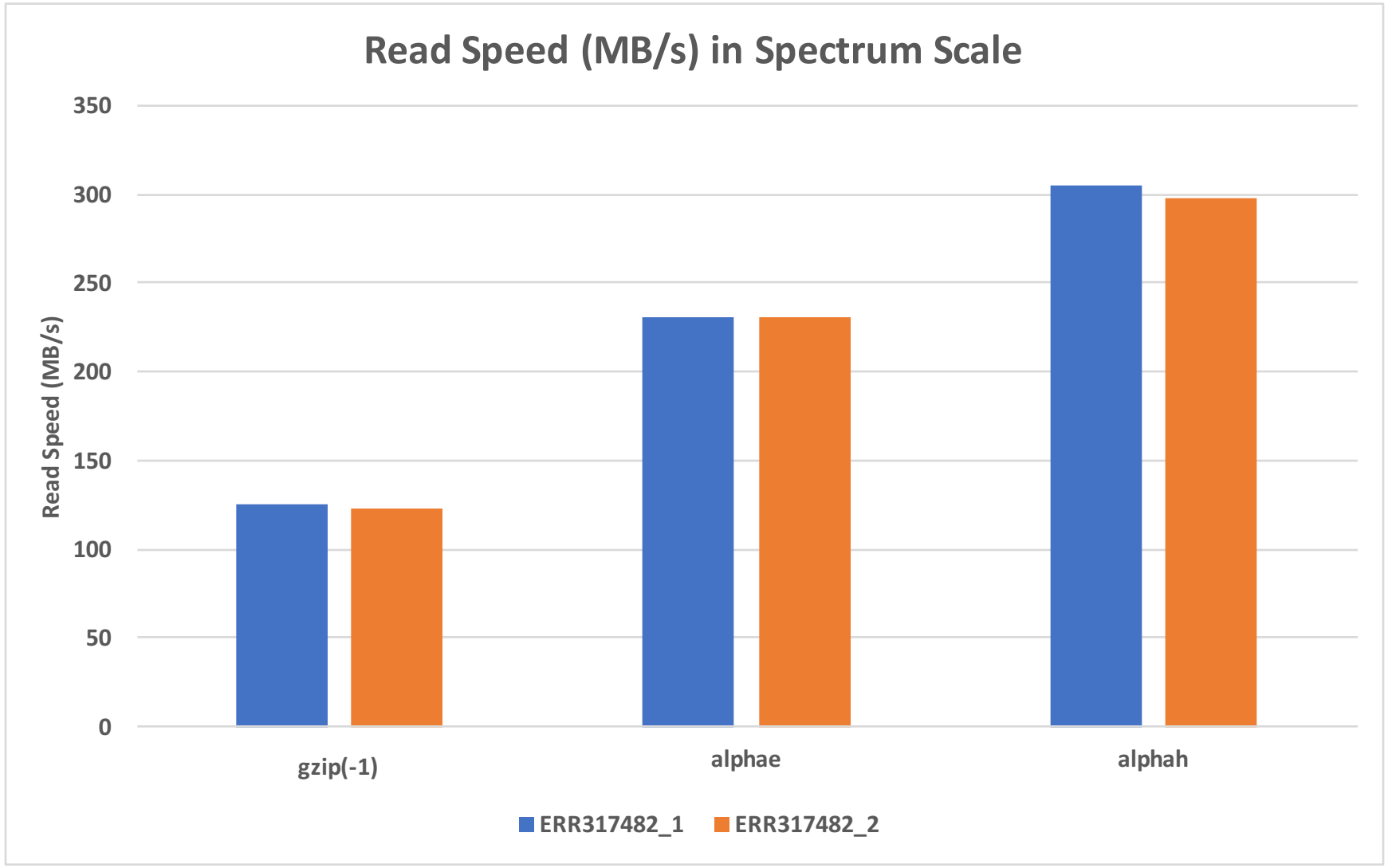
Partitions – units of compression for fast random access.

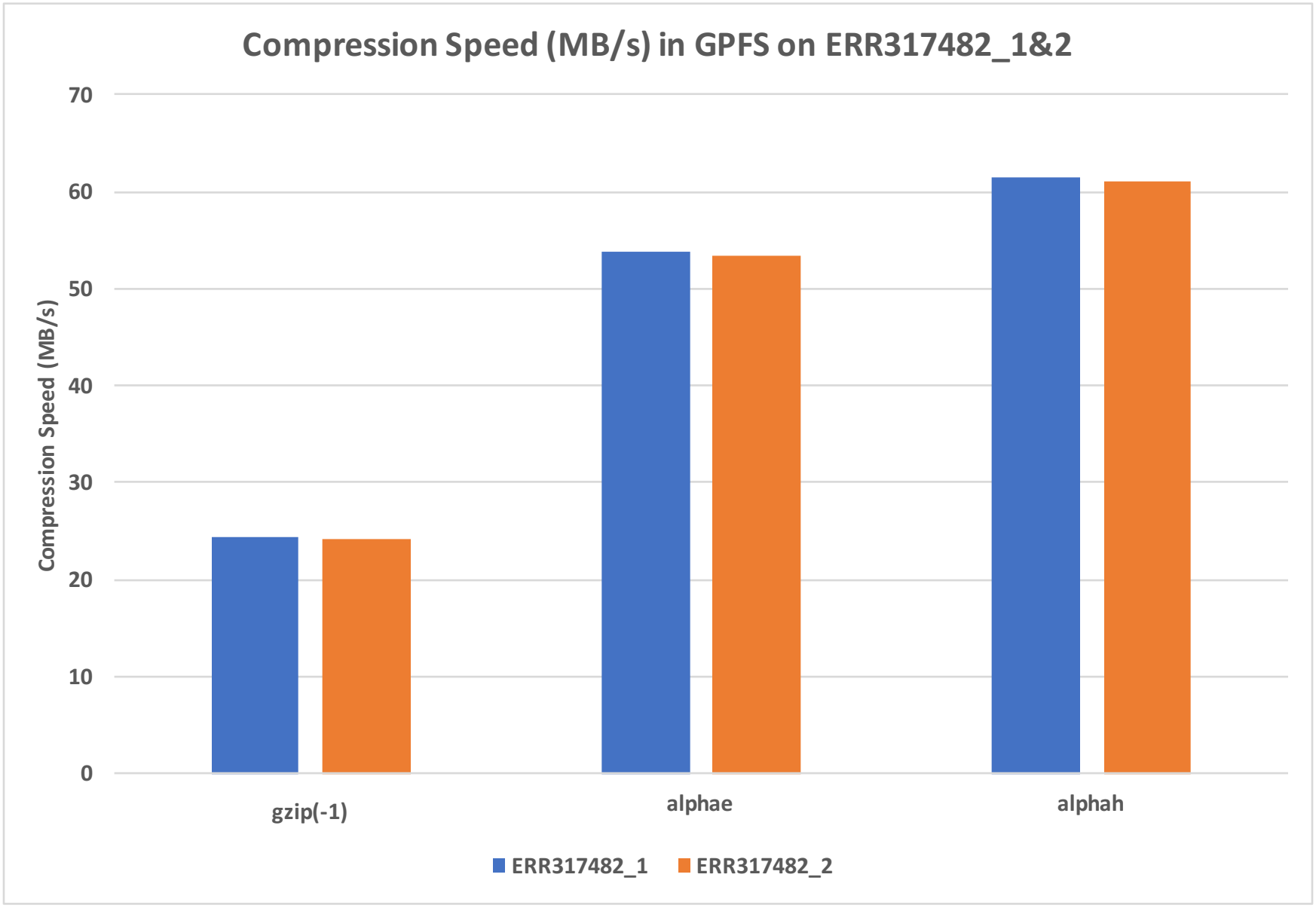
- Original data blocks are partitioned into chunks, named *partitions* (32-64KB each).
- Partitions are independently compressed to ensure fast random access.

- **Genomic compression is not available in IBM Spectrum Scale product**

- The benchmarks were obtained on a prototype.


Compression Ratios in GPFS vs. offline for ERR317482_1 & 2 (6.8 GB)
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
Compression Ratio (%)
gzip(-1)
alphae
alphah
Compression Method
ERR_1_GPFS
ERR_2_GPFS
ERR_1_OFFLINE
ERR_2_OFFLINE


Read Speed (MB/s) in Spectrum Scale
Read Speed (MB/s)
350
300
250
200
150
100
50
0
gzip(-1)
alphae
alphah
ERR317482_1
ERR317482_2


Compression Speed (MB/s) in GPFS on ERR317482_1&2
Compression Speed (MB/s)
70
60
50
40
30
20
10
0
gzip(-1)
alphae
alphah
ERR317482_1
ERR317482_2

Advantages of transparent compression of genomic data:

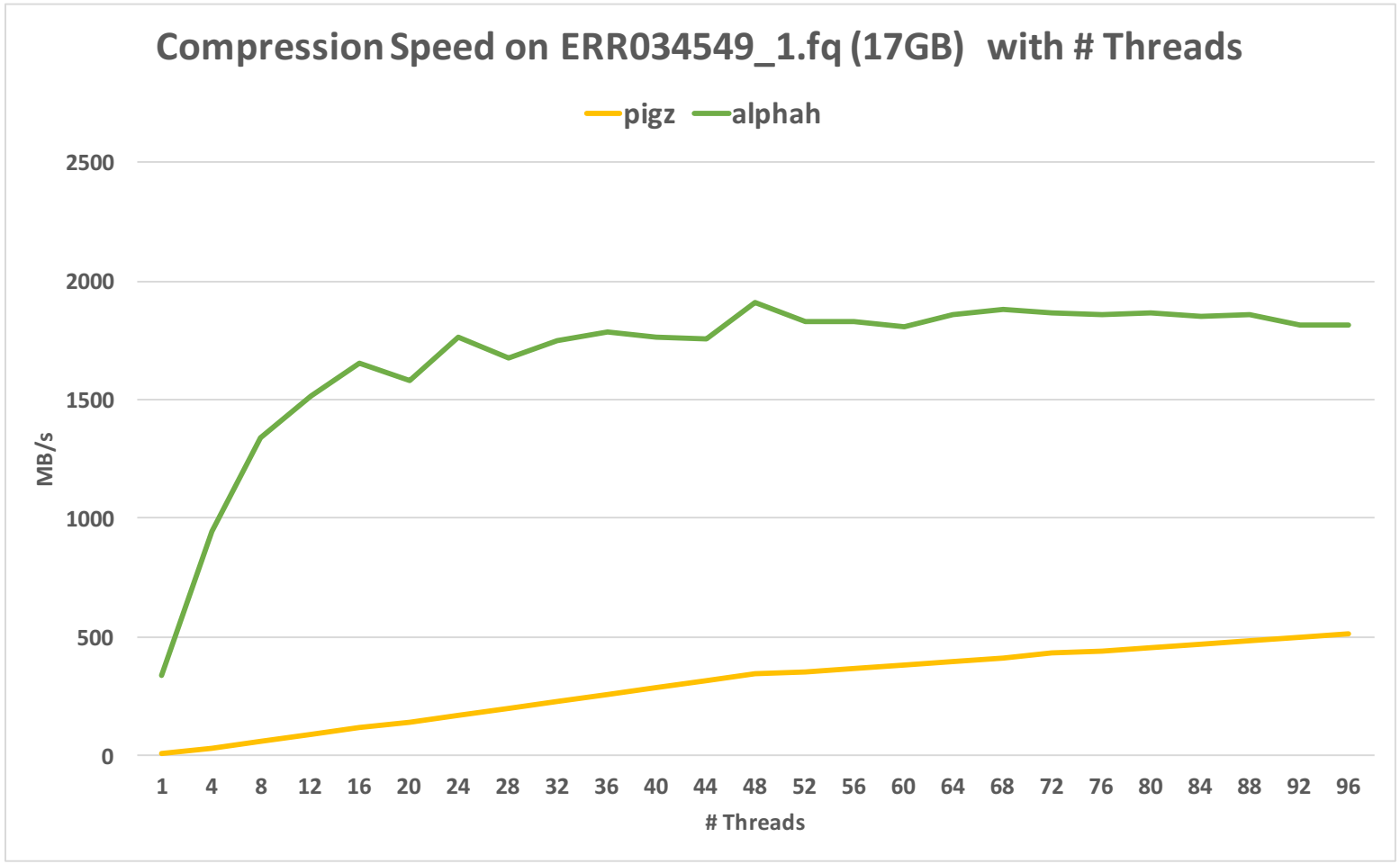
- Simplify user's interaction allowing them to concentrate on biological research rather than dealing with manual compression and decompression of genomic data.
- Eases the challenge of capacity management of genomic data files for IT infrastructure teams in the genomic sequencing domain.
- Researchers and/or genomic analysis pipelines can work on the files as if they would be uncompressed files and do not need to care if the file is actually compressed or not.
- The infrastructure team can compress the file independently and transparently in the background at any time to save storage space without interfering with the work of the user / researcher or genomic pipeline.

Outline of the Talk:

- Introduction / Motivation
- Data used in experiments
- General purpose compressors comparison
- Simple Improvements
- Special purpose compression
- Transparent compression – working on compressed data
- Parallelism / Multithreading
- Conclusion

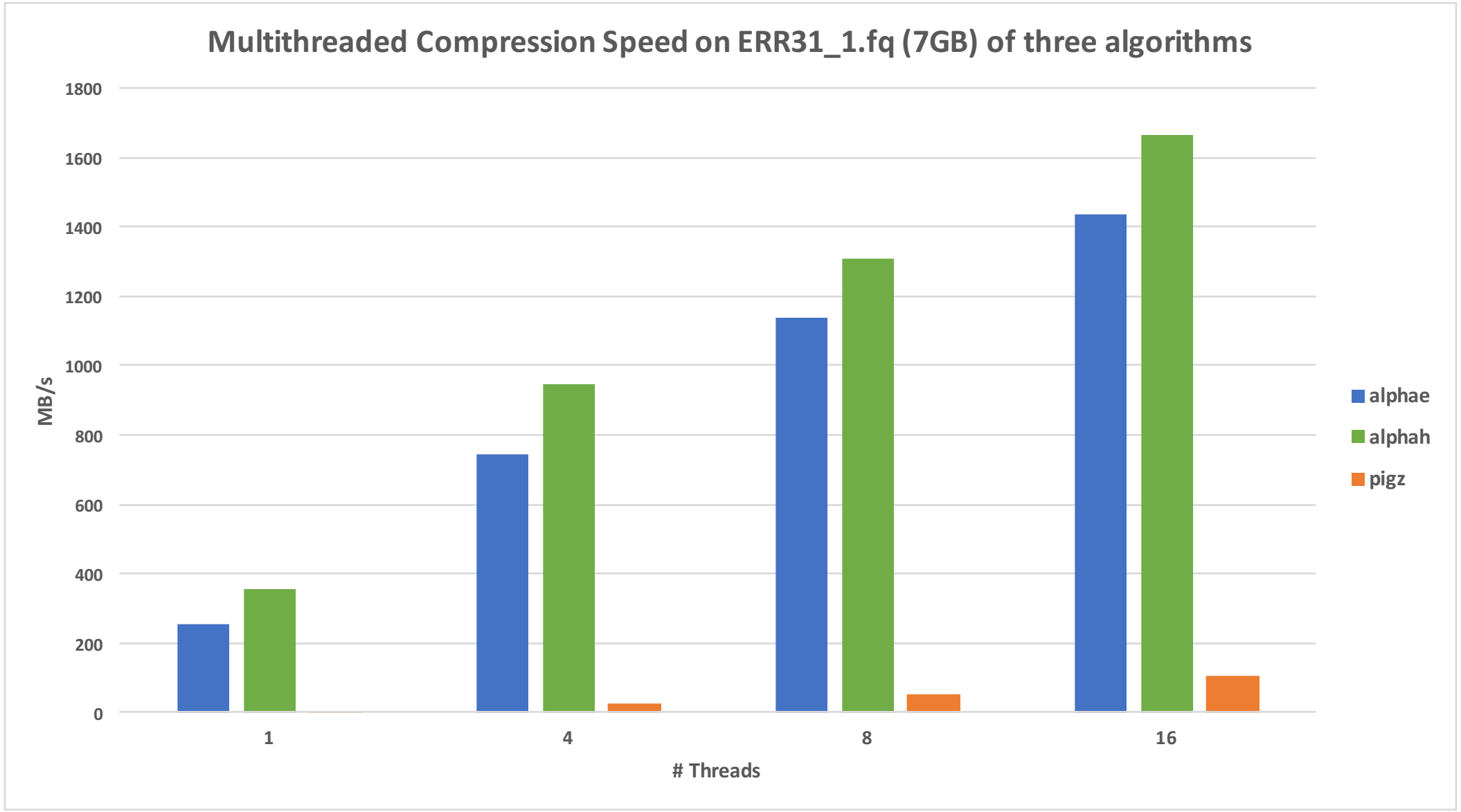
Multithreaded Compression with Alpha{e,h} vs. Pigz:

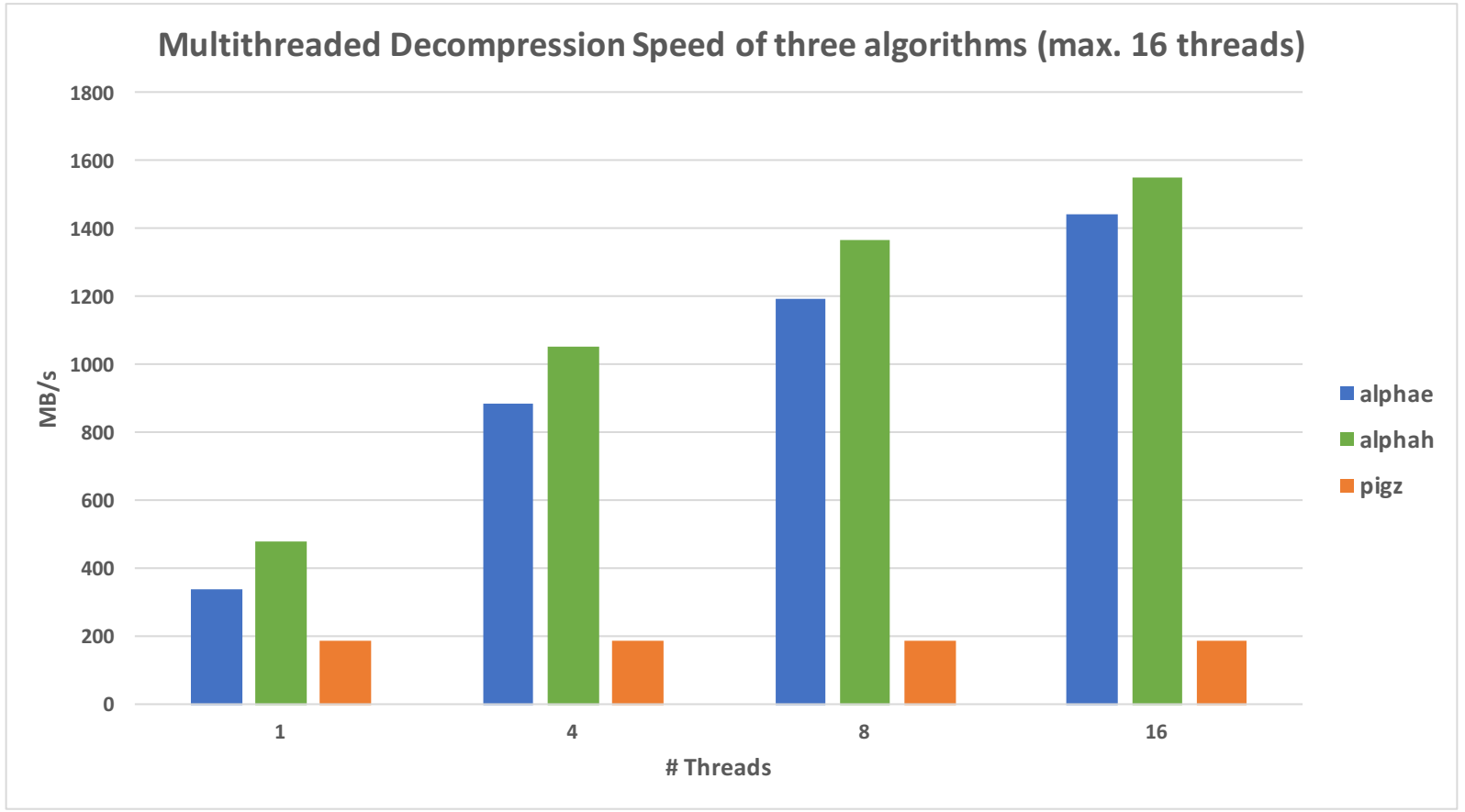
- Parallel implementation of the Alpha{e,h} is done using OpenMP package in C/C++.
- Pigz is open source implemented by the gzip developers using Pthreads in C.



- Using one thread Alphah is about as fast as pigz using 96 threads!

  (Intel Xeon Platinum 8160, 2.1GHz, 48 cores / 96 CPUs)

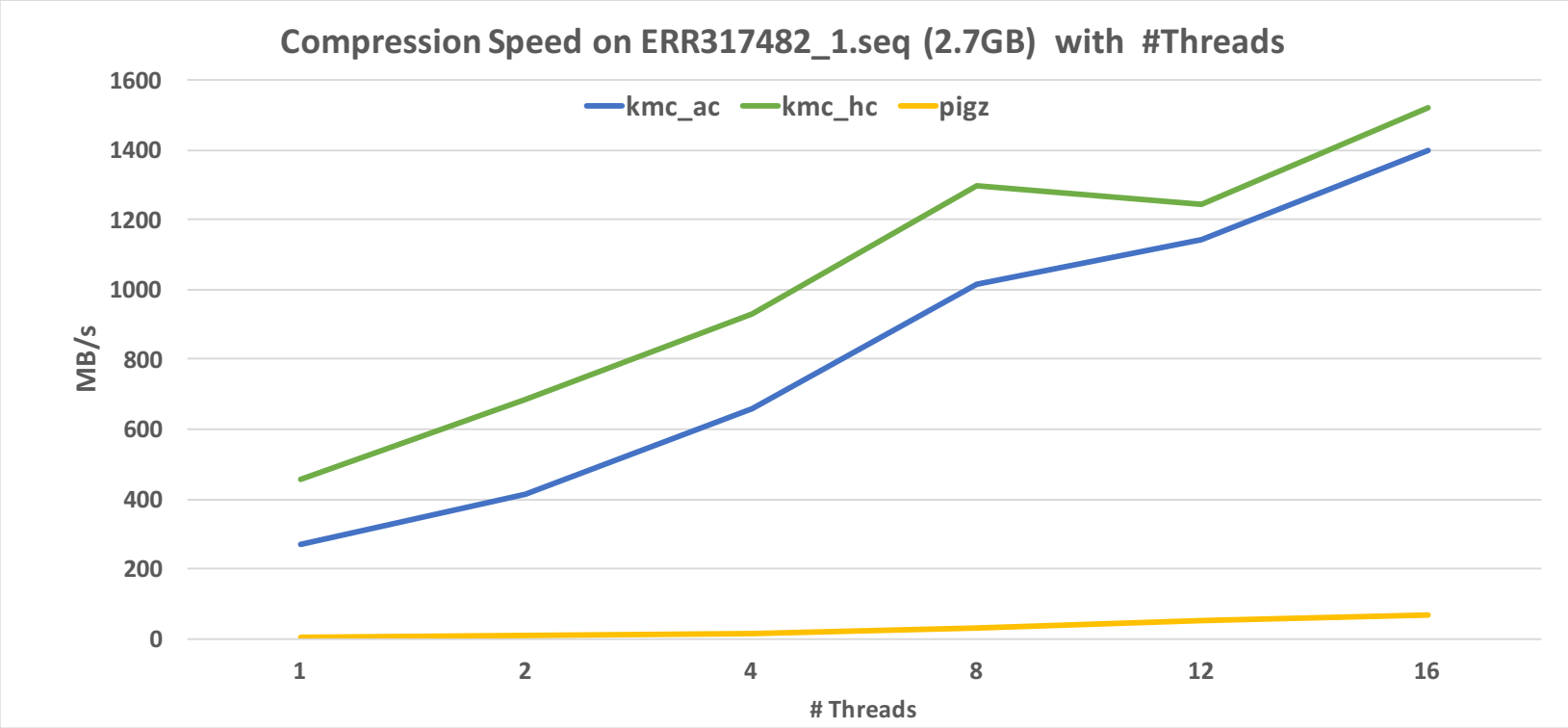

Multithreaded Compression Speed on ERR31_1.fq (7GB) of three algorithms
1800
1600
1400
1200
1000
800
600
400
200
0
MB/s
1
4
8
16
Threads
alphae
alphah
pigz



- Compression ratios are as for sequential execution:

FASTQ Data File	Original (MB)	Pigz (MB)	Alphah (MB)	Alphae (MB)	% better (alphae vs pigz)
ERR317482_1.fq	6,949	2,641	2,204	2,138	24%
ERR034549_1.fq	16,603	5,766	5,145	4,993	15%

Multithreaded FASTA Compression vs. Pigz:



FASTA Data	Original (MB)	Pigz (MB)	Kmc_ac (MB)	Kmc_hc (MB)	% better (kmc_ac vs pigz)
ERR317482_1.seq	2,719	802	692	751	16%
ERR317482_2.seq	2,719	807	692	751	17%

(*)*kmc* stands for *k-mer* compression.

Summary:

- Better and faster compression methods for genomic data (than *gzip*) exist.

- It would be good that FASTQ data generated by sequencing machines to be compressed with these methods (eg. *alphae/h*).

- Adding transparent genomic compression to a file system can alleviate many problems for the users & IT personal, with no need for a standardization.