Energy-Efficiency and Performance Trade-offs of Data-reduction techniques

Tim Dobert

Arbeitsbereich Wissenschaftliches Rechnen
Fachbereich Informatik
Fakultät für Mathematik, Informatik und Naturwissenschaften
Universität Hamburg

17.12.2014

Overview

Motivation

Data reduction should be considered for three main reasons:

- Storage/Hard drives constantly use power

Component	Peak power	Count	Total	Percentage
CPU	40 W	2	80 W	37.6 %
Memory	9 W	4	36 W	16.9 %
Disk	12 W	1	12 W	5.6 %
PCI slots	25 W	2	50 W	23.5 %
Motherboard	25 W	1	25 W	11.7 %
Fan	10 W	1	10 W	4.7 %
System total			213 W	

Table I. Component peak power breakdown for a typical server . [1]

- Transmitting data costs energy
- Growing speed gap between CPUs and memory

General Approach

Using additional computing power to reduce data.

- This creates a time/energy overhead
- Reduces needed hard drives, could replace with SSDs
- Reduce transmission energy (important on mobile devices)

This can be done using three different techniques.

Compression

General Idea

Encoding output data to reduce redundancy and therefore size

- This involves encoding output, decoding input
- Best case: Integrated into file I/O (i.e. ZFS)
- Compression must be lossless

Trade-off

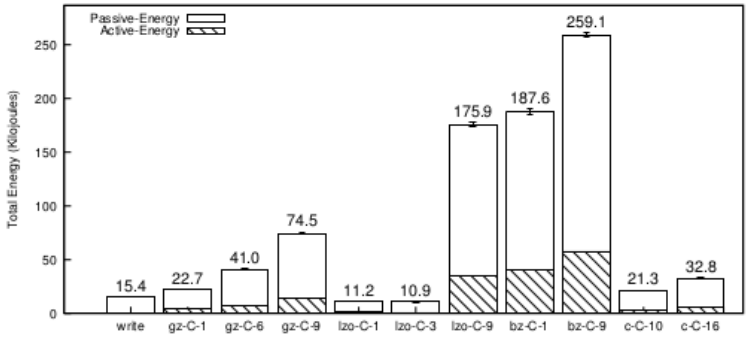
Whether it's worth it depends on:

- Computational overhead vs. space savings
- Compression algorithm used
- Computing power

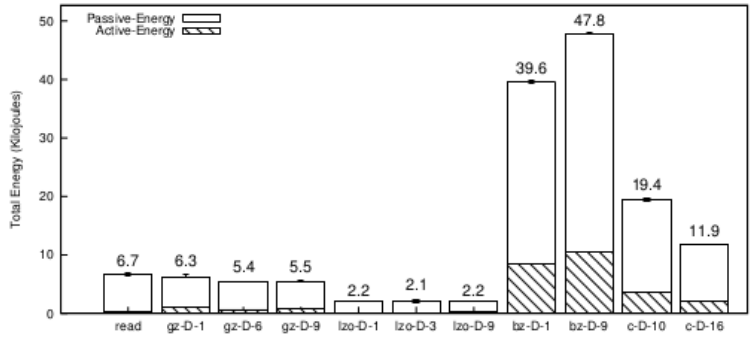
In Practice

Energy Consumption for:

Text file



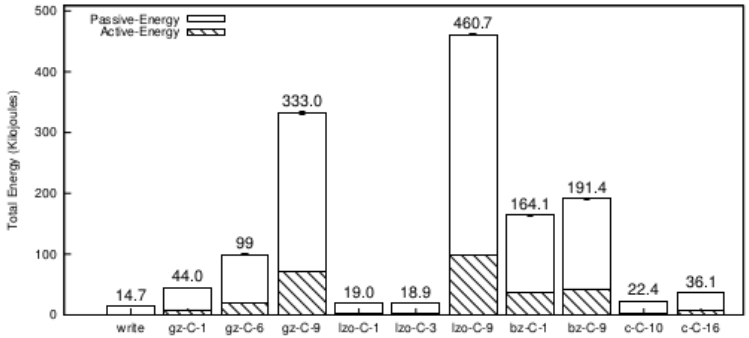
(c) **Energy** consumed for write vs. compression



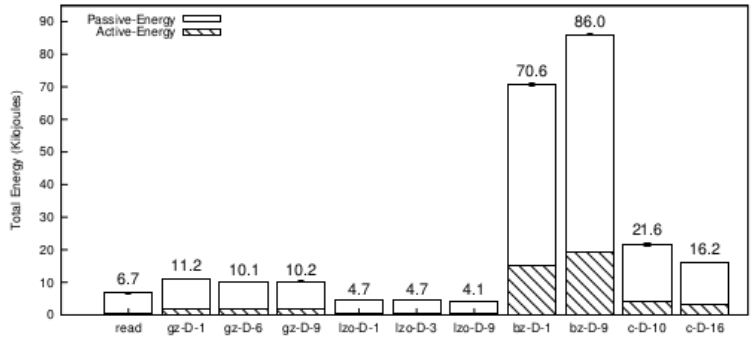
(d) **Energy** consumed for read vs. decompression

[3]

Binary file



(a) **Energy** consumed for write vs. compression



(b) **Energy** consumed for read vs. decompression

[3]

Additional Thoughts: Communication

There are different approaches for optimizing energy consumption for a whole system or just parts of it.

- Lots of data transmission $\Rightarrow$ overall savings might outweigh increase on nodes
- Asymmetric compression for single devices

Just choosing the fastest algorithm or the one with the best compression ratio is rarely ideal.

Data Deduplication

Concept

The Goal is to reduce redundancy across all files.

This is how it's done:

- Data is divided into blocks
- Calculate fingerprints, store in hash table
- Every block is only stored once

This approach is most effective for data centers, HPC and backups.

Parameters and Trade-offs

A deduplication system can have the following parameters:

- Size of the blocks:
  - static
  - variable
- Block size affects the size of the Table
  - Big, static $\Rightarrow$ small Table
  - Small, variable $\Rightarrow$ Big Table

  Normally a block size of 16kB is used.
- Smaller tables are faster to work with

Like with compression, there is a reduction-overhead trade off.

In Practice

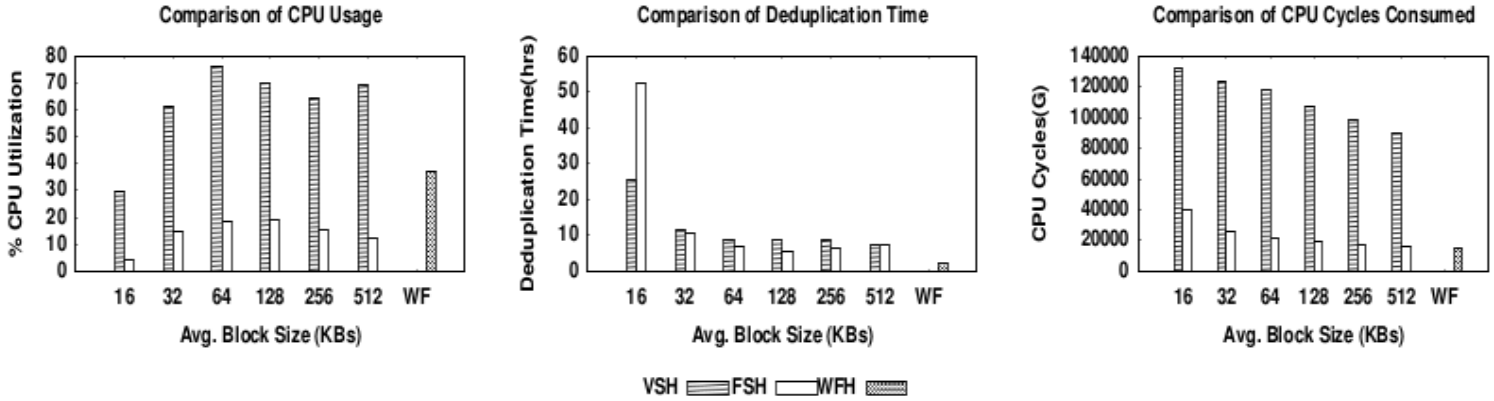
Evaluation:

TABLE II
DEDUPLICATION RATIO USING CONTENT-DEFINED CHUNKING WITH AN AVERAGE CHUNK SIZE OF 8 KB ON DIFFERENT HPC DATA SETS.

Data set	Ratio	Data set	Ratio
BSC-BD	7.0%	DKRZ-B5	29.5%
BSC-MOD	21.3%	DKRZ-B6	22.5%
BSC-PRO	29.3%	DKRZ-B7	14.1%
BSC-SCRA	38.3%	DKRZ-B8	13.9%
DKRZ-A	17.9%	DKRZ-K	49.3%
DKRZ-B1	19.7%	DKRZ-M1	15.0%
DKRZ-B2	27.6%	DKRZ-M2	21.1%
DKRZ-B3	74.4%	RENCI	23.8%
DKRZ-B4	27.1%	RWTH	23.2%

[5]

Efficiency depending on Block Size



(a) CPU Utilization (b) Deduplication Time (c) CPU Cycles

[6]

Recomputation

General Idea

The result of a computation is not saved. It will be recomputed when needed.

- Only store input, maybe partial results
- Perform computation every time the data is needed
- Advances in hard- and software may make future computations more efficient

Example: Online video transcoding

Trade-offs

Of the three techniques, this is the most situational.

- Computing is obviously costly, slow
- Only applicable to rarely used data
- Computation must be deterministic
- Code preservation, emulation needed

It's difficult to say in advance, if the trade-off will be worth it.

Conclusion

Conclusion

All of the presented techniques come at a cost. In the end, you should keep in mind:

- More reduction is more computationally expensive
- The effectiveness depends on the application and parameters
- Deduplication is not very effective when compression is already being used
- If nothing else, try a light compression method

Experiments

Processing a 4k bitmap image with and without compression.
Compression algorithm: lz4 (https://code.google.com/p/lz4/)
Uncompressed file size: 24.9MB

	normal	compression	compressed size
Picture	read: 0.010s write: 0.027s total: 0.100s	read: 0.030s write: 0.123s total: 0.220s	22.0 MB
Black	- - -	read: 0.017s write: 0.004s total: 0.076s	97.6 kB
Random	- - -	read: 0.020s write: 0.030s total: 0.110s	25.0 MB

One more thought:
Compressing on the GPU. It's generally less efficient, though[9].

Sources I


[1] A Survey on Techniques for Improving the Energy Efficiency of Large Scale Distributed Systems
https://www.fsl.cs.sunysb.edu/docs/greencomp/green-compress.pdf


[2] Benefits of Data Compression in HPC Storage
http://wr.informatik.uni-hamburg.de/_media/research/publications/2014/epbodcihss14-evaluating_power_performace_benefits_of_data_compression_in_hpc_storage_servers.pd


[3] Energy and Performance Evaluation of Lossless File Data Compression on Server Systems
https://www.fsl.cs.sunysb.edu/docs/greencomp/green-compress.pdf


[4] Energy-aware lossless data compression
http://dl.acm.org/citation.cfm?id=1151692


[5] Deduplication in HPC Storage
http://wr.informatik.uni-hamburg.de/_media/research/publications/2012/asoddihssm12-a_study_on_data_deduplication_in_hpc_storage_systems.pdf


[6] Demystifying Data Deduplication
http://dl.acm.org/citation.cfm?id=1462739

Sources II


[7] Maximizing Efficiency By Trading Storage for Computation
https://www.usenix.org/legacy/event/hotcloud09/tech/full_papers/adams/adams_html/ (09.12.14)


[8] Exascale Storage Systems
http://superfri.org/superfri/article/download/20/6


[9] Parallel Lossless Data Compression on the GPU
http://www.idav.ucdavis.edu/func/return_pdf?pub_id=1087

Zip graphic:

http://wikimediafoundation.org/wiki/File:Simple_Comic_zip.png

Summary

- Less data means less energy consumption
- Three main techniques for data reduction
  - Compression
  - Deduplication
  - Recomputation
- Compression works on single files and can be integrated into the file system
- Deduplication works across all files, can be costly though
- Recomputation is highly situational

	Processor	Memory	Network	Storage
Re-computation of results	-	-	-	+
Deduplication	-	--	0	+
Compression (client side)	-	0	+	++
Compression (server side)	-	0	0	++
User education	+	+	+	+

Table 7. Benefits and penalties of different concepts for data reduction (+ benefits; - penalties)

[8]