Data Reduction Techniques

Kira Duwe

Seminar - WR - UHH

09.11.2015

Structure

1. Introduction
2. Scientists and their methods
3. Computer scientists and their methods
   - Compression
   - Comparison of algorithms
     - Fast compression algorithms
     - Algorithms with good ratio
   - Deduplication
   - Recomputing
4. Conclusion

Motivation

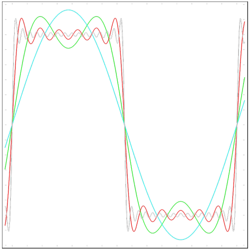
- gap between computational speed and storage speed is widening
- in HPC storage systems in two-digit PB and throughput around TB/s
- growing difficulties handling I/O reducible by compression
- little or no support by file system, especially parallel
- usually needs constant power supply
- transmitting data costs time and energy (e.g.Keppler satellite)
- data intense research (e.g. square kilometre array (SKA) some PB/s)

Different Approaches To Data Reduction

- Scientists vs computer scientists
- Which data is really necessary?
- Which precision is needed?
- How can I compress the data as small and fast as possible?

Data Reduction

- rounding
- Fourier series (periodic phenomena)/ Fourier transform (aperiodic p.): transforming data and only analysing until certain precision



$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty}(a_k \cdot cos(kt) + b_k \cdot sin(kt))$$

- statistical methods like linear regression and smoothing

Compression

- compression speed = uncompressed size/Δ time; in $[\frac{MB}{s}]$
- compression ratio = uncompressed size/compressed size, e.g. $\frac{12MB}{3MB} = 4$
- space savings = 1 - (compressed size/uncompressed size), e.g. $1 - \frac{3}{12} = 0.75$
- lossy vs lossless compression
- system level (e.g. part of file system) vs user level

Detecting incompressible data

- Komolgorov complexity
  - shortest description of statements that will produce the character sequence looked at
  - data: ABABABABABABABABABAB
  - shortest description: AB*10
  - measurement for structuredness of data and compressibility
- prefix estimation
- entropy: $H = \sum_{i=1}^{m} p_i log_2 p_i$ (Huffman-Encoding)
- heuristics, e.g. IBM solution

Detecting incompressible data

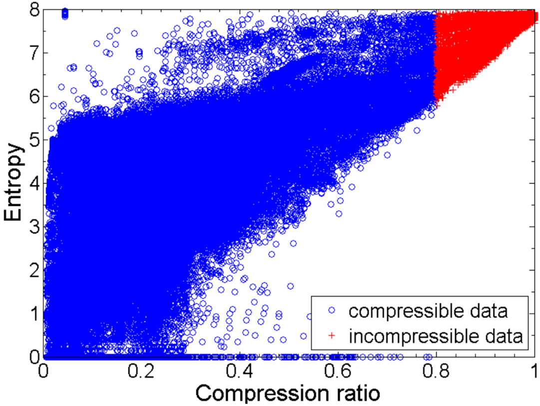


Abbildung: Entropy - compressibility relation [HKM+13, p. 8]

(here compression ratio: compressed size/uncompressed size)
(0.8≙1.25)

Detecting incompressible data

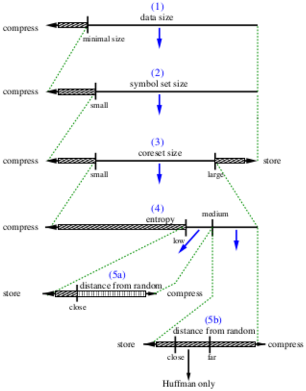


Abbildung: IBM heuristics to detect incompressible data

[HKM+13, p. 9]

Comparison of different compression algorithms

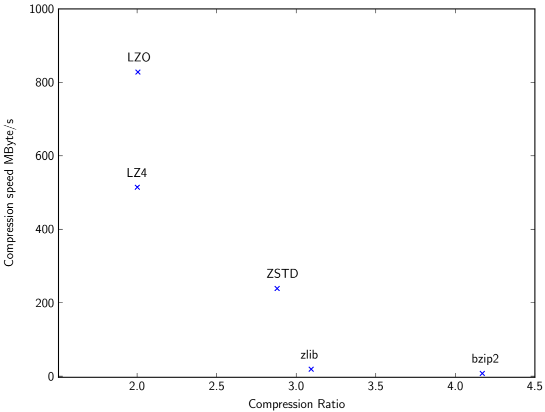


Abbildung: Results PRE-project by Hans Ole Hatzel

Fast compression algorithms

- LZ-family (LZ77,LZ78, LZSS, LZMA, LZRW, LZO, LZJB, LZ4)
- dictionary or search buffer and sliding window

Buffer						sliding w.				(x,y,z)
1	2	3	4	5	6	1	2	3	4	(-,-,-)
						B	A	N	A	(0,0,B)
					B	A	N	A	N	(0,0,A)
				B	A	N	A	N	E	(0,0,N)
			B	A	N	A	N	E		(5,2,E)
B	A	N	A	N	E					(-,-,-)

- LZO = Lempel-Ziv-Oberhumer (1996)
- LZ4: author: Yann Collet(2011), default in OpenZFS
- LZ4 Fast: different way of updating hash table

Why LZO and LZ4 are so much faster

- "Be warned:
  the main source code in the 'src' directory is a real pain to understand as I've experimented with hundreds of slightly different versions.
  It contains many #if and some gotos, and is *completely optimized for speed* and not for readability." - Markus Franz Xaver Oberhumer
- do not search all possible matches for best solution, missing some short matches
- no entropy encoding, no analysis of symbol frequency
- byte output not bit output
- optimisation for modern CPUs
- possibly using heuristics to detect incompressible data

Deflate format

- RFC 1950(ZLIB),1951(DEFLATE),1952(GZIP)
- deflate = LZSS + Huffman Encoding; not patented itself
- ZIP: Phil Katz; patented implementations (.zip)
- GZIP: Jean-Loup Gailly, Mark Adler; (.gz /.tgz)
- ZLIB: Jean-Loup Gailly, Mark Adler; originally intended for libpng

GZIP-Level

- RFC 1950: "FLEVEL (Compression level) These flags are available for use by specific compression methods. The 'deflate' method (CM = 8) sets these flags as follows:
  0 - compressor used fastest algorithm
  1 - compressor used fast algorithm
  2 - compressor used default algorithm
  3 - compressor used maximum compression, slowest algorithm"
- ZLIB manual:" Compression levels.
  #define Z_NO_COMPRESSION 0
  #define Z_BEST_SPEED 1
  #define Z_BEST_COMPRESSION 9
  #define Z_DEFAULT_COMPRESSION (-1) "
- -1 is equivalent to level 6 best compromise

Zopfli and Gipfeli and Brotli

author: Jyrki Alakuijala

“Gipfeli is a Swiss name for croissant. We chose this name, because croissants can be compressed well and quickly”

- implements deflate format
- slower than gzip (around x80-100)
- compression density is about 5% better

- based on LZ77 with fast entropy coding
- compression ratio is similar to Zlib in the fastest mode
- more than three times faster than Zlib

- new format
- increase of around 20% in compression density in contrast to deflate, speed stays the same
- pre-defined static dictionary of around 13.000 strings

Bzip2 and pbzip2

- author: Julian Seward
- Run-length encoding
- Borrows-Wheeler transform

rotation	sort	result
ANANAS$	$ANANAS	S
$ANANAS	ANANAS$	$
S$ANANA	ANAS$AN	N
AS$ANAN	AS$ANAN	N
NAS$ANA	NANAS$A	A
ANAS$AN	NAS$ANA	A
NANAS$A	S$ANANA	A

- Move to front coding
- Huffman encoding
- slow compression speed but one of the best compression ratios
- parallel version with nearly linear speed up

Blosc

- blocking, shuffling and compression library project
- improving speed between memory and CPU
- goal: be faster than memcpy()
- BloscLZ based on FASTLZ
- support for LZ4, LZ4HC, Zlib, Snappy

Deduplication

- data split in blocks
- block stored only once
- table needed to find right block
- a lot of memory needed to hold tables

Recomputing

- only suitable for data which is accessed rarely
- results are not saved
- recomputed when needed
- virtualisation provides necessary environment to recompute data on different system

Conclusion

- data reduction necessary to slow down gap widening between computational speed and storage speed
- mathematical operations to keep more important data and discard less valuable
- on system level fast algorithms to compress all the time
- user level: algorithms with very good compression ratio
- deduplication: difficult to handle
- recomputation: not ready yet

Literatur I

Matthew Ahrens.
OpenZFS: a Community of Open Source ZFS Developers.".
AsiaBSDCon 2014, page 27, 2014.

Jyrki Alakuijala, Evgenii Kliuchnikov, Zoltan Szabadka, and Lode Vandevenne.
Comparison of Brotli, Deflate, Zopfli, LZMA, LZHAM and Bzip2 Compression Algorithms.
Technical report, Google, Inc., September 2015.

Jyrki Alakuijala.
Brotli Compressed Data Format, 2015.

Francesc Alted.
Why modern CPUs are starving and what can be done about it.
Computing in Science & Engineering, 12(2):68–71, 2010.

Literatur II

Francesc Alted.
Blosc, 2015.

Jyrki Alakuijala and Lode Vandevenne.
Data compression using Zopfli.
Tech. rep. Google Inc., Feb. 2013.: https://zopfli. googlecode. com/files/Data _ compression_using_Zopfli. pdf.

Kul Bhushan.
Zopfli: Google's new data compression algorithm, March 2013.

Guy E Blelloch.
Introduction to data compression.
Computer Science Department, Carnegie Mellon University, 2001.

Literatur III

Max Bruning.
ZFS On-Disk Data Walk (Or: Where's My Data).
In *OpenSolaris Developer Conference*, 2008.

MA BASIR and MH YOUSAF.
TRANSPARENT COMPRESSION SCHEME FOR LINUX FILE SYSTEM.
Nucleus, 49(2):129–137, 2012.

L Peter Deutsch.
DEFLATE compressed data format specification version 1.3.
1996.

Jean-loup Gailly and Mark Adler.
gzip, 2015.

Literatur IV

Jeff Gilchrist.
Parallel data compression with bzip2.
In *Proceedings of the 16th IASTED International Conference on Parallel and Distributed Computing and Systems*, volume 16, pages 559–564, 2004.

Danny Harnik, Ronen I Kat, Oded Margalit, Dmitry Sotnikov, and Avishay Traeger.
To Zip or not to Zip: effective resource usage for real-time compression.
In *FAST*, pages 229–242, 2013.

D. Harnik, E. Khaitzin, D. Sotnikov, and S. Taharlev.
A Fast Implementation of Deflate.
In *Data Compression Conference (DCC), 2014*, pages 223–232, March 2014.

Literatur V

Per Karlsen and Lasse Collin.
lzma, 2015.

R. Lenhardt and J. Alakuijala.
Gipfeli - High Speed Compression Algorithm.
In *Data Compression Conference (DCC), 2012*, pages 109–118, April 2012.

Dirk Meister, André Brinkmann, and Tim Sü\s s.
File recipe compression in data deduplication systems.
In *FAST*, pages 175–182, 2013.

Dirk Meister.
Advanced data deduplication techniques and their application.
PhD thesis, Universitätsbibliothek Mainz, 2013.

Literatur VI

Nagapramod Mandagere, Pin Zhou, Mark A Smith, and Sandeep Uttamchandani.
Demystifying Data Deduplication.
In *Proceedings of the ACM/IFIP/USENIX Middleware '08 Conference Companion*, Companion '08, pages 12–17, New York, NY, USA, 2008. ACM.

Emily Namey, Greg Guest, Lucy Thairu, and Laura Johnson.
Data reduction techniques for large qualitative data sets.
Handbook for team-based qualitative research, pages 137–161, 2008.

Prabhat and Quincey Koziol, editors.
High performance parallel I/O.
CRC Press, 2015.

Literatur VII

Amir Said.
Introduction to arithmetic coding-theory and practice.
Hewlett Packard Laboratories Report, 2004.

Divya Singh, Vimal Bibhu, Abhishek Anand, Kamalesh Maity, and Bhaskar Joshi.
STUDY OF VARIOUS DATA COMPRESSION TOOLS.
2014.

Milosh Stolikj, Pieter JL Cuijpers, and Johan J Lukkien.
Energy-aware reprogramming of sensor networks using incremental update and compression.
Procedia Computer Science, 10:179–187, 2012.

Julian Seward.
Bzip2, September 2010.

Literatur VIII

Eric R Schendel, Saurabh V Pendse, John Jenkins, David A Boyuka II, Zhenhuan Gong, Sriram Lakshminarasimhan, Qing Liu, Hemanth Kolla, Jackie Chen, Scott Klasky, and others.
ISOBAR hybrid compression-I/O interleaving for large-scale parallel I/O optimization.
In *Proceedings of the 21st international symposium on High-Performance Parallel and Distributed Computing*, pages 61–72. ACM, 2012.

Zaid Bin Tariq, Naveed Arshad, and Muhammad Nabeel.
Enhanced LZMA and BZIP2 for improved energy data compression.
In *Smart Cities and Green ICT Systems (SMARTGREENS), 2015 International Conference on*, pages 1–8, May 2015.

Literatur IX

S. J. Tingay, R. Goeke, J. D. Bowman, D. Emrich, S. M. Ord, D. A. Mitchell, M. F. Morales, T. Booler, B. Crosse, R. B. Wayth, C. J. Lonsdale, S. Tremblay, D. Pallot, T. Colegate, A. Wicenec, N. Kudryavtseva, W. Arcus, D. Barnes, G. Bernardi, F. Briggs, S. Burns, J. D. Bunton, R. J. Cappallo, B. E. Corey, A. Deshpande, L. Desouza, B. M. Gaensler, L. J. Greenhill, P. J. Hall, B. J. Hazelton, D. Herne, J. N. Hewitt, M. Johnston-Hollitt, D. L. Kaplan, J. C. Kasper, B. B. Kincaid, R. Koenig, E. Kratzenberg, M. J. Lynch, B. Mckinley, S. R. Mcwhirter, E. Morgan, D. Oberoi, J. Pathikulangara, T. Prabu, R. A. Remillard, A. E. E. Rogers, A. Roshi, J. E. Salah, R. J. Sault, N. Udaya-Shankar, F. Schlagenhaufer, K. S. Srivani, J. Stevens, R. Subrahmanyan, M. Waterson, R. L. Webster, A. R. Whitney, A. Williams, C. L. Williams, and J. S. B. Wyithe.

Literatur X

The Murchison Widefield Array: The Square Kilometre Array Precursor at Low Radio Frequencies.
PASA - Publications of the Astronomical Society of Australia, 30, 2013.

Yinjun Wu, Zhen Chen, Yuhao Wen, Junwei Cao, Wenxun Zheng, and Ge Ma.
A General Analytical Model for Spatial and Temporal Performance of Bitmap Index Compression Algorithms in Big Data.
In *Computer Communication and Networks (ICCCN), 2015 24th International Conference on*, pages 1–10, August 2015.

Literatur XI

Andreas Wicenec, Derek K Gerstmann, Christopher Harris, and Kevin Vinsen.
Integrating HPC into Radio-Astronomical data reduction.
In *General Assembly and Scientific Symposium, 2011 XXXth URSI*, pages 1–1. IEEE, 2011.

Jacob Ziv and Abraham Lempel.
A universal algorithm for sequential data compression.
IEEE Transactions on information theory, 23(3):337–343, 1977.