Introduction to Data Compression

Guillaume Tochon

guillaume.tochon@lrde.epita.fr

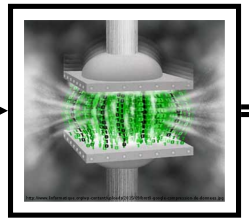
LRDE, EPITA

Data compression: whatizit?

Data compression: whatizit?

Data **compression** is the process of modifying, encoding or converting the bits structure of some input data in order to reduce the storage size of this data.

Input data,
large size



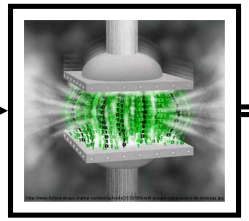
Compression algorithm

Compressed data,
smaller size

Data compression: whatizit?

Data **compression** is the process of modifying, encoding or converting the bits structure of some input data in order to reduce the storage size of this data.

Input data,
large size



Compression algorithm

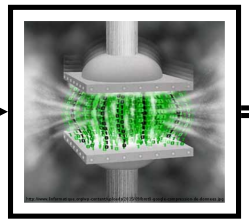
Compressed data,
smaller size

Data **decompression** is the inverse process, namely restoring the compressed data back to its original form (or a close one) for usage.

Data compression: whatizit?

Data **compression** is the process of modifying, encoding or converting the bits structure of some input data in order to reduce the storage size of this data.

Input data,
large size



Compression algorithm

Compressed data,
smaller size

Data **decompression** is the inverse process, namely restoring the compressed data back to its original form (or a close one) for usage.

Talking about data compression $\Leftrightarrow$ talking about the compression **and** the decompression algorithm.

Why does it matter?

An illustrative example

How much would weigh a non compressed HD movie?

Why does it matter?

An illustrative example

How much would weigh a non compressed HD movie?

$\rightarrow$ Each color is encoded over 1 byte ($2^8 = 256$ possible gray levels).

Why does it matter?

An illustrative example

How much would weigh a non compressed HD movie?

→ Each color is encoded over 1 byte ($2^8 = 256$ possible gray levels).

→ Each pixel is composed of 3 colors (R,G,B).

Why does it matter?

An illustrative example

How much would weigh a non compressed HD movie?

- → Each color is encoded over 1 byte ($2^8 = 256$ possible gray levels).
- → Each pixel is composed of 3 colors (R,G,B).
- → There are 1280×720 pixels in a given frame.

Why does it matter?

An illustrative example

How much would weigh a non compressed HD movie?

- → Each color is encoded over 1 byte ($2^8 = 256$ possible gray levels).
- → Each pixel is composed of 3 colors (R,G,B).
- → There are 1280×720 pixels in a given frame.
- → There are 25 frames per second.

Why does it matter?

An illustrative example

How much would weigh a non compressed HD movie?

- → Each color is encoded over 1 byte ($2^8 = 256$ possible gray levels).
- → Each pixel is composed of 3 colors (R,G,B).
- → There are 1280×720 pixels in a given frame.
- → There are 25 frames per second.
- → An average movie is 1 hour 30 minutes long ($\equiv$ 5400 seconds).

Why does it matter?

An illustrative example

How much would weigh a non compressed HD movie?

- → Each color is encoded over 1 byte ($2^8 = 256$ possible gray levels).
- → Each pixel is composed of 3 colors (R,G,B).
- → There are 1280×720 pixels in a given frame.
- → There are 25 frames per second.
- → An average movie is 1 hour 30 minutes long ($\equiv$ 5400 seconds).

Leading to a total weight of: $5400 \times 25 \times 1280 \times 720 \times 3 =$

Why does it matter?

An illustrative example

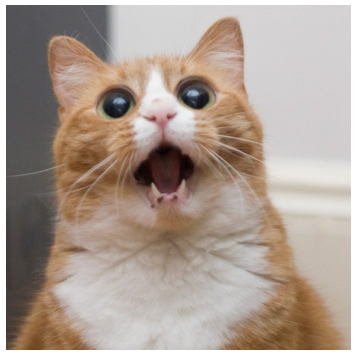
How much would weigh a non compressed HD movie?

- → Each color is encoded over 1 byte ($2^8 = 256$ possible gray levels).
- → Each pixel is composed of 3 colors (R,G,B).
- → There are 1280×720 pixels in a given frame.
- → There are 25 frames per second.
- → An average movie is 1 hour 30 minutes long ($\equiv$ 5400 seconds).

Leading to a total weight of: $5400 \times 25 \times 1280 \times 720 \times 3 =$ **373 248 Mb.**

$\simeq$ 80 single-side, single-layer DVDs !

And that is not even considering the sound...



Why does it matter?

If the previous example didn't convince you...

Data compression is interesting for several reasons:

→ To save space/memory.

++ Particularly true in the early days of computer science, when memory was über costly (it nonetheless remains the case nowadays).

The IBM Model 350 disk file with a storage space of 5MB from 1956 and a Micro SD Card

Source: https://ourworldindata.org/technological-progress

Why does it matter?

If the previous example didn't convince you...

Data compression is interesting for several reasons:

- → To save space/memory.
  - ++ Particularly true in the early days of computer science, when memory was über costly (it nonetheless remains the case nowadays).
- → To speed-up processing time (especially for images).
  - ++ Can also ease the transfer of voluminous data (less bandwidth needed).

Why does it matter?

If the previous example didn't convince you...

Data compression is interesting for several reasons:

- → To save space/memory.
  - ++ Particularly true in the early days of computer science, when memory was über costly (it nonetheless remains the case nowadays).
- → To speed-up processing time (especially for images).
  - ++ Can also ease the transfer of voluminous data (less bandwidth needed).
- → To increase compatibility.
  - ++ There are as many RAW formats as imaging sensor manufacturers. A given image viewer is very likely not to be compatible with all of them, but it will always be compatible with JPEG.

Why does it matter?

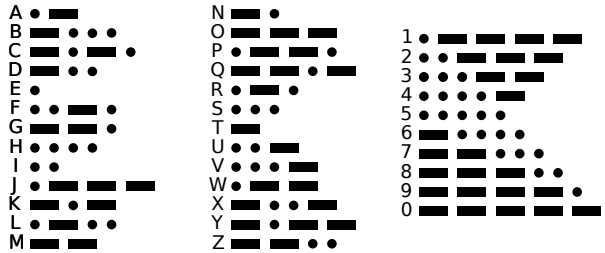
If the previous example didn't convince you...

Data compression is interesting for several reasons:

- → To save space/memory.
  - ++ Particularly true in the early days of computer science, when memory was über costly (it nonetheless remains the case nowadays).
- → To speed-up processing time (especially for images).
  - ++ Can also ease the transfer of voluminous data (less bandwidth needed).
- → To increase compatibility.
  - ++ There are as many RAW formats as imaging sensor manufacturers. A given image viewer is very likely not to be compatible with all of them, but it will always be compatible with JPEG.
- → To increase security.
  - ++ Data compression and cryptography are strongly linked: a compressed data is illegible for anyone who does not possess the correct decompression algorithm.
  - −− However, a corrupted compressed file is irremediably lost (can be a problem for some applications).

A brief historical review

1838 Morse code can be considered as the first compression algorithm since frequent letters ('e','t') are given shorter support.



1948 Claude Shannon establishes the Information Theory with its seminal paper *A Mathematical Theory of Communication*, laying the mathematical basis for data compression and transmission.

1952 David Huffman publishes the encoding algorithm that is now named after him.

1977 Abraham Lempel and Jacob Ziv introduce LZ77 as the first adaptive compression algorithm.

1984 Terry Welch improves LZ77 to give birth to the LZW algorithm.

1980s Computing power and storage capacities increase, allowing for the manipulation of sound and images and calling for lossy compression algorithms.

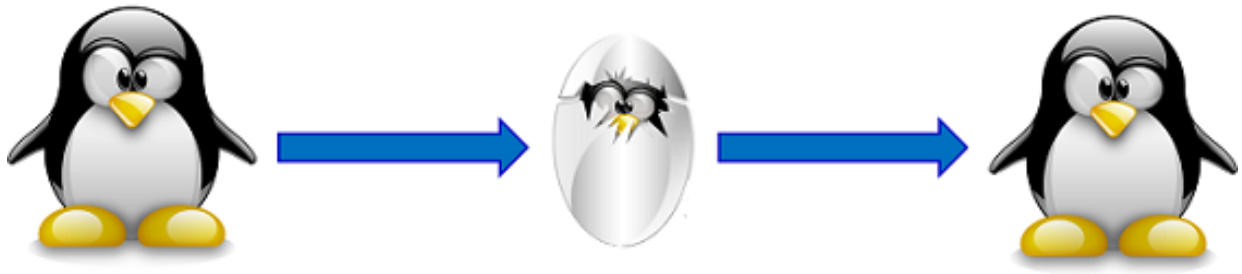
1992 The first JPEG standard is released (still evolving nowadays).

1993 Following JPEG, the first MPEG-1 standard is completed.

Data compression

Two wide and highly different families of algorithms

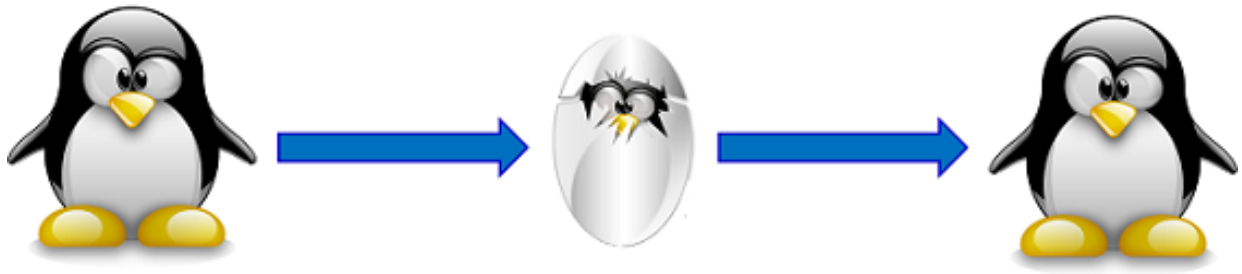
Lossless compression: exploits statistical redundancy of the data to represent it without losing any information. Data before and after decompression are identical.



Data compression

Two wide and highly different families of algorithms

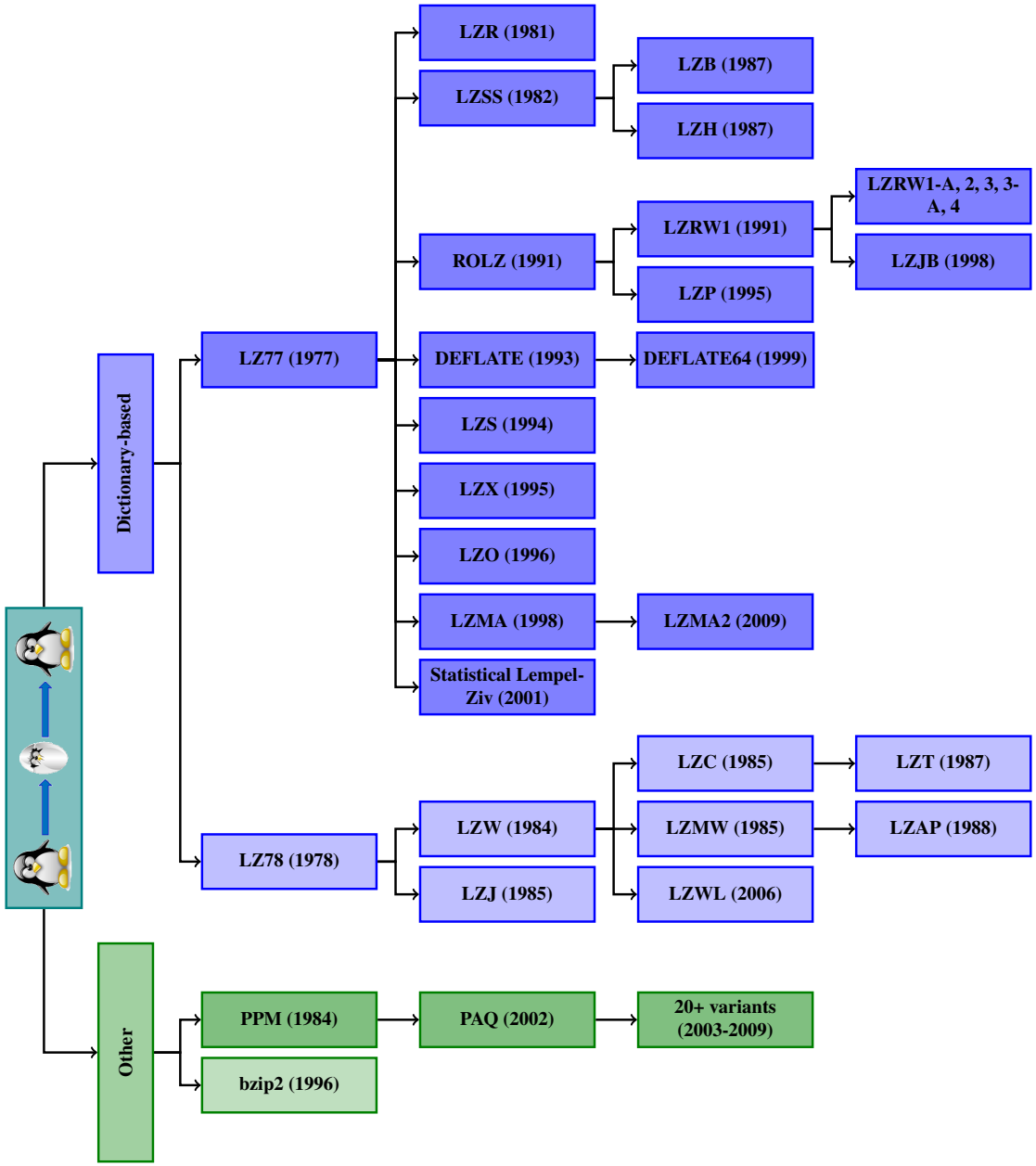
Lossless compression: exploits statistical redundancy of the data to represent it without losing any information. Data before and after decompression are identical.



Lossy compression: discard some information during the compression/decompression process. Data after compression is not the same as before compression.

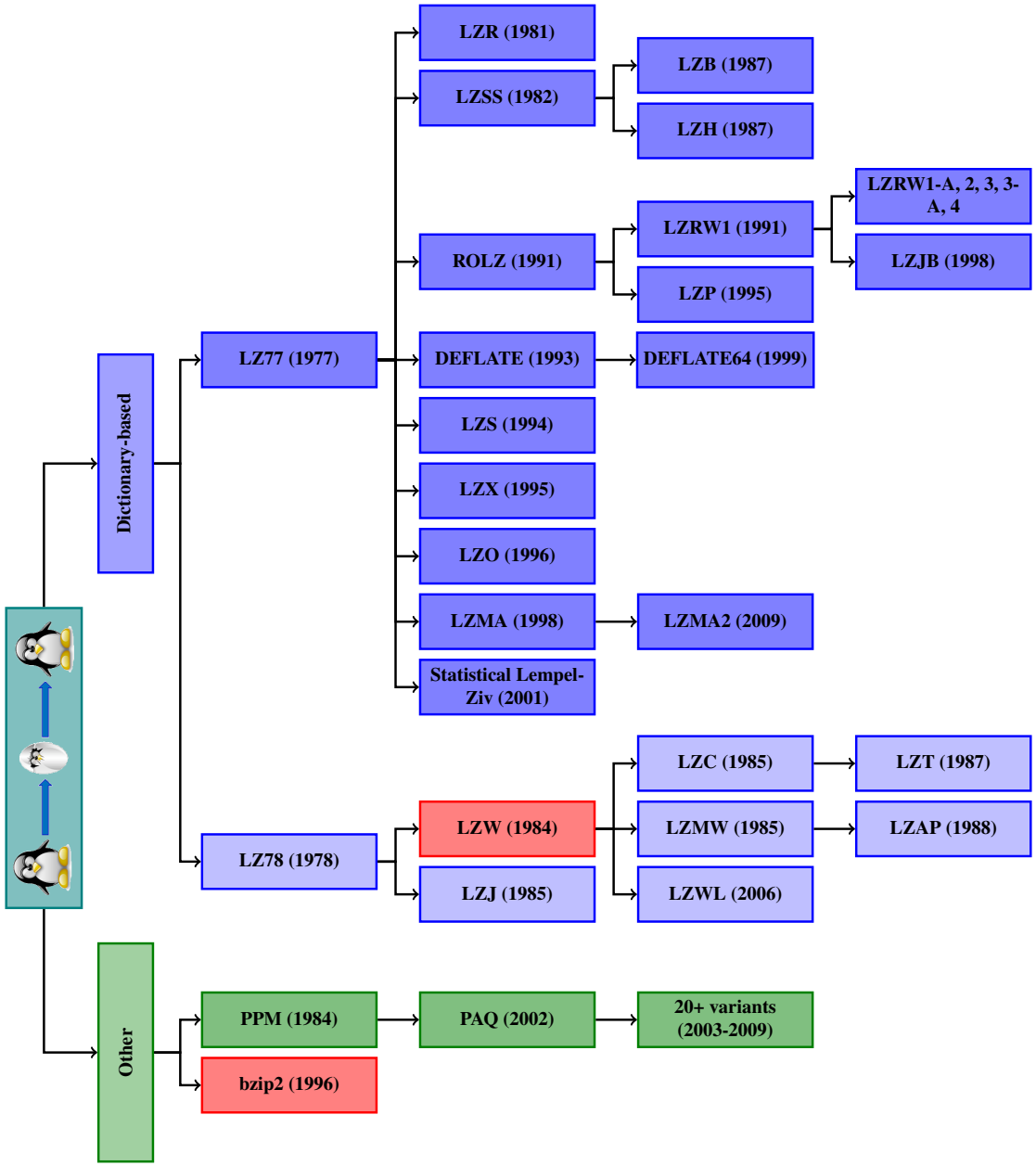
Lossless compression

- → Mature discipline (not so much research going on in the field nowadays).
- → Very efficient on noise-free data (such as text documents, executable files, etc), but performances degrade with noise.
- → Serves as base units for more elaborated lossy compression algorithms.



Lossless compression

- → Mature discipline (not so much research going on in the field nowadays).
- → Very efficient on noise-free data (such as text documents, executable files, etc), but performances degrade with noise.
- → Serves as base units for more elaborated lossy compression algorithms.



Lossy compression

→ Lossy compression algorithms assume that some part of the data to compress can be discarded, such that the human user won't notice the difference.

→ But how to evaluate which information is relevant and which one is redundant/useless in some data?

→ Well suited for sound and image compression, but not for text files (you may not want a piece of code to be altered after compression/decompression...).

→ Still an active field of research (wavelets, compressed sensing, etc).

high compression, bad quality

low compression, good quality

```
-rw-r--r-- 1 gtochon lrde 4,2K févr. 17 16:39 lemonhead_cat_highcompression.jpg
-rw-r--r-- 1 gtochon lrde  26K févr. 17 16:39 lemonhead_cat_lowcompression.jpg
```

General outline

1. Introduction
2. A flavor of Information Theory
3. Lossless compression algorithms
   - Run-length encoding algorithm
   - Huffman compression algorithm
   - bzip2 compression algorithm
   - LZW compression algorithm
4. Analog-to-digital conversion
5. Lossy compression algorithms
   - Some mathematical preliminaries to JPEG
   - JPEG compression algorithm for grayscale images
   - JPEG compression algorithm for color images
   - The one and only Principal Component Analysis