EXHIBIT 6

Data compression

In signal processing, **data compression**, **source coding**,[1] or **bit-rate reduction** involves encoding information using fewer bits than the original representation.[2] Compression can be either lossy or lossless. Lossless compression reduces bits by identifying and eliminating statistical redundancy. No information is lost in lossless compression. Lossy compression reduces bits by removing unnecessary or less important information.[3]

The process of reducing the size of a data file is often referred to as data compression. In the context of data transmission, it is called source coding; encoding done at the source of the data before it is stored or transmitted.[4] Source coding should not be confused with channel coding, for error detection and correction or line coding, the means for mapping data onto a signal.

Compression is useful because it reduces resources required to store and transmit data. Computational resources are consumed in the compression process and, usually, in the reversal of the process (decompression). Data compression is subject to a space–time complexity trade-off. For instance, a compression scheme for video may require expensive hardware for the video to be decompressed fast enough to be viewed as it is being decompressed, and the option to decompress the video in full before watching it may be inconvenient or require additional storage. The design of data compression schemes involves trade-offs among various factors, including the degree of compression, the amount of distortion introduced (when using lossy data compression), and the computational resources required to compress and decompress the data.[5][6]

Contents

Lossless

the process is reversible. Lossless compression is possible because most real-world data exhibits statistical redundancy. For example, an image may have areas of color that do not change over several pixels; instead of coding "red pixel, red pixel, ..." the data may be encoded as "279 red pixels". This is a basic example of run-length encoding; there are many schemes to reduce file size by eliminating redundancy.

The Lempel–Ziv (LZ) compression methods are among the most popular algorithms for lossless storage.[7] DEFLATE is a variation on LZ optimized for decompression speed and compression ratio, but compression can be slow. In the mid-1980s, following work by Terry Welch, the Lempel–Ziv–Welch (LZW) algorithm rapidly became the method of choice for most general-purpose compression systems. LZW is used in GIF images, programs such as PKZIP, and hardware devices such as modems.[8] LZ methods use a table-based compression model where table entries are substituted for repeated strings of data. For most LZ methods, this table is generate dynamically from earlier data in the input. The table itself is often Huffman encoded. Grammar-based codes like this can compress highly repetitive input extremely effectively, for instance, a biological data collection of the same or closely related species, a huge versioned document collection, internet archival, etc. The basic task of grammar-based codes is constructing a context-free grammar deriving a single string. Other practical grammar compression algorithms include Sequitur and Re-Pair.

The strongest modern lossless compressors use probabilistic models, such as prediction by partial matching. The Burrows–Wheeler transform can also be viewed as an indirect form of statistical modelling.[9] In a further refinement of the direct use of probabilistic modelling, statistical estimates can be coupled to an algorithm called arithmetic coding. Arithmetic coding is a more modern coding technique that uses the mathematical calculations of a finite-state machine to produce a string of encoded bits from a series of input data symbols. It can achieve superior compression compared to other techniques such as the better-known Huffman algorithm. It uses an internal memory state to avoid the need to perform a one-to-one mapping of individual input symbols to distinct representations that use an integer number of bits, and it clears out the internal memory only after encoding the entire string of data symbols. Arithmetic coding applies especially well to adaptive data compression tasks where the statistics vary and are context-dependent, as it can be easily coupled with an adaptive model of the probability distribution of the input data. An early example of the use of arithmetic coding was in an optional (but not widely used) feature of the JPEG image coding standard.[10] It has since been applied in various other designs including H.263, H.264/MPEG-4 AVC and HEVC for video coding.[11]

Lossy

In the late 1980s, digital images became more common, and standards for lossless image compression emerged. In the early 1990s, lossy compression methods began to be widely used.[8] In these schemes, some loss of information is accepted as dropping nonessential detail can save storage space. There is a corresponding trade-off between preserving information and reducing size. Lossy data compression schemes are designed by research on how people perceive the data in question. For example, the human eye is more sensitive to subtle variations in luminance than it is to the variations in color. JPEG image compression works in part by rounding off nonessential bits of information.[12] A number of popular compression formats exploit these perceptual differences, including psychoacoustics for sound, and psychovisuals for images and video.

Lossy image compression is used in digital cameras, to increase storage capacities. Similarly, DVDs, Blu Ray and streaming video use the lossy video coding format.

In lossy audio compression, methods of psychoacoustics are used to remove non-audible (or less audible) components of the audio signal. Compression of human speech is often performed with even more specialized techniques; speech coding is distinguished as a separate discipline from general-purpose audio compression. Speech coding is used in internet telephony, for example, audio compression is used for CD ripping and is decoded by the audio players.[9]

Theory

The theoretical background of compression is provided by information theory (which is closely related to algorithmic information theory) for lossless compression and rate–distortion theory for lossy compression. These areas of study were essentially created by

Machine learning

There is a close connection between machine learning and compression: a system that predicts the posterior probabilities of a sequence given its entire history can be used for optimal data compression (by using arithmetic coding on the output distribution) while an optimal compressor can be used for prediction (by finding the symbol that compresses best, given the previous history). Thi equivalence has been used as a justification for using data compression as a benchmark for "general intelligence."[14][15][16]

Feature space vectors

However a new, alternative view can show compression algorithms implicitly map strings into implicit feature space vectors, and compression-based similarity measures compute similarity within these feature spaces. For each compressor C(.) we define an associated vector space ℵ, such that C(.) maps an input string x, corresponds to the vector norm ||~x||. An exhaustive examination of the feature spaces underlying all compression algorithms is precluded by space; instead, feature vectors chooses to examine three representative lossless compression methods, LZW, LZ77, and PPM.[17]

Data differencing

Data compression can be viewed as a special case of data differencing.[18][19] Data differencing consists of producing a *difference* given a *source* and a *target,* with patching producing a *target* given a *source* and a *difference,* while data compression consists of producing a compressed file given a target, and decompression consists of producing a target given only a compressed file. Thus, one can consider data compression as data differencing with empty source data, the compressed file corresponding to a "difference from nothing." This is the same as considering absolute entropy (corresponding to data compression) as a special case of relative entropy (corresponding to data differencing) with no initial data.

When one wishes to emphasize the connection, one may use the term *differential compression* to refer to data differencing.

Uses

Audio

Audio data compression, not to be confused with dynamic range compression, has the potential to reduce the transmission bandwidth and storage requirements of audio data. Audio compression algorithms are implemented in software as audio codecs. Lossy audio compression algorithms provide higher compression at the cost of fidelity and are used in numerous audio applications. These algorithms almost all rely on psychoacoustics to eliminate or reduce fidelity of less audible sounds, thereby reducing the space required to store or transmit them.[2]

In both lossy and lossless compression, information redundancy is reduced, using methods such as coding, pattern recognition, and linear prediction to reduce the amount of information used to represent the uncompressed data.

The acceptable trade-off between loss of audio quality and transmission or storage size depends upon the application. For example, one 640 MB compact disc (CD) holds approximately one hour of uncompressed high fidelity music, less than 2 hours of music compressed losslessly, or 7 hours of music compressed in the MP3 format at a medium bit rate. A digital sound recorder can typically store around 200 hours of clearly intelligible speech in 640 MB.[20]

Lossless audio compression produces a representation of digital data that decompress to an exact digital duplicate of the original audio stream, unlike playback from lossy compression techniques such as Vorbis and MP3. Compression ratios are around 50–60% of original size,[21] which is similar to those for generic lossless data compression. Lossless compression is unable to attain high compression ratios due to the complexity of waveforms and the rapid changes in sound forms. Codecs like FLAC, Shorten, and TTA whiten or flatten the spectrum, thereby allowing traditional lossless compression to work more efficiently. The process is reversed upon decompression.

When audio files are to be processed, either by further compression or for editing, it is desirable to work from an unchanged original (uncompressed or losslessly compressed). Processing of a lossily compressed file for some purpose usually produces a final result inferior to the creation of the same compressed file from an uncompressed original. In addition to sound editing or mixing, lossless audio compression is often used for archival storage, or as master copies.

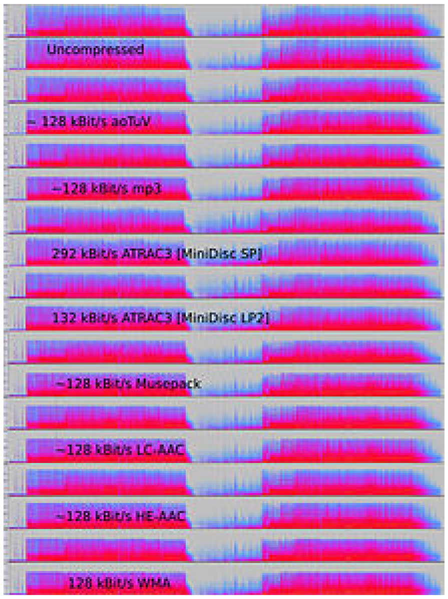
A number of lossless audio compression formats exist. Shorten was an early lossless format. Newer ones include Free Lossless Audio Codec (FLAC), Apple's Apple Lossless (ALAC), MPEG-4 ALS, Microsoft's Windows Media Audio 9 Lossless (WMA Lossless), Monkey's Audio, TTA, and WavPack. See list of lossless codecs for a complete listing.

Some audio formats feature a combination of a lossy format and a lossless correction; this allows stripping the correction to easily obtain a lossy file. Such formats include MPEG-4 SLS (Scalable to Lossless), WavPack, and OptimFROG DualStream.

Other formats are associated with a distinct system, such as:

- Direct Stream Transfer, used in Super Audio CD
- Meridian Lossless Packing, used in DVD-Audio, Dolby TrueHD, Blu-ray and HD DVD

Lossy audio compression



Comparison of spectrograms of audio in an uncompressed format and several lossy formats. The lossy spectrograms show bandlimiting of higher frequencies, a common technique associated with lossy audio compression.

Lossy audio compression is used in a wide range of applications. In addition to the direct applications (MP3 players or computers), digitally compressed audio streams are used in most video DVDs, digital television, streaming media on the internet, satellite and cable radio, and increasingly in terrestrial radio broadcasts. Lossy compression typically achieves far greater compression than lossless compression (5–20% of the original size, rather than 50–60%), by discarding less-critical data.[22]

The innovation of lossy audio compression was to use psychoacoustics to recognize that not all data in an audio stream can be perceived by the human auditory system. Most lossy compression reduces perceptual redundancy by first identifying perceptually irrelevant sounds, that is, sounds that are very hard to hear. Typical examples include high frequencies or sounds that occur at the same time as louder sounds. Those sounds are coded with decreased accuracy or not at all.

Due to the nature of lossy algorithms, audio quality suffers when a file is decompressed and recompressed (digital generation loss). This makes lossy compression unsuitable for storing the intermediate results in professional audio engineering applications, such as sound editing and multitrack recording. However, they are very popular with end users (particularly MP3) as a megabyte can store about a minute's worth of music at adequate quality.

Coding methods

To determine what information in an audio signal is perceptually irrelevant, most lossy compression algorithms use transforms such as the modified discrete cosine transform (MDCT) to convert time domain sampled waveforms into a transform domain. Once transformed, typically into the frequency domain, component frequencies can be allocated bits according to how audible they are. Audibility of spectral components calculated using the absolute threshold of hearing and the principles of simultaneous masking—the phenomenon wherein a signal is masked by another signal separated by frequency

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase®
Smarter legal research.