CSEP 521 **Igor Zinkovsky**

Data Compression Algorithms

The challenge to store and transmit various data has been around for centuries. In our generation digital computers have transformed the way most data is stored and transmitted. This paper will concentrate on one of the most common challenges surrounding data, which is data compression. Digital data compression has been an increasingly important subject to all of computer science, and is one of the most interesting ones. Advances in numerous areas of computer hardware and software have enabled variety of new business and consumer applications, where enormous amounts of data get manipulated every instant. Data compression plays a central role in supporting, and in many cases enabling, such applications. It's clear; data compression plays a very significant part in all of computer science.

Depending on the types of data being manipulated, applications have different requirements for data compression engines. Picking the right compression algorithm involves performing tradeoff analysis on various factors which are important to the application. Such factors include degree of compression, amount of distortion introduced (in case of lossy compression), and the amount of work needed to compress/decompress the data. Some applications could be more interested in getting smaller compression ratios while gaining on decompression speed, while others will pick algorithms for maximizing compression ratios even if compression and decompression take long time. For example, visual data applications (such

as movie players) have a requirement to show data in real time, and therefore are willing to tradeoff smaller compression ratios for faster decompression rates.

Data compression falls into two categories: lossless and lossy. Lossless compression allows applications to get back the exact data that was originally compressed, while lossy compression causes parts of data to be distorted at the gain of higher degrees of compression. Lossless compression algorithms typically exploit data redundancy, and aim to increase effective data density, allowing the application to get back the original data in its entirety. Lossy algorithms exploit the fact that it is acceptable for some parts of the data to be lost in exchange for greater degree of compression Unlike lossless compression, it's not possible to get the original data after decompression. This detail might or might not be important for different types of data.

In practice, lossless algorithms are most often used to compress the following three types of data: text, images, and sound. One of the most known lossless compression algorithms is LZW (an abbreviation of Lempel-Ziv-Welch), which is a one of the LZ-family algorithms. In general, LZ algorithms compress data by replacing duplicate chunks with references into some table which holds the actual data. For a long time LZW algorithm provided the best compression ratio than any known algorithm at the time. It would typically be able to compress large English texts by the factor of 2. While any lossless compression algorithm will work for any type of data being compressed, certain algorithms are more optimized to work with certain types of data. For example, sound data would not be a good candidate to be compressed using LZW algorithm.

Most widely used consumer image compression algorithms are lossy. Lossless compression is typically used for images requiring precise preservation of all details, such as medical imagery, technical drawings, or images made to be archived. Lossy compression is especially useful for photographs, where small (and often imperceptible) loss of data would result in dramatic increases of compression factor. From known image compression algorithms BMP and GIF are most popular lossless algorithms, while JPEG is the most popular lossy method (thought variations of lossless JPEG method are available). Important metrics when analyzing image compression algorithms are scalability, quality, degree of compression, and compression/decompression rates.

Audio compression algorithms are typically referred to as codecs. With increased network bandwidth, lossless codecs (such as Dolby, TrueHD) are becoming more and more popular. In contrast to text data, finding duplications in audio recordings is much less frequent. Sound data is much more complex than text, resulting in more chaotic sequences, and thus is less compressible using lossless techniques. Also, because sound data is typically a quantized collection of samples, long collections of consecutive bytes are typically absent. Lossy algorithms do a much better job compressing audio data. One of the most fundamental guidelines in lossy audio compression is recognition that much of the data in an audio stream cannot be perceived by human auditory system. For example, most humans cannot hear sounds of frequencies higher than 22 KHz, which is used by a variety of lossy audio codecs. Typical metrics used to examine audio codecs are perceived audio quality, compression factor, and compression/decompression rates.

Important property of lossless compression algorithms is there will always be some data for which the resultant (compressed) data will actually be larger than original. For this reason many algorithm provide an escape facility, where compression gets entirely turned off in case compressed files become larger.

LZX is another algorithm of LZ family. LZX was invented by Jonathan Forbes in 1995, and is now part of many Microsoft applications. As other LZ algorithms, LZX is a lossless algorithm primarily used to compress text and binary data. In 1997, Jonathan Forbes joined Microsoft, and Microsoft started using LZX for Cabinet and CHM files. The strength of LZX is still shown today, as Windows Vista is reported to use LZX its WIM (Windows Imaging Format) installation files.

Bibliography:

LZX Algorithm:
http://en.wikipedia.org/wiki/LZX_%28algorithm%29
LZW Algorithm:
http://en.wikipedia.org/wiki/LZW
Data compression:
http://en.wikipedia.org/wiki/Data_compression
Lossless data compression:
http://en.wikipedia.org/wiki/Lossless_compression
“Lossless data compression software benchmarks / comparisons”:
http://www.maximumcompression.com/index.html