(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
30 August 2001 (30.08.2001)

PCT

(10) International Publication Number
WO 01/63772 A1

(51) International Patent Classification[7]: H03M 7/34, H04N 1/415

(21) International Application Number: PCT/US01/05722

(22) International Filing Date: 22 February 2001 (22.02.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/513,309 25 February 2000 (25.02.2000) US

(71) Applicant: PHYSICAL OPTICS CORPORATION [US/US]; 20600 Gramercy Place, Building 100, Torrance, CA 90501-1821 (US).

(72) Inventors: TERNOVSKIY, Igor, V.; 5700 Ravensbur Drive, #209, Rancho Palos Verdes, CA 90275 (US). DEVIVYE, Aleksandr, A.; 6517 Kester Avenue, #4, Van Nuys, CA 91411 (US). ROTENBERG, Joseph; 719 Price Drive, Burbank, CA 91504 (US). LIN, Freddie; 800 Calle De Arboles, Redondo Beach, CA 90277 (US).

(74) Agents: NILLES, Andrew, J. et al.; Nilles & Nilles, S.C., Firstar Center, Suite 2000, 777 East Wisconsin Avenue, Milwaukee, WI 53202 (US).

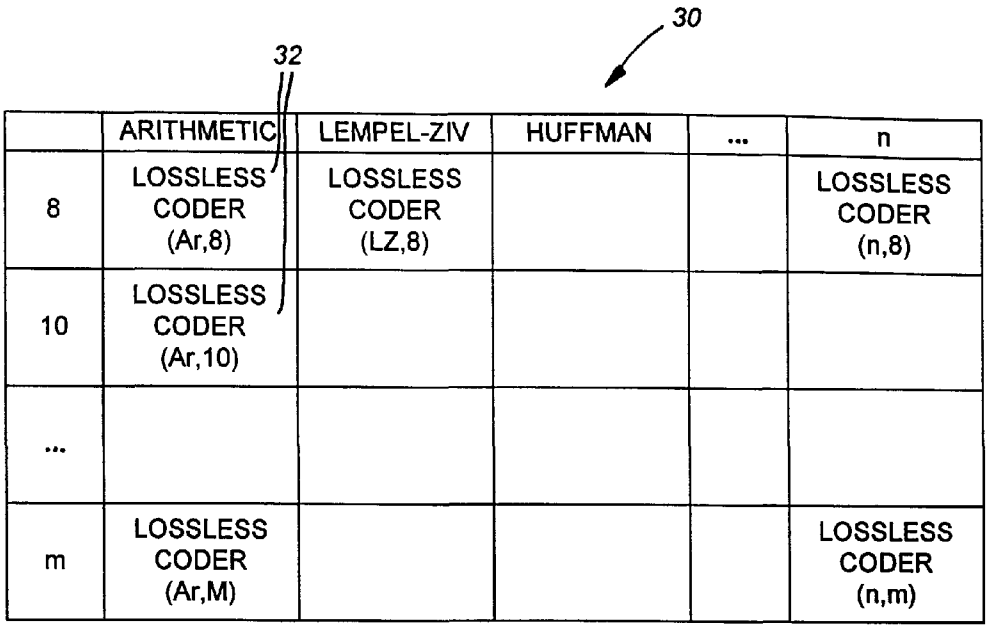
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: METHOD AND APPARATUS FOR OPTIMIZED LOSSLESS COMPRESSION USING A PLURALITY OF CODERS



(57) Abstract: A method of lossless compression of a stream of data first includes using a plurality of lossless coders to compress a test portion of the data stream (30). Once the test portion is compressed, the method determines a performance characteristic(s) associated with each of the lossless coders (32). Then the method selects one of the lossless coders based on the performance characteristic(s) and encodes a first portion of the data stream with the selected coder. Thereafter, the method includes repeating the using, determining, selecting and encoding steps for another test portion and a second portion of the data stream. Notably, the repeating step may include selecting a different one of the lossless coders.

WO 01/63772 A1

METHOD AND APPARATUS FOR OPTIMIZED LOSSLESS COMPRESSION USING A PLURALITY OF CODERS

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention is directed to data compression techniques and, more particularly, to a method and apparatus for selecting among different types of lossless compression coders to optimize system performance.

2. Description of the Related Art

Data compression operates to minimize the number of bits used to store or transmit information and encompasses a wide array of software and hardware compression techniques. Notably, depending on the type of data to be compressed and any number of other factors, particular compression techniques can provide markedly superior performance in terms of compression ratio and coding speed.

Generally, data compression includes taking a stream of symbols or phrases and converting them into codes that are smaller (in bit length) than the original data. Known compression techniques and algorithms can be divided into two major families including lossy and lossless. Lossy data compression can be used to greatly increase data compression ratios; however, increased compression comes at the expense of a certain loss in accuracy. As a result, lossy compression typically is implemented only in those instances in which some data loss is acceptable. For example, lossy compression is used effectively used when applied to digitized voice signals and graphics images. Lossless compression, on the other hand, is a family of data compression that utilizes techniques designed to generate an exact duplicate of the input data stream after a compression/decompression cycle. This type of compression is necessary when storing database records, word processing files, etc., where loss of information is absolutely unacceptable. The present invention is directed to lossless data compression.

Some lossless compression algorithms use information theory to generate variable length codes when given a probability table for a given set of symbols. The decision to output a certain code for a particular symbol or set of symbols (i.e., message) is based on a model. The model is a set of rules used to process input

messages, and in response, determine which codes to output. An algorithm or program uses the model to analyze the symbols (e.g., determine a probability associated with the symbol) and then outputs an appropriate code based on that processing. There are any number of ways to model data, all of which can use the same coding technique to produce their output. In general, to compress data efficiently, a model should be selected that predicts symbols or phrases with high probabilities because symbols or messages that have a high probability have a low information content, and therefore require fewer bits to encode. The next step is to encode the symbols using a particular lossless coder.

Conventionally, lossless compression coders can be grouped according to whether they implement statistical modeling or dictionary-based modeling. Statistical modeling reads and encodes a single symbol at a time using the probability of the character's appearance, while dictionary-based modeling uses a single code to replace strings of symbols. Notably, in dictionary-based modeling, the model is significantly more important than in statistical-based modeling because problems associated with encoding every symbol are significantly reduced.

One form of statistical data compression is known as Shannon-Fano (S-F) coding. S-F coding was developed to provide variable-length bit coding so as to allow coding symbols with exactly (or a close approximation to) the number of bits of information that the message or symbol contains. S-F coding relies on knowing the probability of each symbol's appearance in a message. After the probabilities are determined, a table of codes is constructed with each code having a different number of bits (advantageously, symbols with low probabilities have more bits). One problem with a coding technique such as this is that it creates variable length codes that have an integral number of bits, even though the information to be coded likely will require a non-integral number of bits.

Another type of coding, Huffman coding, is similar to S-F coding in that it creates variable length codes that are an integral number of bits, but it utilizes a completely different algorithm. Generally, S-F and Huffman codings are close in performance but Huffman coding, it has been determined, always at least equals the

efficiency of S-F coding so it is therefore preferred, especially since both algorithms take a similar amount of processing power. Although Huffman coding is relatively easy to implement and economical for both coding and decoding, it is inefficient due to its use of an integral number of bits per code as in S-F coding. If a particular symbol is determined to have an information content (i.e., entropy) of 1.5 bits, a Huffman coder will generate a code having a bit count that is either one or two bits. Generally, if a statistical method could assign a 90% probability to a given symbol, the optimal code size would be 0.15 bits; however, Huffman or S-F coding likely would assign a one bit code to the symbol, which is six times larger than necessary.

In view of this problem associated with utilizing an integral number of bits, arithmetic coding was developed. Arithmetic coding replaces a stream of input symbols with a single floating point output number, and bypasses the step of replacing an input symbol with a specific code. Because an arithmetic code is not limited to being optimal only when the symbol probabilities are integral powers of one-half (which is most often not the case), it attains the theoretical entropy of the symbol to be coded, thus maximizing compression efficiency for any known source. In other words, if the entropy of a given character is 1.5 bits, arithmetic coding uses 1.5 bits to encode the symbol, an impossibility for Huffman and Shannon-Fano coding. Although arithmetic coding is extremely efficient, it consumes rather large amounts of computing resources, both in terms of CPU power and memory. This is due to the fact that sophisticated models that demand a significant amount of memory must be built, and that the algorithm itself requires a significant amount of computational operations.

In an alternative to the above types of lossless coding, known as substitutional or dictionary-based coding, dictionary-based compression algorithms replace occurrences of particular phrases (i.e., groups of bytes) in a data stream with a reference to a previous occurrence of those phrases. Unlike the above systems that achieve compression by encoding symbols into bit strings that use fewer bits than the original symbols, dictionary-based algorithms do not encode single symbols. Rather, dictionary-based compression techniques encode variable

length strings of symbols as single "tokens." It is these tokens that form an index to a phrase dictionary. Because the tokens are smaller than the phrases they replace, compression occurs. Two main classes of dictionary-based compression schemes are known as the LZ77 and LZ78 compression algorithms of the Lempel-Ziv family of compression coders. Notably, dictionary-based coding is utilized extensively in desktop general purpose compression and has been implemented by Compuserve Information Service to encode bit-mapped graphical images. For example, the GIF format uses a LZW variant to compress repeated sequences and screen images. Although dictionary-based compression techniques are very popular forms of compression, the disadvantage of such algorithms is that a more sophisticated data structure is needed to handle the dictionary.

Overall, as communication mediums such as the internet expand, data compression will continue to be extremely important to the efficient communication of data, with different compression algorithms providing particular advantages in particular arenas. There are many types of data compression methods that are being implemented in the art, including those described above as well as others. In addition, there are many variants associated with each type of known compression algorithm and many improvements have been developed. Again, depending on any number of factors associated with the system and the type of data being compressed, each may be preferred to provide optimum data encoding.

Because different ones of known coding techniques provide unique benefits depending upon various operational factors including the data to be encoded, a lossless compression system that selectively encodes data with different types of coders was desired. The telecommunications industry, in particular, is in need of a system which implements different types of coders, especially when the incoming data is received from multiple sources that provide different types of unknown data, i.e., when different portions of the data stream would be optimally compressed with different coding techniques.

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.