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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
12 July 2001 (12.07.2001)

PCT

(10) International Publication Number
WO 01/50325 A2

(51) International Patent Classification[7]: G06F 17/00

(21) International Application Number: PCT/US00/35672

(22) International Filing Date: 29 December 2000 (29.12.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/174,305 3 January 2000 (03.01.2000) US

(71) Applicant: EFECKTA TECHNOLOGIES CORPORATION [US/US]; 1370 Bob Adams Drive, Steamboat Springs, CA 80487 (US).

(72) Inventors: AVERY, Caleb; -. TOBELMANN, Ralph; -.

(74) Agents: SACHS, Robert et al.; Fenwick & West LLP, Two Palo Alto Square, Palo Alto, CA 94306 (US).

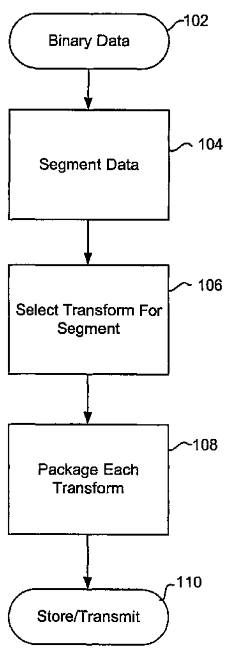
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, 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:
— *Without international search report and to be republished upon receipt of that 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:** EFFICIENT AND LOSSLESS CONVERSION FOR TRANSMISSION OR STORAGE OF DATA



(57) **Abstract:** A system and method for lossless data compression. A mathematical transform equivalent to the content value of the data, and taking fewer bits to represent, is found.

WO 01/50325 A2

Efficient and Lossless Conversion for Transmission or Storage of Data

This application claims priority under 35 U.S.C. §119(e) from U.S. Patent Application serial No. 60/174,305, filed January 3, 2000, which is incorporated by reference in its entirety.

Background

Field of the Invention

This invention relates to data transformation, more particularly, to lossless data compression.

Background of the Invention

Existing compression technology focuses on finding and removing redundancy in the input binary data. Early compression approaches focused on the format of the data. These format approaches utilize run-length encoding (RLE) and variations of frequency mapping methods. These pattern-coding approaches work well for ASCII character data, but never reached the compression potential for other data formats.

Advances in compression technology evolved from information theory, particularly Claude Shannon's work on information entropy. The bulk of this work is statistical in nature. Shannon-Fano and Huffman encoding build probability trees of symbols in descending order of their occurrence in the source data, allowing the generation of "good" variable-size codes. This is often referred to as entropy coding. Compression is accomplished because more frequently occurring binary patterns are assigned shorter codes, allowing for a reduction in the overall average of bits required for a message.

Shannon-Fano and Huffman encoding are optimal only when the probability of a pattern's occurrence is a negative power of 2. These methods engendered a number of adaptive versions that optimize the probability trees as the data varies.

Arithmetic Coding overcame the negative power of 2 probabilities problem by assigning one (normally long) code to the entire data. This method reads the data, symbol by symbol, and appends bits to the output code each time more patterns are recognized.

The need for more efficiency in text encoding led to the development and evolution of dictionary encoding, typified by LZ family of algorithms developed by J. Ziv and A. Lempel. These methods spawned numerous variations. In these methods, strings of symbols (a dictionary) are built up as they are encountered, and then coded as tokens. Output is then a mix of an index and raw data.

As with entropy coding, dictionary methods can be static, or adaptive. Variants of the LZ family make use of different techniques to optimize the dictionary and its index. These techniques include: search buffers, look-ahead buffers, history buffers, sliding windows, hash tables, pointers, and circular queues. These techniques serve to reduce the bloat of seldom-used dictionary entries. The popularity of these methods is due to their simplicity, speed, reasonable compression rates, and low memory requirements.

Different types of information tend to create specific binary patterns. Redundancy or entropy compression methods are directly dependent upon symbolic data, and the inherent patterns that can be recognized, mapped, and reduced. As a result, different methods must be optimized for different types of information. The compression is as efficient as the method of modeling the underlying data. However, there are limits to the structures that can be mapped and reduced.

The redundancy-based methodologies are limited in application and/or performance. In general, entropy coding either compromises speed or compression when addressing the limited redundancy that can be efficiently removed. Typically, these methods have very low compression gain. The primary advantage is that entropy coding can be implemented to remain lossless.

Lossy compression can often be applied to diffuse data such as data representing speech, audio, image, and video. Lossy compression implies that the data cannot be reconstructed exactly. Certain applications can afford to lose data during compression and reconstitution because of the limitations of human auditory and visual systems in interpreting the information. Perceptual coding techniques are used to exploit these limitations of the human eyes and ears. A perceptual coding model followed by entropy encoding, which uses one of the previously discussed techniques, produces effective compression. However, a unique model (and entropy coder) is needed for each type of data because the requirements are so different. Further, the lossy nature of such compression techniques mean the results lose some fidelity, at times noticable, from the original, and make them unsuitable for many purposes.

Thus, a method for compression that is both lossless and capable of high compression gain is needed.

SUMMARY OF THE INVENTION

The present invention compresses binary data. The data is split into segments. Each of these segments has a numerical value. A transform, along with state information for that transform, is selected for each segment. The numerical value of the transform with its state information is equal to the numerical value of the segment. The transform, state information and packet overhead are packaged into a transform packet. The bit-length of the transform packet is compared to the bit-length of a segment packet that includes the raw segment and any necessary packet overhead. The packet with the smaller bit-length is chosen and stored or transmitted. After reception of the packets, or retrieval of the packets from storage, the numerical value of each segment is recalculated from the transform and state information, if necessary. The segments are recombined to reconstitute the original binary data.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a flow chart showing an overview of the compression process.

Fig. 2 is a flow chart showing an overview of the recovery process.

Fig. 3 is a block diagram of the compression system and illustrates the general flow of data through the compression system.

Fig. 4 is a flow chart of the pre-processor.

Fig. 5 is a flow chart of the transform engine.

Fig. 6 is a flow chart of a transform process.

Fig. 7 illustrates finding a set of favorable state information without testing every possible set of state information.

Fig. 8 illustrates partial solutions within a segment.

Detailed Description of the Preferred Embodiments

Fig. 1 is a flow chart showing an overview of the compression process 100. The initial input 102 is binary data. Any piece of binary data is simply a number expressed in binary form. Thus, any piece of binary data has a numerical value. This numerical value is the data's "content value."

The input binary data 102 is split 104 into segments, if necessary. The input binary data may be short enough that it is not further split. Each segment has a content value. The content value of each segment is identified and a transform along with appropriate state information is selected and tested 106 for each segment. A general transform is capable of representing many values. The state information provides the information necessary to specify an exact value for the transform. The term "state information" includes any variables, coefficients, remainders, or any other information necessary to set the specific numerical value for the transform. In some embodiments, "packet overhead," is added to the transform and state information. The packet overhead includes any information beyond the transform and state information needed to allow later recalculation of the original segment and reconstitution of the original input binary data.

The transform with its state information has the identical numerical value as the corresponding segment. The following equation represents the transform concept:

$$M=T(State\ Information),$$

where M is the content value of the segment, and T is the transform. The transform is an arithmetic transform, a logical transform, or another mathematical transform.

The transform with its state information and packet overhead has a representation efficiency gain ("REG"). The REG is a measurement of the efficiency of the transform, state information, and any packet overhead. The REG is defined as the ratio of Log_2M/ Log_2T, where Log_2M is the number of binary bits required to represent M, and Log_2T is the number of binary bits required to decodably represent the transform, state information, and packet

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.