

(19)



Europäisches Patentamt
European Patent Office
Office européen des brevets



(11)

EP 1 142 129 B1

(12)

EUROPEAN PATENT SPECIFICATION

(45) Date of publication and mention
of the grant of the patent:
16.06.2004 Bulletin 2004/25

(51) Int Cl.7: **H03M 7/42**

(21) Application number: **99966062.4**

(86) International application number:
PCT/US1999/029108

(22) Date of filing: **07.12.1999**

(87) International publication number:
WO 2000/036752 (22.06.2000 Gazette 2000/25)

(54) VARIABLE TO VARIABLE LENGTH ENTROPY ENCODING

ENTROPIEKODIERUNG VON VARIABLER ZU VARIABLER LÄNGE

CODAGE ENTROPIQUE DU TYPE LONGUEUR VARIABLE POUR LONGUEUR VARIABLE

(84) Designated Contracting States:
**AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU
MC NL PT SE**

(56) References cited:
**EP-A- 0 283 735 EP-A- 0 535 571
WO-A-98/40969 US-A- 4 744 085
US-A- 5 959 560**

(30) Priority: **14.12.1998 US 211294**

(43) Date of publication of application:
10.10.2001 Bulletin 2001/41

(73) Proprietor: **MICROSOFT CORPORATION
Redmond, Washington 98052-6399 (US)**

(72) Inventors:
• **CHEN, Wei-ge
Issaquah, WA 98029 (US)**
• **LEE, Ming-Chieh
Bellevue, WA 98006 (US)**

(74) Representative: **Grünecker, Kinkeldey,
Stockmair & Schwanhäusser Anwaltssozietät
Maximilianstrasse 58
80538 München (DE)**

- **PATENT ABSTRACTS OF JAPAN vol. 1998, no. 01, 30 January 1998 (1998-01-30) & JP 09 232968 A (KOKUSAI DENSHIN DENWA CO LTD <KDD>), 5 September 1997 (1997-09-05) -& US 5 883 589 A (WADA MASAHIRO ET AL) 16 March 1999 (1999-03-16)**
- **PATENT ABSTRACTS OF JAPAN vol. 012, no. 118 (E-600), 13 April 1988 (1988-04-13) & JP 62 247626 A (FUJI PHOTO FILM CO LTD), 28 October 1987 (1987-10-28)**
- **SHAMOONT.; HEEGARD C.: 'A Rapidly Adaptive Lossless Compression Algorithm for High Fidelity Audio Coding', 29 March 1994, IEEE COMP. SOC., LOS ALAMITOS, CA, USA**
- **VON RODEN T.: 'H.261 AND MPEG1- a comparison' IEEE 15TH ANNUAL INTERNATIONAL CONFERENCE ON COMPUTERS AND COMMUNICATION 27 March 1996, NEW YORK, USA, pages 65 - 71**

EP 1 142 129 B1

Note: Within nine months from the publication of the mention of the grant of the European patent, any person may give notice to the European Patent Office of opposition to the European patent granted. Notice of opposition shall be filed in a written reasoned statement. It shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European Patent Convention).

Description

[0001] The invention generally relates to data compression, and more specifically relates to a form of entropy coding.

[0002] In a typical audio coding environment, data is represented as a long sequence of symbols which is input to an encoder. The input data is encoded by an encoder, transmitted over a communication channel (or simply stored), and decoded by a decoder. During encoding, the input is pre-processed, sampled, converted, compressed or otherwise manipulated into a form for transmission or storage. After transmission or storage, the decoder attempts to reconstruct the original input.

[0003] Audio coding techniques can be generally categorized into two classes, namely the time-domain techniques and frequency-domain ones. Time-domain techniques, e.g., ADPCM, LPC, operate directly in the time domain while the frequency-domain techniques transform the audio signals into the frequency domain where the actual compression happens. The frequency-domain codecs can be further separated into either sub-band or transform coders although the distinction between the two is not always clear. Processing an audio signal in the frequency domain is motivated by both classical signal processing theories and human perception models (e.g., psychoacoustics). The inner ear, specifically the basilar membrane, behaves like a spectral analyzer and transforms the audio signal into spectral data before further neural processing proceeds.

[0004] The frequency-domain audio codecs often take advantage of many kinds of auditory masking that are going on with the human hear system to modify the original signal and eliminate a great many details/redundancies. Since the human ears are not capable of perceiving these modifications, efficient compression is achieved. Masking is usually conducted in conjunction with quantization so that quantization noise can be conveniently "masked." In modern audio coding techniques, the quantized spectral data are usually further compressed by applying entropy coding, e.g., Huffman coding.

[0005] Compression is required because a fundamental limitation of the communication model is that transmission channels usually have limited capacity or bandwidth. Consequently, it is frequently necessary to reduce the information content of input data in order to allow it to be reliably transmitted, if at all, over the communication channel. Over time, tremendous effort has been invested in developing lossless and lossy compression techniques for reducing the size of data to transmit or store. One popular lossless technique is Huffman encoding, which is a particular form of entropy encoding.

[0006] Entropy coding assigns code words to different input sequences, and stores all input sequences in a code book. The complexity of entropy encoding depends on the number m of possible values an input se-

quence X may take. For small m , there are few possible input combinations, and therefore the code book for the messages can be very small (e.g., only a few bits are needed to unambiguously represent all possible input sequences). For digital applications, the code alphabet is most likely a series of binary digits {0, 1}, and code word lengths are measured in bits.

[0007] If it is known that input is composed of symbols having equal probability of occurring, an optimal encoding is to use equal length code words. But, it is not typical that an input stream has equal probability of receiving any particular message. In practice, certain messages are more likely than others, and entropy encoders take advantage of this to minimize the average length of code words among expected inputs. Traditionally, however, fixed length input sequences are assigned variable length codes (or conversely, variable length sequences are assigned fixed length codes).

[0008] International Publication No. WO 98/40969 describes a system for compressing an ASCII or similarly encoded text file. Pipelining of certain data compression algorithms is described in Bailey and Mulkamala, "Pipelining Data Compression Algorithms", *The Computer Journal*, vol. 33 no. 4 (August 1990). An adaptive algorithm for lossless compression of digital audio is described in Shamoan and Heegard, "A Rapidly Adaptive Lossless Compression Algorithm for High Fidelity Audio Coding", *Proc. IEEE Data Compression Conf.*, 430-39 (1994). A comparison of the H.261 and MPEG1 video compression standards is described in von Roden, "H.261 and MPEG1 - A Comparison", *Proc. IEEE 15th Ann. Int'l Conf. On Computers and Comm.*, 65-71 (1996).

[0009] The invention is defined by the subject matters of the independent claims.

[0010] Preferred embodiments are defined by the dependent claims.

[0011] The invention concerns using a variable-to-variable entropy encoder to code an arbitrary input stream. A variable-to-variable entropy encoder codes variable length input sequences with variable length codes. To limit code book size, entropy-type codes may be assigned to only probable inputs, and alternate codes used to identify less probable sequences.

[0012] To optimize searching the code book, it may be organized into sections that are searched separately. For example, one arrangement is to group all stored input sequences in the book according to the first symbol of the input sequence. A hash encoding function, collection of pointers, or other method may be used to immediately jump to a given section of the code book. Each section may further be sorted according to the probability associated with the entry. For example, each section may be sorted with highest probable inputs located first in the section, thus increasing the likelihood that a match will be found quickly.

[0013] Matching code book entries depends on the internal representation of the book. For example, in a tree structure, nodes may represent each character of the

input such that reaching a leaf signifies the end and identification of a particular grouping of input symbols. In a table structure, a pattern matching algorithm can be applied to each table entry within the appropriate section. Depending on the implementation of the table and matching algorithms, searching may be facilitated by recognition that only as many input symbols as the longest grouping in the code book section need to be considered. After finding a code book match, the corresponding entropy-type code can be output and the search repeated with the next symbol following the matched input.

[0014] The illustrated embodiments focus on encoding audio data, and the input stream is expected to be any data stream, such as numbers, characters, or a binary data, that encodes audio. For simplicity, the input stream is referenced herein as a series of symbols, where each "symbol" refers to the appropriate measurement unit for the particular input. The input stream may originate from local storage, or from Intranets, the Internet, or streaming data (e.g., Microsoft's "NETSHOW" client/server streaming architecture).

FIG. 1 is a block diagram of a computer system that may be used to implement variable to variable entropy encoding.

FIG. 2 shows a basic communication model for transmitting streaming and non-streaming data.

FIG. 3 is a flowchart showing creation of a code book having variable length entries for variable length symbol groupings.

FIGS. 4-10 illustrate creation of a code book pursuant to FIG. 3 for an alphabet {A, B, C}.

FIG. 11 shows encoding of audio data.

FIG. 12 illustrates an entropy encoder.

[0015] The invention has been implemented in an audio/visual codec (compressor/de-compressor). This is only one example of how the invention may be implemented. The invention is designed to be utilized wherever entropy-type coding may be utilized, and is applicable to compression of any type of data. Briefly described, optimal entropy encoding requires excessive resources, and the illustrated embodiments provide a nearly optimal encoding solution requiring far fewer resources.

Exemplary Operating Environment

[0016] FIG. 1 and the following discussion are intended to provide a brief, general description of a suitable computing environment in which the invention may be implemented. While the invention will be described in the general context of computer-executable instructions of a computer program that runs on a personal computer, those skilled in the art will recognize that the invention also may be implemented in combination with other program modules. Generally, program modules include

5 routines, programs, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Moreover, those skilled in the art will appreciate that the invention may be practiced with other computer system configurations, including hand-held devices, multiprocessor systems, microprocessor-based or programmable consumer electronics, mini-computers, mainframe computers, and the like. The illustrated embodiment of the invention also is practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. But, some embodiments of the invention can be practiced on stand alone computers. In a distributed computing environment, program modules may be located in both local and remote memory storage devices.

[0017] With reference to FIG. 1, an exemplary system for implementing the invention includes a computer 20, including a processing unit 21, a system memory 22, and a system bus 23 that couples various system components including the system memory to the processing unit 21. The processing unit may be any of various commercially available processors, including Intel x86, Pentium and compatible microprocessors from Intel and others, the Alpha processor by Digital, and the PowerPC from IBM and Motorola. Dual microprocessors and other multi-processor architectures also can be used as the processing unit 21.

[0018] The system bus may be any of several types of bus structure including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of conventional bus architectures such as PCI, AGP, VESA, Microchannel, ISA and EISA, to name a few. The system memory includes read only memory (ROM) 24 and random access memory (RAM) 25. A basic input/output system (BIOS), containing the basic routines that help to transfer information between elements within the computer 20, such as during start-up, is stored in ROM 24.

[0019] The computer 20 further includes a hard disk drive 27, a magnetic disk drive 28, e.g., to read from or write to a removable disk 29, and an optical disk drive 30, e.g., for reading a CD-ROM disk 31 or to read from or write to other optical media. The hard disk drive 27, magnetic disk drive 28, and optical disk drive 30 are connected to the system bus 23 by a hard disk drive interface 32, a magnetic disk drive interface 33, and an optical drive interface 34, respectively. The drives and their associated computer-readable media provide nonvolatile storage of data, data structures, computer-executable instructions, etc. for the computer 20. Although the description of computer-readable media above refers to a hard disk, a removable magnetic disk and a CD, it should be appreciated by those skilled in the art that other types of media which are readable by a computer, such as magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, and the like, may also be used in the exemplary operating environment.

[0020] A number of program modules may be stored in the drives and RAM 25, including an operating system 35, one or more application programs (e.g., Internet browser software) 36, other program modules 37, and program data 38.

[0021] A user may enter commands and information into the computer 20 through a keyboard 40 and pointing device, such as a mouse 42. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 21 through a serial port interface 46 that is coupled to the system bus, but may be connected by other interfaces, such as a parallel port, game port or a universal serial bus (USB). A monitor 47 or other type of display device is also connected to the system bus 23 via an interface, such as a video adapter 48. In addition to the monitor, personal computers typically include other peripheral output devices (not shown), such as speakers and printers.

[0022] The computer 20 is expected to operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 49. The remote computer 49 may be a web server, a router, a peer device or other common network node, and typically includes many or all of the elements described relative to the computer 20, although only a memory storage device 50 has been illustrated in FIG. 1. The computer 20 can contact the remote computer 49 over an Internet connection established through a Gateway 55 (e.g., a router, dedicated-line, or other network link), a modem 54 link, or by an intra-office local area network (LAN) 51 or wide area network (WAN) 52. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

[0023] In accordance with the practices of persons skilled in the art of computer programming, the present invention is described below with reference to acts and symbolic representations of operations that are performed by the computer 20, unless indicated otherwise. Such acts and operations are sometimes referred to as being computer-executed. It will be appreciated that the acts and symbolically represented operations include the manipulation by the processing unit 21 of electrical signals representing data bits which causes a resulting transformation or reduction of the electrical signal representation, and the maintenance of data bits at memory locations in the memory system (including the system memory 22, hard drive 27, floppy disks 29, and CD-ROM 31) to thereby reconfigure or otherwise alter the computer system's operation, as well as other processing of signals. The memory locations where data bits are maintained are physical locations that have particular electrical, magnetic, or optical properties corresponding to the data bits.

[0024] FIG. 2 shows a basic communication model.

In a basic communication model, there is a data source or sender 200, a communication channel 204, and a data receiver 208. The source may be someone speaking on a telephone, over telephone wires, to another person. Or, the source may be a television or radio broadcast transmitted by wireless methods to television or radio receivers. Or, the source may be a digital encoding of audio data, transmitted over a wired or wireless communication link (e.g., a LAN or the Internet) to a corresponding decoder for the information.

[0025] To facilitate transmission and receipt of the data, an encoder 202 is used to prepare the data source for transmission over the communication channel 204. The encoder is responsible for converting the source data into a format appropriate for the channel 204. For example, in the context of a common telephone call, one's voice is typically converted by the phone's handset from voice sounds to analog impulses that are sent as analog data to local telephone receiving equipment. This analog signal is then converted into digital form, multiplexed with numerous other conversations similarly encoded, and transmitted over a common line towards the receiver. Thus, in FIG. 2, the channel 204 corresponds in large part to a common pathway shared by multiple senders and receivers. For network applications, the channel 204 is commonly an intranet or the Internet. At the receiving end 208, a decoder 206 is required to reverse the encoding process so as to present sensible data to the receiver.

[0026] This simple model does not take into account, however, the real-world demands of application programs. For example, a client (e.g., an application program) commonly wants to process, display or play received data in real-time as it is retrieved over a network link. To do so a streaming delivery system is required, i.e., an adaptive data transmission system that allows application-level bandwidth reservation for a data stream. Streaming environments contrast with traditional networking programs, such as certain versions of Internet browsers that download web page content on a non-prioritized basis, and allow data content delivery over the network link 204 to be orchestrated (and optimized) for particular retrieval needs (such as a slow dial-up link).

[0027] An exemplary streaming format (SF) is the Microsoft Active Streaming Format. Generally, a SF defines the structure of complex, synchronized object data streams. Any object can be placed into a SF data stream, including audio and video data objects, scripts, ActiveX controls, and HTML documents. SF data can be stored in a file or created in real-time using audio and video capture hardware. An Application Programming interface (API) corresponding to an implementation of the SF can provide an application with support for delivering and receiving streaming content. One such API is the Microsoft Audio Compression Manager (ACM), which provides functions for processing (e.g., compressing and delivering) audio data. Other networking APIs that can be used to support the SF include the Mi-

icrosoft Win32 internet Extensions (Wininet), WinSock, and TCP/IP APIs. (For more information see the 1998 Visual Studio 6.0 MSDN Library.) Note that it is intended that processed data can be stored for later retrieval by a client, and that such retrieval can be performed in a non-streaming format (e.g., by a small playback appli-

[0028] To transmit streaming or non-streaming data, networks such as the Internet convert the source data into packet form suitable for the network. Packets generally include routing information as well as the actual data. SF data streams are preferably made up of data packets that can be transmitted over any digital network by inserting them one at a time into the data field of network packets. Each SF packet may contain a prioritized mix of data from different objects within the stream, so that the bandwidth can be concentrated on higher priority objects (or organized to optimize throughput). This data can be captured in real time, stored to nonvolatile storage, converted from existing audio or video formats, created by combining audio with pictures and scripts, or delivered over the network to a client program or viewer. The client receiver 208 of the streaming data can be a traditional "helper" application (for compatibility with the old Web publishing approach), or a more modern web page control (e.g., an ActiveX object) embedded in a web page.

[0029] SF data streams are distinguished over traditional network content as being viewed progressively in real time as a client receives it. Unfortunately, playback of streamed content becomes susceptible to transmission delays. If data does not arrive reliably, or if transmission speed falls below an acceptable minimum, playback of the content cannot continue at an acceptable rate. Smooth-streaming playback at a client requires that the transmission require a bandwidth less than the client's available bandwidth (e.g. the speed of the link 204 less networking overhead). Typically a dial-up connection to the Internet provides a bandwidth of 28-34 Kbps. Consequently, audiovisual source data (which is bandwidth intensive) must be significantly compressed to allow its transmission over low bit-rate connections. The degree of compression necessarily impacts the quality of the reproduced signal. Preferably a server provides multiple sources optimized for different networking speeds, or utilizes an adaptive feedback system to perform real-time analysis of the client's actual throughput.

[0030] Once SF data packets are encoded 202 and placed inside network packets and sent over the network 204, the routing technology of the network takes care of delivering the network packets to the receiver 208. Preferably a variety of network and application protocols, such as UDP, TCP, RTP, IP Multicast, IPX, and HTTP, are supported by the broadcast sender 200.

[0031] As discussed above, bandwidth is limited and the encoder 202 generally must compress data prior to transmission. A particularly effective method for encod-

ing source data frequency coefficients to ensure reliable transmission over a communication channel is entropy encoding. Entropy coding capitalizes on data coherency, and is effective when symbols have non-uniform probability distribution.

[0032] FIG. 3 is a flowchart showing a preferred method for generating an entropy encoder's code book. In particular, in contrast with prior art techniques, FIG. 3 illustrates how to create a code book having variable length code assignments for variable length symbol groupings. As discussed above, prior art techniques either require fixed-length codes or fixed blocks of input. Preferred implementations overcome the resource requirements of large dimension vector encoding, and the inapplicability of coding into words of equal lengths, by providing an entropy based variable-to-variable code, where variable length code words are used to encode variable length X sequences.

[0033] Let y_i represent each source symbol group $\{x_j\}$, for $1 \leq j \leq N_i$, having probability P_i of occurring within the input stream (FIG. 2 channel 204), and that each group is assigned a corresponding code word having L_i bits. It is presumed that each x_j is drawn from a fixed alphabet of predetermined size. The objective is to minimize the

$$\text{equation } L = \sum_i \frac{L_i * P_i}{N_i} .$$

[0034] Instead of finding a general solution to the problem, the problem is separated into two different tasks. The first task is identification of a (sub-optimal) grouping of a set of input symbols $\{x_i\}$ through an empirical approach described below. The second task is assigning an entropy-type code for the grouped symbols $\{y_i\}$. Note that it is known that if the source is not coherent (i.e., the input is independent or without memory), any grouping that has the same configuration of $\{N_j\}$ can achieve the same coding efficiency. In this situation, the first task becomes inconsequential.

[0035] To perform the first task, an initial trivial symbol grouping 300 is prepared, such as $\{y_i\} = \{x_i\}$. This initial configuration assumes that an exemplary input stream is being used to train creation of the code book. It is understood that a computer may be programmed with software constructions such as data structures to track receipt of each symbol from an input. Such data structures may be implemented as a binary-type tree structure, hash table, or some combination of the two. Other equivalent structures may also be used.

[0036] After determining the trivial grouping, the probability of occurrence for each y_i is computed 302. Such probability is determined with respect to any exemplary input used to train code book generation. As further symbols are added to the symbol data structure, the probabilities are dynamically adjusted.

[0037] Next, the most probable grouping y_i is identified 304 (denoted as y_{mp}). If 306 the highest probability

symbol is a grouping of previously lower probability symbols, then the grouping is split 308 into its constituent symbols, and processing restarted from step 302. (Although symbols may be combined, the group retains memory of all symbols therein so that symbols can be extracted.)

[0038] If the symbol is not a grouping, then processing continues with step 310, in which the most probable grouping is then tentatively extended 310 with single symbol extensions xi's. Preferably ymp is extended with each symbol from the X alphabet used. However, a predictor can be used to only generate an extension set containing only probable extensions, if the alphabet is very large and it is known many extensions are unlikely. For example, such a predictor may be based on semantic or contextual meaning, so that very improbable extensions can be ignored a priori.

[0039] The probability for each tentative expansion of ymp is then computed 312, and only the most probable extension retained 314. The rest of the lower probability extensions are collapsed together 316 as a combined grouping and stored in code book with a special symbol to indicate a combined grouping. This wild-card symbol represents any arbitrary symbol grouping having ymp as a prefix, but with an extension (suffix) different from the most probable extension. That is, if ymp + xmp is the most probable root and extension, then the other less probable extensions are represented as ymp*, * ≠ xmp. (Note that this discussion presumes, for clarity, serial processing of single-symbol extensions; however, parallel execution of multiple symbol extensions is contemplated.) It is understood by one skilled in the art that applying single symbol extensions, and keeping only one most probable grouping, are restrictions imposed for clarity of discussion. It is further understood that although discussion focuses on serial processing, code book construction may be paralleled.

[0040] Code book construction is completed by repeating 318 steps 302-316 until all extensions have been made, or the number of the code book entries reaches a predetermined limit. That is, repeating computing probabilities for each current yi 302, where the code book set {Y} now includes ymp + xmp, and respectively choosing 304 and grouping the most and least likely extensions. The effect of repeatedly applying the above operations is to automatically collect symbol groupings having high correlation, so that inter-group correlation is minimized. This minimizes the numerator of L, while simultaneously maximizing the length of the most probable yi so that the denominator of L is maximized.

[0041] FIGS. 4-10 illustrate creation of a code book pursuant to FIG. 3 for an alphabet {A, B, C}. For this discussion, the code book is defined with respect to an exemplary input stream "A A A B B A A C A B A B B A B". As discussed above, one or more exemplary inputs may be used to generate a code book that is then used by encoders and decoders to process arbitrary inputs.

For clarity, the code book is presented as a tree structure, although it may in fact be implemented as a linear table, hash table, database, etc. As illustrated, the tree is oriented left-to-right, where the left column (e.g., "A" and "X0") represents the top row of a tree-type structure, and successively indented rows represent the "children" of the previous row's node (e.g., in a top-down tree for FIG. 5, node "A" is a first-row parent node for a second-row middle-child- node "B").

[0042] In preparing the code book, the general rule is to pick the most probable leaf node, expand it, re-compute probabilities to determine the most probable leaf-node, and then compact the remaining sibling nodes into a single Xn node (n=0..N, tracking each time nodes have been combined). If it turns out that the most probable node is a group node, then the group is split, probabilities recalculated, and the most probable member node retained (i.e., the remaining group members are re-grouped). Processing cycles until a stop state is reached, such as a code book having predetermined size.

[0043] FIG. 4 shows an initial grouping for the input stream "A A A B B A A C A B A B B A B". An initial parsing of the input shows probabilities of occurrence of A = 8/15, B = 6/15, and C=1/15. This initial trivial grouping can be created based on different criteria, the simplest being having a first-level node for every character in the alphabet. However, if the input alphabet is large, the trivial grouping may be limited to some subset of symbols having highest probability, where the remaining symbols are combined into an X grouping. FIG. 4 illustrates this technique by starting with only two initial groups, group A 400 having probability 8/15, and group X0 402 having probability 7/15, where X0 represents all remaining low probability symbols in the alphabet, e.g., B and C.

[0044] After preparing an initial trivial grouping, the leaf-node having highest probability is selected for extension (see also FIG. 3 discussion regarding processing sequence). Hence, as shown in FIG. 5, group A 400 is tentatively expanded by each character in the alphabet (or one may limit the expansion to some subset thereof as described for creating the initial grouping). Probabilities are then recomputed with respect to the input stream "A A A B B A A C A B A B B A B" to determine values for the tentative extensions A 406, B 408, and C 410. The result is nine parsing groups, where "A A" appears 2/9, "A B" appears 4/9, and "A C" appears 0/9. Therefore, the most probable extension "a b" is retained and the other extensions collapsed into X1 =A,C. Note that although this discussion repeatedly recalculates all probabilities, a more efficient approach is to retain probabilities and symbol associations for each node within the node, and only computing information as necessary.

[0045] FIG. 6 shows the collapse into X1 for FIG. 5. Processing repeats with identification of the node having highest probability, e.g., node B 408 at probability 4/9.

[0046] As shown in FIG. 7, this node 408 is tentatively

extended with symbols A 414, B 416, C 418, and as discussed above, the tentative grouping with highest probability is retained. After recalculating probabilities, the result is eight parsing groups in which the symbol sequence "A B A" 414 appears once, "A B B" 416 appears once, and "A B C" 418 does not appear at all. Since tentative extensions A 414 and B 416 have the same probability of occurrence, a rule needs to be defined to choose which symbol to retain. For this discussion, whenever there are equal probabilities, the highest row node (e.g., the left-most child node in a top-down tree) is retained. Similarly, when there is a conflict between tree rows, the left-most row's node (e.g., the node closest to the root of a top-down tree) is retained.

[0047] Note that the above described parsing of the exemplary input does not account for the trailing two symbols "A B" of the input. As illustrated in FIG. 7, there is no leaf corresponding to "A B" as that configuration was expanded into "A B A", "A B B", and "A B C". To compensate, code book entries can be created to account for such end of input sequences, or the input having no entry can be escaped with a special character and inserted in the encoded output stream. For example, a special symbol can be used to indicate end of input, therefore implying how to handle the trailing characters on decoding.

[0048] Therefore, as shown in FIG. 8, node A 414 is retained and nodes B 416 and C 418 are combined into node $X_2 = B, C$ 420, having combined probability of $1/8 + 0/8$. Now, the next step is to expand the node currently having highest probability with respect to the input stream. As shown, nodes $X_1 = A, C$ 412 and $X_0 = B, C$ 402 have the same probability of occurrence ($3/8$). As discussed above, the highest node in the tree (X_0 402) is extended. (Although it is only necessary to be consistent, it is preferable to expand higher level nodes since this may increase coding efficiency by increasing the number of long code words.)

[0049] However, X_0 402 is a combined node, so it must be split instead of extended. FIG. 9 illustrates the result of splitting node X_0 into its constituent symbols B 422 and C 424. Recalculating probabilities indicates that symbol sequences "A B A" appears $1/8$, "A B X_2 " appears $1/8$, "A X_1 " appears $3/8$, "B" 422 appears $2/8$, and "C" appears $1/8$. Since this is a split operation, the split node having highest probability, e.g., node B 422, is retained, and the remaining node(s) re-combined back into $X_0 = C$ 424.

[0050] FIG. 10 shows the result of retaining high-probability node B 422. Note that grouping X_0 now only represents a single symbol "C". After revising probabilities, the node having highest probability must be identified and split or extended. As shown, symbol sequence "A B A" appears $1/8$, "A B X_2 " appears $1/8$, "A X_1 " appears $3/8$, "B" appears $2/8$, and " X_0 " appears $1/8$. Therefore node X_1 412, as a combined node, must be split.

[0051] Splitting proceeds as discussed above, and processing the code book cycles as illustrated in FIG.

3, with highest probability nodes being extended or split until a stop state is reached (e.g., the code book reaches a maximum size). Once the code book has reached a stop state, it is available for encoding data to transmit over a communication channel. Note that for the FIG. 10 configuration, the average bits per input symbol, assuming fractional bits under "ideal" scalar Huffman encoding, is approximately 0.8 bits/symbol (varies depending on how the trailing input "A B" is handled). This represents a significant (about 10%) savings over previous lossless compression techniques.

[0052] FIG. 11 shows a transmission model for transmitting audio data over a channel 460. It is presumed that the channel 460 is of limited bandwidth, and therefore some compression of source data 450 is required before the data can be reliably sent. Note that although this discussion focuses on transmission of audio data, the invention applies to transfer of other data, such as audio visual information having embedded audio data (e.g., multiplexed within an MPEG data stream), or other data sources having compressible data patterns (e.g., coherent data).

[0053] As illustrated, source data 450 is input to a time / frequency transform encoder 452 such as a filter bank or discrete-cosine type transform. Transform encoder 452 is designed so as to convert a continuous or sampled time-domain input, such as an audio data source, into multiple frequency bands of predetermined (although perhaps differing) bandwidth. These bands can then be analyzed with respect to a human auditory perception model 454 (for example, a psychoacoustic model) in order to determine components of the signal that may be safely reduced without audible impact. For example, it is well known that certain frequencies are inaudible when certain other sounds or frequencies are present in the input signal (simultaneous masking). Consequently, such inaudible signals can be safely removed from the input signal. Use of human auditory models is well known, e.g., the MPEG 1, 2 and 4 standards. (Note that such models may be combined into a quantization 456 operation.)

[0054] After performing the time/frequency transformation, frequency coefficients within each range are quantized 456 to convert each coefficient (amplitude levels) to a value taken from a finite set of possible values, where each value has a size based on the bits allocated to representing the frequency range. The quantizer may be a conventional uniform or non-uniform quantizer, such as a midriser or midtreader quantizer with (or without) memory. The general quantization goal is identifying an optimum bit allocation for representing the input signal data, i.e., to distribute usage of available encoding bits to ensure encoding the (acoustically) significant portions of the source data. Various quantization methods, such as quantization step size prediction to meet a desired bit rate (assuming constant bit rate) can be used. After the source 450 has been quantized 456, the resultant data is then entropy encoded 458 accord-

ing to the code book of FIG. 3.

[0055] FIG. 12 shows one method for implementing the entropy encoder 458 of FIG. 11 through application of the code book of FIG. 3 to the quantized data. The code book for variable-to-variable encoding can be used to encode other types of data. As illustrated, the quantized data is received 480 as input to the entropy encoder 458 of FIG. 11. It is understood that the input is in some form of discrete signals or data packets, and that for simplicity of discussion, all input is simply assumed to be a long series of discrete symbols. The received input 480 is scanned 482 in order to locate 484 a corresponding code book key in the code book of FIG. 3. Such scanning corresponds to a data look-up, and depending on the data structure used to implement the code book, the exact method of look-up will vary.

[0056] There are various techniques available for storing and manipulating the code book. One structure for a code book is traversal and storage of a N-ary (e.g., binary, tertiary, etc.) tree, where symbol groupings guide a traversal of the tree structure. The path to a leaf node of the tree represents the end of a recognized symbol sequence, where an entropy-type code is associated with the sequence. (Note that the code book may be implemented as a table, where a table entry contains the entire input sequence, e.g., the path to the node.) Nodes can be coded in software as a structure, class definition, or other structure allowing storage of a symbol or symbols associated with the node, and association of a corresponding entropy-type code 486.

[0057] Alternatively, the code book may be structured as a table having each string of input symbol sorted by probability of occurrence, with highly probable input at the top of the table. For large tables, the table can be sorted according to the first symbol, i.e., all symbol series beginning with "A" are grouped together, followed by series starting with "B", etc. With this arrangement, all entries within the grouping are sorted according to their probabilities of occurrence. The position of the beginning of each section is marked/tracked so that a hash-type function (e.g., a look-up based on the first symbol) can be used to locate the correct portion of the code book table. In this look-up table approach to storing the code book, once the first symbol is hashed, then the corresponding table section is exhaustively searched until a matching entry is located. The code 484 associated with the matching entry is then output 486 as the encoded substitute.

[0058] Continuing now with FIG. 11, once the output 486 is known, this output is transmitted over the communication channel 460. The receiving end 470 then implements a reverse-encoding process, i.e., a series of steps to undo the encoding of the source data 450. That is, the encoded data 486 is received as input to an entropy decoder 462 which performs a reverse code book look-up to convert the encoded output 486 back into the original input symbol series 480 (FIG. 12). The recovered input data 480 is then processed by a de-quantizer

464 and time/frequency transform decoder 466 to reverse the original coding operations, resulting in a reconstructed data 468 that is similar to the original source data 450. It should be noted that the reconstructed data 468 only approximates the original source data 450 when, as it presumed herein, a lossy system is employed.

[0059] Having described and illustrated the principles of my invention with reference to an illustrated embodiment, it will be recognized that the illustrated embodiment can be modified in arrangement and detail without departing from such principles.

15 Claims

1. A method of encoding a sequence of digital audio data symbols (450) for transmission over a communications channel (460), the method comprising identifying a first variable size grouping of audio data symbols within the sequence of digital audio data symbols, wherein the method is **characterized by:**

coding (486) the first variable size grouping of symbols with a coding arrangement which produces as output an entropy code word corresponding to the first grouping of symbols, wherein the coding utilizes a pre-constructed code book that associates variable size groupings of audio data symbols with corresponding entropy code words for variable-to-variable compression; and

repeatedly identifying and coding (486) subsequent variable size groupings of symbols such that at least two identified groupings of symbols within the sequence of digital audio data symbols have differing sizes.

2. The method of claim 1 in which the code book is organized into sections according to a first symbol of each grouping in the code book, where each section is further sorted by probability of occurrence of each entry within each section, and wherein identifying a variable size grouping comprises:

identifying a section by a first symbol of the variable size grouping; and
matching the variable size grouping against each section entry until a match is found.

3. The method of claim 2 wherein the match has a size, such size indicating a new position in the sequence of digital audio data symbols for identifying a second variable size grouping of symbols.
4. The method of claim 3 wherein the first and the second variable size grouping have differing sizes.

5. The method of claim 1, wherein the code book is organized as a table having sections, such sections defined by a first symbol of each variable size grouping of symbols, and wherein table entries within a section are sorted according to probability of occurrence of each variable size grouping of symbols within the section.
6. The method of any of claims 1 to 5 wherein an input channel is in communication with a disk storage, the method further comprising the step of reading the sequence of digital data symbols from the disk storage, so as to allow identification and coding of the variable size groupings of symbols.
7. The method of any of claims 1 to 5 further comprising:
- receiving the sequence of digital audio data symbols from an input channel; and
after the coding (486) of a variable size grouping of symbols, transmitting the entropy code word corresponding to the grouping of symbols over the communications channel (460).
8. The method of claim 7 wherein the sequence of digital audio data symbols is received in real-time, and the transmitting is performed in real-time.
9. The method of claim 7 further comprising receiving a connection request from a client over a client network connection, wherein the transmitting is performed over the client network connection.
10. The method of claim 9 wherein the sequence of digital audio data symbols is received and stored in non-volatile storage, and the transmitting is delayed until receipt of the connection request from the client.
11. A method for decoding a compressed audio data stream from a communications channel (460), comprising:
- receiving an entropy code word;
- looking up the entropy code word in a pre-constructed code book containing a correspondence between entropy code words and variable size series of symbols for variable-to-variable decompression; and
- outputting a variable size series of audio data symbols corresponding to the code word in the code book.
12. The method of claim 11 wherein the step of looking up the entropy code word includes hashing the entropy code word to obtain an index to an entry in a hash table.
13. The method of claim 12 wherein the code book is organized into sections according to a first symbol of each variable size series of symbols stored within the code book.
14. A computer readable medium having stored therein computer-executable instructions for causing a computer to perform the method of any of claims 1 to 13.
15. A system for transmitting a compressed audiovisual data stream from a server network service to a client network service over a network link (460), such compressed audiovisual data stream formed by replacing a variable size sequence of audiovisual data input symbols with an output entropy code, the system comprising:
- an input buffer for storing a series of uncompressed audiovisual data symbols to compress and transmit to the client;
- an output memory for storing an entropy code word representing a compressed version of the series of uncompressed symbols in the input buffer;
- wherein the system is **characterized by**:
- a code book memory for storing a pre-existing code book containing an entropy code word for a variable size series of symbols, wherein the code book associates variable size series of symbols with corresponding entropy code words for variable-to-variable compression;
- a searching arrangement for looking up an entropy code word for a particular series of symbols in the code book; and
- an encoding arrangement having an input channel in communication with the input buffer and an output in communication with the output memory;
- wherein the encoding arrangement applies the searching arrangement to look up the entropy code word for the series of uncompressed symbols for storage in the output memory.
16. The system of claim 15 further comprising transmission means for transmitting the contents of the output memory to the client over the network link (460).
17. The system of claim 16 wherein a streaming net-

working protocol is utilized to communicate over the network link (460).

18. A system for decoding compressed audiovisual data received from a network link (460), the system comprising:

an input memory for storing a code word;
an output buffer for storing a series of uncompressed audiovisual data symbols corresponding to the code word in the input memory;

wherein the system is **characterized by**:

a code book memory for storing a pre-existing code book containing a variable size series of symbols for an entropy code word, wherein the code book associates entropy code words with corresponding variable size series of symbols for variable-to-variable decompression;

a searching arrangement for looking up an entropy code word for a particular series of symbols in the code book; and

a decoding arrangement having an input channel in communication with the input memory and an output in communication with the output buffer,

wherein the decoding arrangement applies the searching arrangement to look up the series of uncompressed symbols corresponding to the entropy code word, and store such series in the output buffer.

19. The system of claim 18 wherein a streaming networking protocol is utilized to communicate over the network link (460).

20. The system of claim 18 in which the decoding arrangement is implemented by an application program that also implements the Hyper-Text Markup Language protocol.

Patentansprüche

1. Verfahren zur Codierung einer Folge von digitalen Audiodatensymbolen (450) zum Übertragen über einen Kommunikationskanal (460), wobei das Verfahren das Identifizieren einer ersten Gruppierung variabler Größe von Audiodatensymbolen in der Folge von digitalen Audiodatensymbolen umfasst, wobei das Verfahren **gekennzeichnet ist durch**:

Codieren (486) der ersten Gruppierung variabler Größe von Symbolen mit einer Codieran-

ordnung, die als Ausgang ein Entropie-Codewort erzeugt, das der ersten Gruppierung von Symbolen entspricht, wobei das Codieren ein vorher konstruiertes Codebuch verwendet, das Gruppierungen variabler Größe von Audiodatensymbolen mit entsprechenden Entropie-Codewörtern zur variabel-zu-variabel-Kompression verbindet, und

wiederholt Identifizieren und Codieren (486) von nachfolgenden Gruppierungen variabler Größe von Symbolen, sodass wenigstens zwei identifizierte Gruppierungen von Symbolen in der Folge von digitalen Audiodatensymbolen unterschiedliche Größen aufweisen.

2. Verfahren nach Anspruch 1, wobei das Codebuch entsprechend einem ersten Symbol jeder Gruppierung in dem Codebuch in Abschnitte gegliedert ist, wobei jeder Abschnitt weiter nach der Wahrscheinlichkeit des Vorkommens jedes Eintrags innerhalb jedes Abschnitts sortiert ist, und wobei das Identifizieren einer Gruppierung variabler Größe umfasst Identifizieren eines Abschnitts durch ein erstes Symbol der Gruppierung variabler Größe, und Abgleichen der Gruppierung variabler Größe gegen jeden Abschnittseintrag, bis ein Gegenstück gefunden ist.

3. Verfahren nach Anspruch 2, wobei das Gegenstück eine Größe hat, wobei eine solche Größe eine neue Position in der Folge von digitalen Audiodatensymbolen zum Identifizieren einer zweiten Gruppierung variabler Größe von Symbolen angibt

4. Verfahren nach Anspruch 3, wobei die erste und die zweite Gruppierung variabler Größe unterschiedliche Größen aufweisen.

5. Verfahren nach Anspruch 1, wobei das Codebuch als eine Tabelle mit Abschnitten aufgebaut ist, wobei solche Abschnitte durch ein erstes Symbol jeder Gruppierung variabler Größe von Symbolen definiert sind, und

wobei Tabelleneinträge innerhalb eines Abschnitts entsprechend der Wahrscheinlichkeit des Vorkommens jeder Gruppierung variabler Größe von Symbolen in dem Abschnitt sortiert sind.

6. Verfahren nach einem der Ansprüche 1 bis 5, wobei ein Eingangskanal mit einem Plattenspeicher in Verbindung steht, wobei das Verfahren den Schritt des Lesens der Folge von digitalen Datensymbolen aus dem Plattenspeicher umfasst, um die Identifikation und Codierung der Gruppierungen variabler Größe zu gestatten.

7. Verfahren nach einem der Ansprüche 1 bis 5, das

- weiter umfasst
Empfangen der Folge von digitalen Audiodatensymbolen von einem Eingangskanal, und nach dem Codieren (486) einer Gruppierung variabler Größe von Symbolen, Übertragen des der Gruppierung von Symbolen entsprechenden Entropie-Codewortes über den Kommunikationskanal (460).
8. Verfahren nach Anspruch 7, wobei die Folge von digitalen Audiodatensymbolen in Echtzeit empfangen wird, und das Übertragen in Echtzeit durchgeführt wird.
9. Verfahren nach Anspruch 7, das weiter das Empfangen einer Verbindungsanforderung von einem Client über eine Client-Netzwerkverbindung umfasst, wobei das Übertragen über die Client Netzwerkverbindung durchgeführt wird.
10. Verfahren nach Anspruch 9, wobei die Folge von digitalen Audiodatensymbolen empfangen und in einem nicht flüchtigen Speicher gespeichert wird, und das Übertragen bis zum Empfang der Verbindungsanforderung von dem Client aufgeschoben wird.
11. Verfahren zur Decodierung eines komprimierten Audiodatenstromes von einem Kommunikationskanal (460), das umfasst:
- Empfangen eines Entropie-Codewortes;
- Nachsehen des Entropie-Codewortes in einem vorher konstruierten Codebuch, das eine Entsprechung zwischen Entropie-Codewörtern und Serien variabler Größe von Symbolen zur variabel-zu-variabel-Dekompression enthält, und
- Ausgeben einer dem Codewort in dem Codebuch entsprechenden Serie variabler Größe von Audiodatensymbolen.
12. Verfahren nach Anspruch 11, wobei der Schritt des Nachsehens des Entropie-Codewortes "Hashing" des Entropie-Codewortes einschließt, um einen Index zu einem Eintrag in einer Hash-Tabelle zu erlangen.
13. Verfahren nach Anspruch 12, wobei das Codebuch entsprechend einem ersten Symbol jeder in dem Codebuch gespeicherten Serie variabler Größe von Symbolen in Abschnitte gegliedert ist
14. Computerlesbares Medium, in dem computerausführbare Anweisungen gespeichert sind, die einen Computer veranlassen, das Verfahren nach einem
- der Ansprüche 1 bis 13 durchzuführen.
15. System zum Übertragen eines komprimierten audiovisuellen Datenstromes von einem Server-Netzwerkservice an einen Client-Netzwerkservice über eine Netzwerkverbindung (460), wobei ein solcher komprimierter audiovisueller Datenstrom durch Ersetzen einer Folge variabler Größe von audiovisuellen Dateneingangssymbolen durch einen Ausgabe-Entropiecode gebildet wird, wobei das System umfasst:
- einen Eingangspuffer zur Speicherung einer zu komprimierenden und an den Client zu übertragenden Serie von unkomprimierten audiovisuellen Datensymbolen;
- einen Ausgangsspeicher zur Speicherung eines Entropie-Codewortes, das eine komprimierte Version der Serie von unkomprimierten Symbolen in dem Eingangspuffer darstellt
- wobei das System **gekennzeichnet ist durch:**
- einen Codebuchspeicher zur Speicherung eines vorher vorhandenen Codebuches, das ein Entropie-Codewort für eine Serie variabler Größe von Symbolen enthält wobei das Codebuch Serien variabler Größe von Symbolen mit entsprechenden Entropie-Codewörtern zur variabel-zu-variabel-Kompression verbindet;
- eine Suchanordnung zum Nachsehen eines Entropie-Codewortes für eine bestimmte Serie von Symbolen in dem Codebuch, und
- eine Codieranordnung mit einem Eingangskanal, der mit dem Eingangspuffer in Verbindung steht, und einem Ausgang, der mit dem Ausgangsspeicher in Verbindung steht,
- wobei die Codieranordnung die Suchanordnung anwendet, um das Entropie-Codewort für die Serie von unkomprimierten Symbolen zur Speicherung in dem Ausgangsspeicher nachzusehen.
16. System nach Anspruch 15, das weiter eine Übertragungseinrichtung zum Übertragen des Inhalts des Ausgangsspeichers an den Client über die Netzwerkverbindung (460) umfasst.
17. System nach Anspruch 16, wobei ein Streaming-Netzbetriebsprotokoll benutzt wird, um über die Netzwerkverbindung (460) zu kommunizieren.
18. System zur Decodierung von komprimierten audiovisuellen Daten, die von einer Netzwerkverbindung (460) empfangen werden, wobei das System um-

fasst:

einen Eingangsspeicher zur Speicherung eines Codewortes;

5

einen Ausgangspuffer zur Speicherung einer dem Codewort in dem Eingangsspeicher entsprechenden Serie von unkomprimierten audiovisuellen Datensymbolen,

10

wobei das System **gekennzeichnet ist durch:**

einen Codebuchspeicher zur Speicherung eines vorher vorhandenen Codebuches, das eine Serie variabler Größe von Symbolen für ein Entropie-Codewort enthält, wobei das Codebuch Entropie-Codewörter mit entsprechenden Serien variabler Größe von Symbolen zur variabel-zu-variabel-Dekompression verbindet;

15

eine Suchanordnung zum Nachsehen eines Entropie-Codewortes für eine bestimmte Serie von Symbolen in dem Codebuch, und

20

eine Decodieranordnung mit einem Eingangskanal, der mit dem Eingangsspeicher in Verbindung steht, und einem Ausgang, der mit dem Ausgangspuffer in Verbindung steht,

25

wobei die Decodieranordnung die Suchanordnung anwendet, um die dem Entropie-Codewort entsprechende Serie von unkomprimierten Symbolen nachzusehen und eine solche Serie in dem Ausgangspuffer zu speichern.

30

19. System nach Anspruch 18, wobei ein Streaming-Netzbetriebsprotokoll benutzt wird, um Ober die Netzwerkverbindung (460) zu kommunizieren.

35

20. System nach Anspruch 18, in dem die Decodieranordnung durch ein Anwendungsprogramm implementiert ist, das auch das "Hyper-Text Markup Language"-Protokoll implementiert.

40

Revendications

1. Procédé de codage d'une séquence de symboles de données audio numériques (450) pour transmission sur une voie de communication (460), le procédé comprenant d'identifier un premier groupement de taille variable de symboles de données audio à l'intérieur de la séquence de symboles de données audio numériques, dans lequel le procédé est **caractérisé par** les étapes consistant à :

45

50

coder (486) le premier groupement de taille variable de symboles avec un arrangement de co-

dage qui produit en sortie un mot de code entropique correspondant au premier groupement de symboles, dans lequel le codage utilise un livre de codes préconstruit qui associe des groupements de taille variable de symboles de données audio à des mots de code entropique pour une compression variable à variable ; et

identifier et coder de manière répétitive (486) des groupements suivants de taille variable de symboles, de telle manière qu'au moins deux groupements identifiés de symboles dans la séquence de symboles de données audio numériques aient des tailles différentes.

2. Procédé selon la revendication 1, dans lequel le livre de codes est organisé en sections selon un premier symbole de chaque groupement dans le livre de codes, dans lequel chaque section est en outre triée par probabilité d'occurrence de chaque entrée à l'intérieur de chaque section et dans lequel l'identification d'un groupement de taille variable comprend les étapes consistant à :

identifier une section par un premier symbole du groupement de taille variable ; et comparer le groupement de taille variable à chaque entrée de section jusqu'à ce qu'une correspondance soit trouvée.

3. Procédé selon la revendication 2, dans lequel la correspondance a une taille, une telle taille indiquant une nouvelle position dans la séquence de symboles de données audio numériques pour identifier un second groupement de taille variable de symboles.

4. Procédé selon la revendication 3, dans lequel le premier et le second groupements de taille variable ont des tailles différentes.

5. Procédé selon la revendication 1, dans lequel le livre de codes est organisé sous la forme d'une table ayant des sections, ces sections étant définies par un premier symbole de chaque groupement de taille variable de symboles, et

dans lequel les entrées de la table à l'intérieur d'une section sont triées selon une probabilité d'occurrence de chaque groupement de taille variable de symboles à l'intérieur de la section.

6. Procédé selon l'une quelconque des revendications 1 à 5, dans lequel une voie d'entrée est en communication avec une mémoire à disque, le procédé comprenant en outre l'étape consistant à lire la séquence de symboles de données numériques à partir de la mémoire à disque de manière à permettre l'identification et le codage des groupements de

- taille variable de symboles.
7. Procédé selon l'une quelconque des revendications 1 à 5, comprenant en outre les étapes consistant à :
- 5 recevoir la séquence de symboles de données audio numériques a partir d'une voie d'entrée ;
et
après le codage (486) d'un groupement de
taille variable de symboles, transmettre le mot
de code entropique correspondant au groupe-
ment de symboles sur la voie de communica-
tion (460). 10
8. Procédé selon la revendication 7, dans lequel la sé-
quence de symboles de données audio numériques
est reçue en temps réel et dans lequel la transmis-
sion est effectuée en temps réel. 15
9. Procédé selon la revendication 7, comprenant en
outre de recevoir une demande de connexion de-
puis un client sur une connexion de réseau client,
dans lequel la transmission est effectuée sur la con-
nexion de réseau client. 20
10. Procédé selon la revendication 9, dans lequel la sé-
quence de symboles de données audio numériques
est reçue et stockée dans une mémoire non volatile
et dans lequel la transmission est retardée jusqu'à
la réception de la demande de connexion depuis le
client. 30
11. Procédé de décodage d'un train de données audio
compressées provenant d'une voie de communica-
tion (460), comprenant les étapes consistant à :
- 35 recevoir un mot de code entropique ;
rechercher le mot de code entropique dans un
livre de codes préconstruit contenant une cor-
respondance entre des mots de code entropi-
ques et des séries de taille variable de symbo-
les pour une décompression variable à
variable ; et
émettre une série de taille variable de symboles
de données audio correspondant au mot de co-
de dans le livre de codes. 40 45
12. Procédé selon la revendication 11, dans lequel
l'étape consistant à rechercher le mot de code en-
tropique comprend de hacher le mot de code entropi-
que pour obtenir un index vers une entrée dans
une table de hachage. 50
13. Procédé selon la revendication 12, dans lequel le
livre de codes est organisé en sections selon un
premier symbole de chaque série de taille variable
de symboles stocké dans le livre de codes. 55
14. Support exploitable par un ordinateur sur lequel
sont stockées des instructions exécutables par un
ordinateur pour entraîner qu'un ordinateur exécute
le procédé de l'une quelconque des revendications
1 à 13.
15. Système de transmission d'un train de données
audiovisuelles compressées depuis un service de
réseau serveur vers un service de réseau client sur
une liaison de réseau (460), un tel train de données
audiovisuelles compressées étant formé en rem-
plaçant une séquence de taille variable de symbo-
les d'entrée de données audiovisuelles par un code
entropique de sortie, le système comprenant :
- un tampon d'entrée pour stocker une série de
symboles de données audiovisuelles non com-
pressées à compresser et à transmettre au
client ;
une mémoire de sortie pour stocker un mot de
code entropique représentant une version
compressée de la série de symboles non com-
pressés dans le tampon d'entrée ;
- 25 dans lequel le système est **caractérisé par** :
- une mémoire de livre de codes pour stocker un
livre de codes préexistant contenant un mot de
code entropique pour une série de taille varia-
ble de symboles, dans laquelle le livre de codes
associe des séries de taille variable de symbo-
les à des mots de code entropique correspon-
dants pour une compression variable à
variable ;
un arrangement de recherche pour rechercher
dans un mot de code entropique une série par-
ticulière de symboles dans le livre de codes ; et
un arrangement de codage ayant une voie
d'entrée en communication avec le tampon
d'entrée et une sortie en communication avec
la mémoire de sortie ;
- 30 dans lequel l'arrangement de codage appli-
que l'arrangement de recherche pour rechercher
dans un mot de code entropique la série de symbo-
les non compressés pour un stockage dans la mé-
moire de sortie.
16. Système selon la revendication 15, comprenant en
outre des moyens de transmission pour transmettre
le contenu de la mémoire de sortie au client sur la
liaison de réseau (460).
17. Système selon la revendication 16, dans lequel un
protocole de réseau de transmission en continu est
utilisé pour communiquer sur la liaison de réseau
(460).

18. Système pour décoder des données audiovisuelles compressées reçues depuis une liaison de réseau (460), le système comprenant :

une mémoire d'entrée pour stocker un mot de code ; 5

un tampon de sortie pour stocker une série de symboles de données audiovisuelles non compressées correspondant au mot de code dans la mémoire d'entrée ; 10

dans lequel le système est **caractérisé par** :

une mémoire de livre de codes pour stocker un livre de codes préexistant contenant une série de taille variable de symboles pour un mot de code entropique, dans lequel le livre de codes associe des mots de code entropique à des séries correspondantes de taille variable de symboles pour une décompression variable à variable ; 15 20

un arrangement de recherche pour rechercher dans un mot de code entropique une série particulière de symboles dans le livre de codes ; et un arrangement de décodage ayant une voie d'entrée en communication avec la mémoire d'entrée et une sortie en communication avec le tampon de sortie ; 25

dans lequel l'arrangement de décodage applique l'arrangement de recherche pour rechercher la série de symboles non compressés correspondant au mot de code entropique et pour stocker une telle série dans le tampon de sortie. 30

35

19. Système selon la revendication 18, dans lequel le protocole de réseau de transmission en continu est utilisé pour communiquer sur la liaison de réseau (460).

40

20. Système selon la revendication 18, dans lequel l'arrangement de décodage est implémenté par un programme d'application qui implémente également le protocole de langage de balisage hypertexte.

45

50

55

FIG. 1

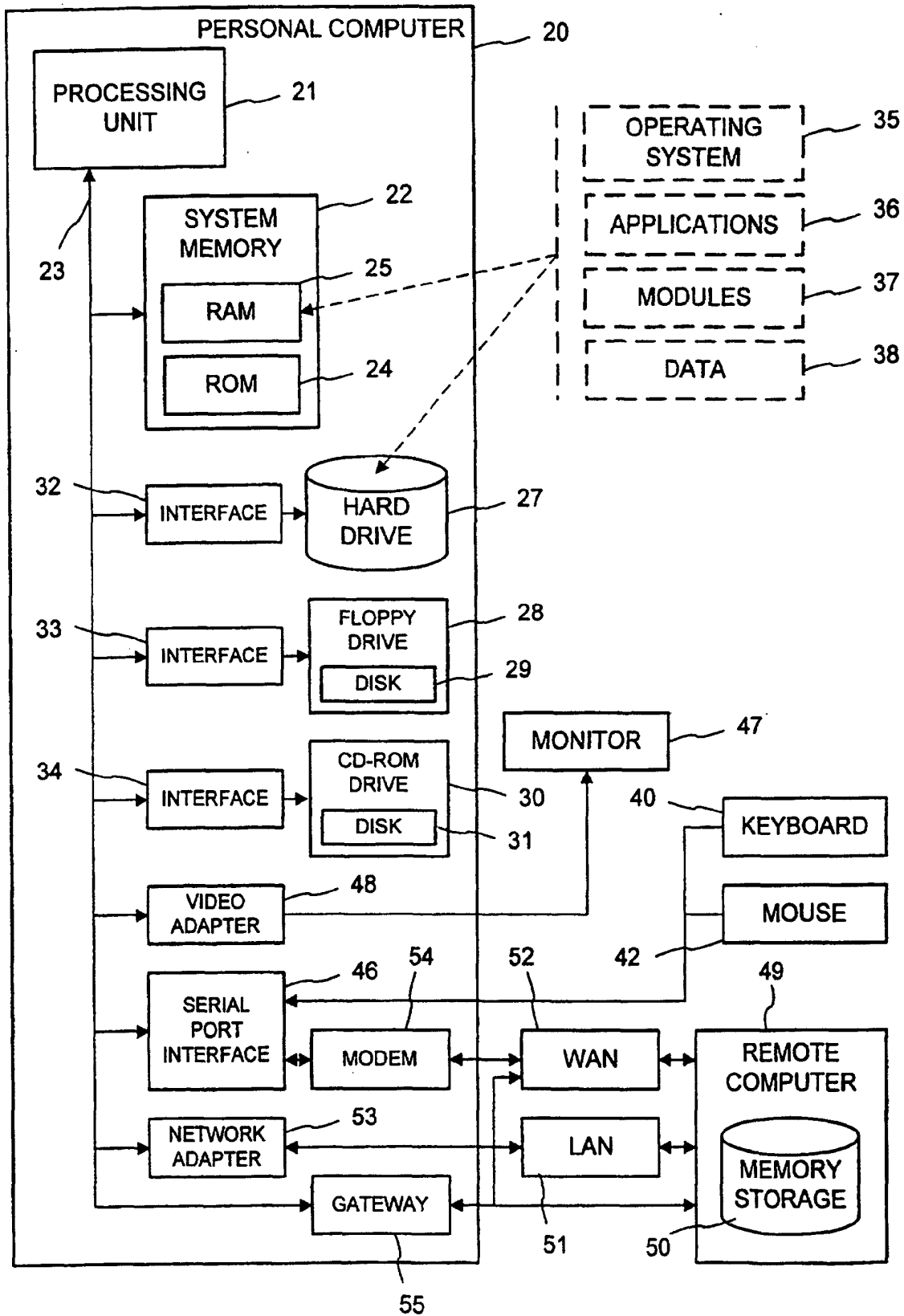


FIG. 2

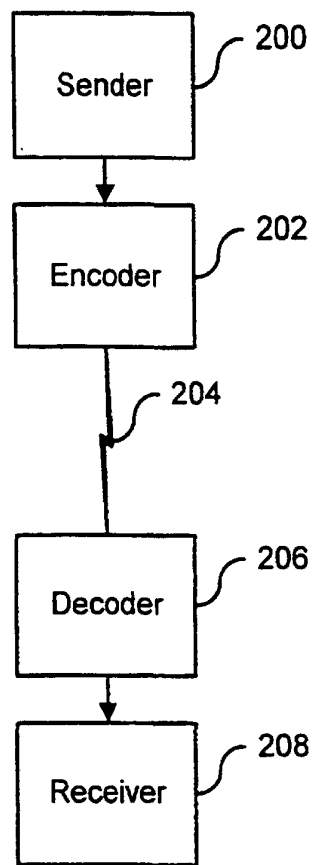


FIG. 3

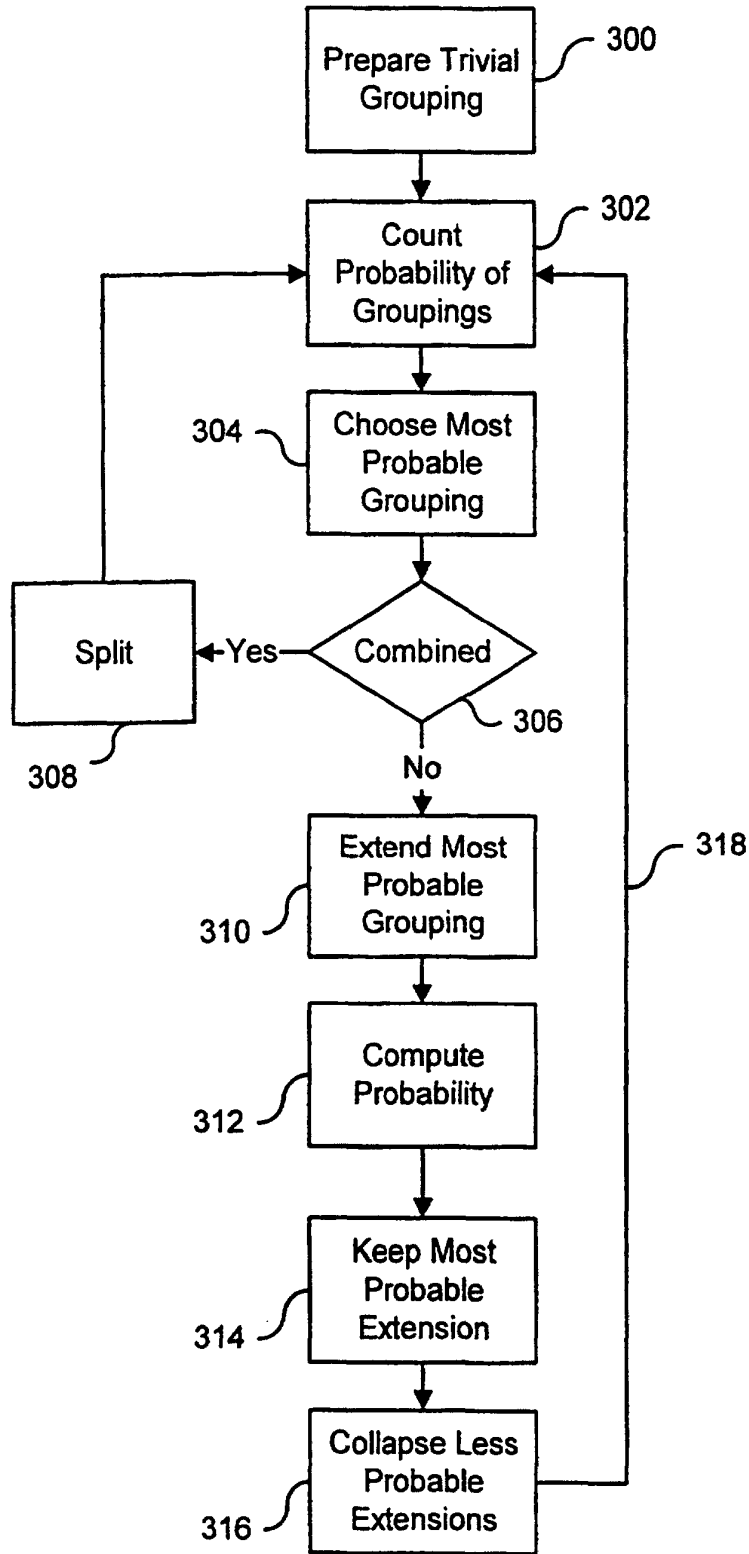
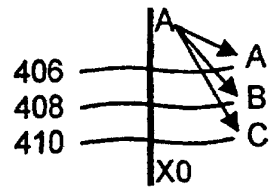


FIG. 4



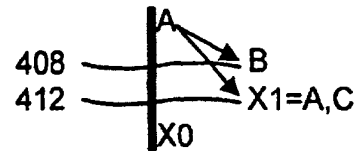
8/15 Expand
7/15

FIG. 5



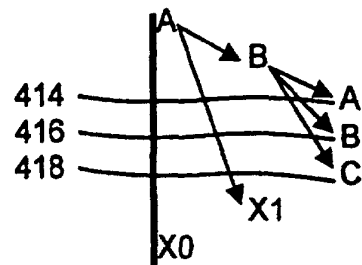
2/9
4/9 Keep
0/9

FIG. 6



4/9 Expand
2/9
3/9

FIG. 7



1/8 Keep
1/8
0/8
2/8
3/8

FIG. 8

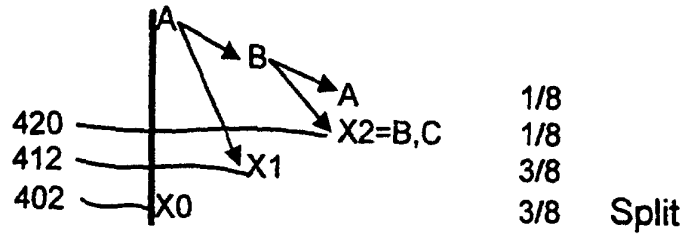


FIG. 9

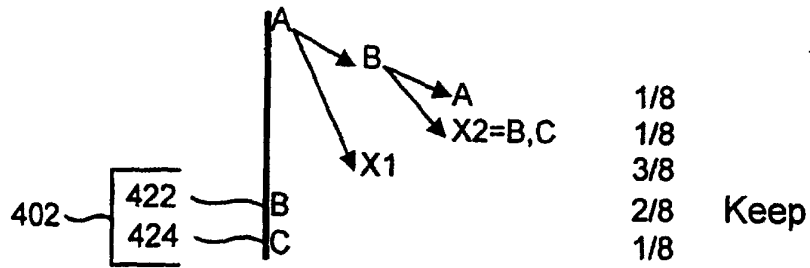


FIG. 10

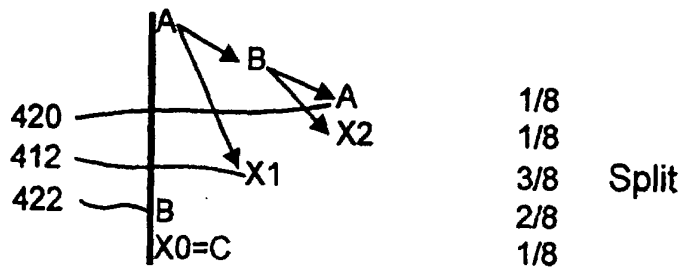


FIG. 11

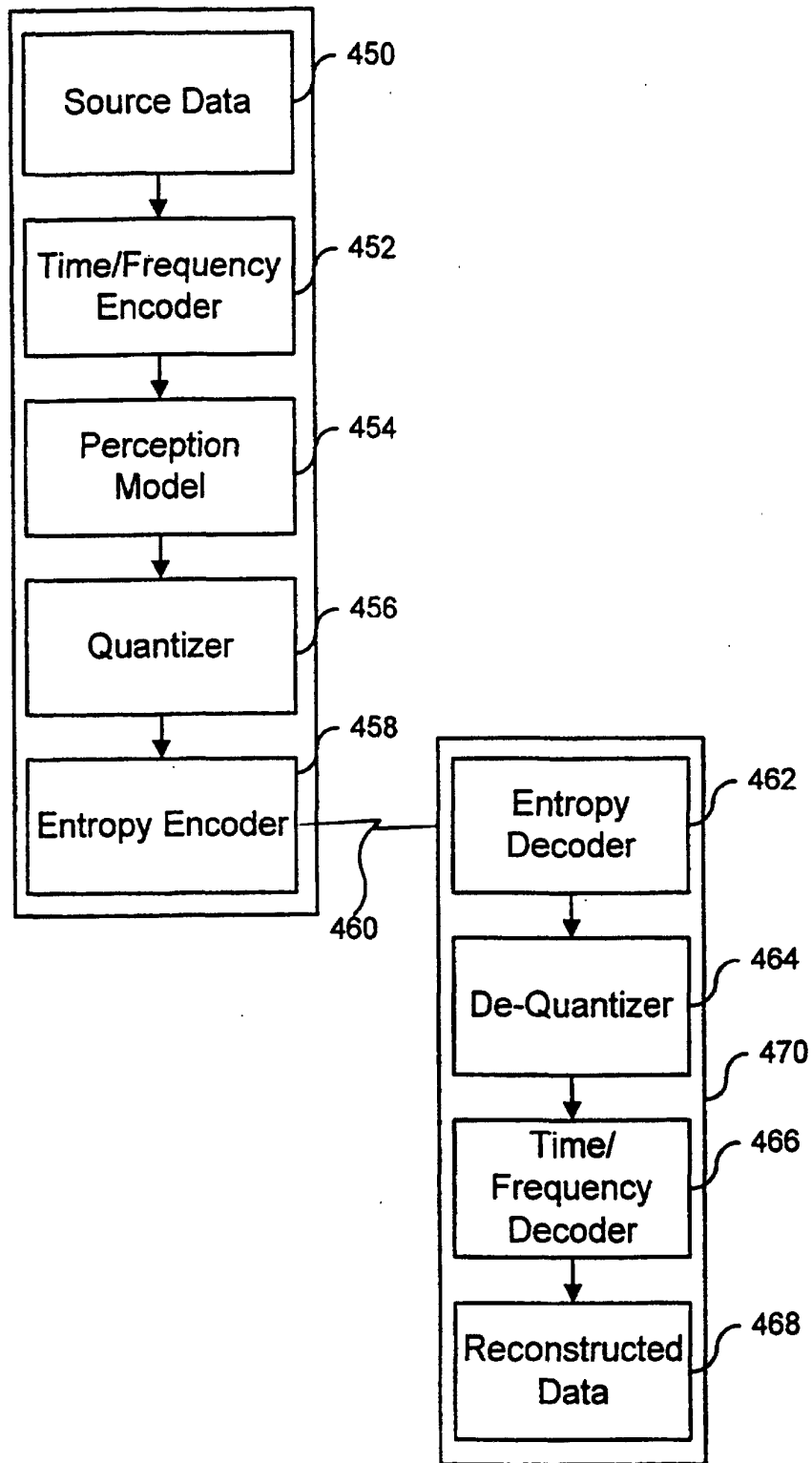


FIG. 12

