(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 289 762**

(51) Int. Cl.:
G06F 7/00 (2006.01)
G06F 15/00 (2006.01)
H03M 7/30 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Número de solicitud europea: **97938108 .4**
(86) Fecha de presentación : **04.08.1997**
(87) Número de publicación de la solicitud: **0970419**
(87) Fecha de publicación de la solicitud: **12.01.2000**

(54) Título: **Técnica de compresión de datos Lempel - Ziv que utiliza un diccionario prerrellenado con combinaciones frecuentes de letras, palabras y/o frases.**

(30) Prioridad: **06.08.1996 US 692474**

(45) Fecha de publicación de la mención BOPI:
01.02.2008

(45) Fecha de la publicación del folleto de la patente:
01.02.2008

(73) Titular/es: **Jeffrey C. Reynar**
1053 Church Street
Abington, Pennsylvania 19001, US
Frederick Herz;
Jason M. Eisner y
Lyle Ungar

(72) Inventor/es: **Reynar, Jeffrey C.;**
Herz, Frederick;
Eisner, Jason, M. y
Ungar, Lyle

(74) Agente: **Carpintero López, Francisco**

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín europeo de patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre concesión de Patentes Europeas).

Venta de fascículos: Oficina Española de Patentes y Marcas. Pº de la Castellana, 75 – 28071 Madrid

DESCRIPCIÓN

Técnica de compresión de datos Lempel - Ziv que utiliza un diccionario prerrellenado con combinaciones frecuentes de letras, palabras y/o frases.

Antecedentes de la invención

Campo de la invención

La presente invención se refiere a un procedimiento y a un aparato para comprimir y para descomprimir datos de texto almacenados en formato digital de una manera que no se pierdan. En otras palabras, los datos originales se reconstruyen en su formato original después de haber sufrido primero el proceso de compresión y después el proceso de descompresión. Se supone que los datos se sacan de un alfabeto particular que se especifica por adelantado, tal como el código ASCII, que consiste en una representación de 7 ó de 8 bits de un conjunto particular de caracteres.

Descripción de la técnica anterior

En la técnica anterior se describen muchos tipos diferentes de técnicas de compresión de texto. Las técnicas de compresión de texto descritas en este documento se basan en las técnicas de compresión de texto desarrolladas por Lempel y Ziv, quienes desarrollaron dos técnicas para la compresión de texto que son similares pero que tienen diferencias importantes. Estos dos procedimientos se esbozan en los documentos titulados "Un algoritmo universal para la compresión secuencial de datos", IEEE Transactions on Information Theory, vol. IT-23, núm. 3, páginas 337 a 343, y "Compresión de secuencias individuales a través de codificación a velocidad variable", IEEE Transactions on Information Theory, volumen IT-24, núm. 5, páginas 530 a 536, y a las que se hace referencia de manera común como LZ77 y LZ78, respectivamente.

LZ77 es una técnica de compresión de texto en la que se usan punteros a material anteriormente comprimido dentro de una ventana de tamaño fijo para comprimir nuevo material. Esta "ventana de compresión" de tamaño fijo se mueve a través de los datos de texto como si se estuviese comprimiendo para explotar el principio de localidad, es decir, qué datos son probables de ser los más similares a los datos próximos. Ahora se describirá un ejemplo de LZ77 con respecto a las figuras 1 (A) y 1 (B).

En las figuras 1 (A) y 1 (B), se supone un tamaño de ventana pequeño de 8 caracteres para propósitos ilustrativos. Como se muestra en la figura 1 (A), el texto que aún no haya sido comprimido se compara con el contenido de una ventana de (hasta) 8 caracteres que contiene los (hasta) 8 caracteres del texto comprimido que precede de manera inmediata al texto que aún no haya sido comprimido. Se identifica la coincidencia más larga que comienza al inicio del texto que aún no ha sido comprimido con una secuencia en la ventana de 8 caracteres. En la figura 1 (A), la coincidencia más larga de la ventana de 8 caracteres es "BB". Un puntero a esta secuencia ("BB"), su longitud (2) y una extensión (el siguiente carácter tras la coincidencia en el texto que se va a comparar) se transmiten o se comprimen después y se almacenan de manera local, dependiendo de la aplicación del algoritmo de compresión de datos. Sin embargo, si no se encuentra coincidencia de secuencia en la ventana de 8 caracteres, entonces se transmite un carácter literal. Una vez que el bloque de datos apuntado por el puntero se haya comprimido y la información relativa a esta compresión se haya transmitido, la ventana se mueve un número de caracteres al que se hace referencia por medio del puntero (BB) más la extensión, en caso de existir. Además, el puntero a la región de texto que está siendo comprimida se actualiza en este número de caracteres. Como se ilustra en la figura 1 (B), este proceso se repite para los datos desplazados hasta que se agotan los datos que se están comprimiendo.

LZ78 se diferencia de LZ77 en que la compresión de texto se consigue por medio del análisis gramatical de los datos que se estén comprimiendo en frases que se introducen dentro de un diccionario de compresión. Los punteros a estas frases o entradas de diccionario se usan después para comprimir nuevos datos. Inicialmente, el diccionario contiene solamente una cadena vacía (frase de longitud cero). La frase que se vaya a comprimir, en cada paso, es la frase más larga al inicio de los nuevos datos de forma que el prefijo de esta frase es una entrada en el diccionario, en el que está definido el prefijo para que sea la frase con carácter final eliminado. El carácter restante se denomina la extensión. De esta manera, cuando se ve la primera frase, se codifica como una referencia a una entrada de diccionario que consiste en la cadena vacía (que es la única entrada inicialmente encontrada en el diccionario), seguida por el último y único carácter de la frase. Este carácter se coloca entonces en el diccionario (suponiendo que el diccionario no está lleno), y se repite el proceso de identificación de una frase y la transmisión de su prefijo (como una referencia a la entrada del diccionario que coincida con el prefijo) y su extensión. Los datos que aún no han sido comprimidos se compararán después con ambas entradas de diccionario: la cadena vacía y la frase que consiste en el carácter que ya se ha encontrado. Si el siguiente carácter en los nuevos datos no coinciden con el carácter ya comprimido, entonces éstos también se comprimirán como la cadena vacía más el carácter que se esté comprimiendo. De esta manera, cada frase de los datos se comprime como un prefijo, que se encuentra en el diccionario y que se elige para que sea tan largo como sea posible, y una extensión, que es el carácter que sigue al prefijo en los datos de entrada. Ahora se describirá un ejemplo de LZ78 con respecto a la figura 2.

La figura 2 muestra una muestra de la compresión LZ78 para una cadena corta de caracteres. Como se muestra, el diccionario inicialmente comienza sin entradas aparte de la cadena vacía, a la que se hace referencia como $\in$, y se coloca un puntero que indica el inicio de la secuencia de caracteres que se vaya a comprimir al comienzo de la

secuencia que se vaya a comprimir. La frase inicial más larga cuyo prefijo está en el diccionario es de un carácter de longitud, ya que el prefijo de esta frase, a saber, ∈, es la única entrada en el diccionario. El primer carácter por lo tanto se codifica como una referencia a ∈ y al primer carácter de la secuencia que se esté transmitiendo. Después, se actualiza el diccionario para que contenga la entrada que consiste en la concatenación de la entrada de diccionario usada y el carácter que la sigue en la secuencia de compresión. El puntero actual se mueve entonces en el número de caracteres comprimidos, y el proceso se repite el mismo, de manera repetitiva identificando la siguiente frase y transmitiendo una versión comprimida de la misma, hasta que el flujo de datos que se vaya a comprimir esté vacío. Como apreciarán los que sean expertos en la técnica, la técnica LZ78 proporciona de manera sustancial más compresión una vez que el diccionario está formado. En la Patente de los Estados Unidos número 4.464.650 - Eastman y colaboradores, se da una descripción más detallada de una implementación particular de la técnica de compresión de texto LZ78, mientras que una buena descripción general de las técnicas de codificación Lempel - Ziv se puede encontrar en el texto titulado "Compresión de texto", de Bell y colaboradores, Englewood Cliffs, New Jersey, Prentice Hall, 1990.

Se han descrito numerosos sistemas de compresión de datos en la técnica primera que utilizan el concepto de un diccionario de compresión como describe Lempel y Ziv.

Por ejemplo, Giltner y colaboradores describe en la Patente de los Estados Unidos número 4.386.416 un sistema para su uso en la transmisión de datos sobre una red de teletexto o una red similar. El sistema descrito por Giltner y colaboradores usa dos diccionarios. El primero de ellos está prerrellenado con palabras frecuentes del lenguaje de los datos, mientras que el segundo diccionario está inicialmente vacío y se rellena con palabras que se encuentran en los datos pero que no están presentes en el primer diccionario. Cuando se transmiten datos, si se encuentra una palabra en el primer diccionario, se transmiten un código de escape y el número de la entrada de palabra en el primer diccionario. Si no se encuentra una palabra en el primer diccionario, se comprime usando la codificación Huffman y se añade al segundo diccionario para su uso con posterioridad. Como resultado de esto, si se encuentra la palabra de nuevo, se puede transmitir enviando un código de escape indicando que el número de la entrada de palabra en el segundo diccionario se refiere al segundo diccionario seguido por el número de la entrada de palabra en el segundo diccionario. Giltner y colaboradores definen una "palabra" para que sea un número predeterminado de caracteres o una secuencia de caracteres rodeada por un espacio en blanco o una combinación de espacio en blanco y signo de puntuación. En el primer diccionario se proporciona en el primer diccionario un número pequeño, pero fijo de palabras comunes para todos los tipos de mensajes manejados por el Teletexto o una red similar, y las "palabras" adicionales se almacenan en el segundo diccionario. Sin embargo, Giltner y colaboradores no se ocupan de secuencias de texto que ocurren frecuentemente y que caen fuera de la definición limitada de una palabra válida. Como resultado de esto, Giltner y colaboradores no toman ventaja del hecho de que la similitud de textos es mayor cuando la comparación entre ellos se hace al nivel de secuencias de caracteres. También, Giltner y colaboradores no se preocupan de cómo se pueden elegir para el primer diccionario palabras que ocurren de manera frecuente en el texto, por lo cual se rellena con palabras válidas que son frecuentes dentro del tipo de texto que se vaya a transmitir. Giltner y colaboradores también fallan en enseñar cómo identificar la librería de texto más apropiada o en la identificación del género del documento que se vaya a comprimir. Además, la librería de Giltner y colaboradores es fija; los usuarios no pueden crear sus propios diccionarios prerrellenados según lo necesiten.

De manera similar, Weng describe en la Patente de los Estados Unidos número 4.881.075 una técnica de compresión de datos "adaptativa" que usa dos diccionarios. El primer diccionario se usa para realizar la compresión o la descompresión, mientras que el segundo diccionario se está reconstruyendo para que refleje mejor las características locales de los datos de entrada más recientes. El segundo diccionario se usa entonces para comprimir y descomprimir los datos de entrada mientras que el primer diccionario se está reconstruyendo usando los datos de entrada más actuales. Weng cambió de manera repetida entre diccionarios hasta completar la compresión.

Kato y colaboradores describe en la Patente de los Estados Unidos número 4.847.619 una modificación a las técnicas de compresión adaptativas en las que se supervisa el grado de compresión del sistema de compresión y el diccionario se reinicializa cuando el grado de compresión cae por debajo de un umbral. No se permite que la reinicialización se produzca antes de que el diccionario esté lo suficientemente lleno con el fin de evitar que el diccionario se reinicialice de manera prematura. Esta técnica se podría usar junto con la técnica de compresión LZ o cualquier otra técnica adaptativa.

En la Patente de los Estados Unidos número 5.153.591, Clark describe una modificación al algoritmo de compresión Lempel - Ziv en el que el diccionario se almacena como una estructura de datos de árbol. Esto permite almacenar grandes diccionarios en menos espacio que en la realización original descrita la Patente de los Estados Unidos número 4.464.650. Además, permite buscar en estos diccionarios de una manera más fácil y más rápida.

En la Patente de los Estados Unidos número 5.243.341, Seroussi y colaboradores esbozan una variante de Lempel - Ziv en la que se usan dos diccionarios. El primer diccionario se usa hasta que se llena, y después se sustituye con un diccionario de reserva, que se rellena con esas entradas del primer diccionario que producen la mayoría de la compresión antes de que continúe la compresión.

En la técnica anterior aparecen muchas otras modificaciones a las técnicas de compresión originales de Lempel - Ziv.

Por ejemplo, Welch describe en la Patente de los Estados Unidos número 4.558.302 una implementación de Lempel - Ziv en la que los procesos de codificación y de descodificación requieren menos cálculo complicado y por lo tanto son más rápidos que en la implementación descrita en la Patente número 4.464.650 - Eastman y colaboradores.

Miller y colaboradores sugieren en la Patente de los Estados Unidos número 4.814.746 varias modificaciones al algoritmo de Lempel - Ziv. La primera de estas modificaciones es incluir todos los posibles caracteres en el diccionario antes de que comience realmente la compresión. Como resultado de esto, no es necesario transmitir un indicador que indique que el siguiente dato es un carácter en lugar de un puntero. Además, Miller y colaboradores también asocian una consigna de hora con cada entrada en el diccionario con el fin de facilitar la retirada de la entrada usada menos reciente cuando el diccionario comience a estar lleno. Estas modificaciones están dirigidas a reducir los requisitos de memoria mediante la limitación del diccionario a un tamaño fijo y mejorando la compresión mediante el permitir que el diccionario refleje de manera más precisa las características actuales de los datos que se vayan a comprimir.

Storer describe en la Patente de los Estados Unidos número 4.876.541 una técnica de compresión que no sufre de algunas de las mismas dificultades que las técnicas anteriores de Lempel - Ziv. En particular, los caracteres no codificados no necesitan nunca ser transmitidos ya que el diccionario de compresión contiene inicialmente todos los caracteres del alfabeto, como en la Patente de los Estados Unidos número 4.814.746 - Miller y colaboradores. Además, se mantiene una cola que es la menos recientemente usada, de forma que se puede depurar el diccionario de las entradas menos útiles. El diccionario de codificación y descodificación del sistema de Storer puede variar en el tamaño y puede haber varios activos a un tiempo. La relación de compresión de cada uno de los diccionarios se supervisa, y el diccionario que produzca la mejor compresión es el que se usa.

En la Patente de los Estados Unidos número 4.906.991, de Fiala y colaboradores, se describe una técnica de compresión de datos de estilo de sustitución que es algo similar a la compresión de Lempel - Ziv. Su técnica depende de la búsqueda de una ventana fija de caracteres (por ejemplo, 4096 caracteres) que ya se hayan comprimido con el fin de determinar si el texto que se está comprimiendo se puede codificar como un puntero a una localización dentro de la ventana. Si el texto que se está comprimiendo no se puede codificar de esta manera, se genera un puntero a la localización de inicio junto con la longitud del solape entre el texto que se está comprimiendo y la localización dentro de la ventana. Si el texto que se está comprimiendo no se puede codificar de esta manera, se codifica como una longitud seguida por una cadena literal de esa longitud. Como en la compresión LZ78, esta técnica falla en la compresión de datos mucho al comienzo de los documentos, ya que la ventana está desprovista de cadenas a las que se pueda apuntar con el fin de provocar la compresión.

O'Brien sugiere en la Patente de los Estados Unidos número 4.988.998 modificaciones al algoritmo de Lempel - Ziv que permiten una compresión mejorada de datos que contengan largas cadenas de caracteres repetidos. Como el algoritmo de Lempel - Ziv añade entradas al diccionario de compresión mediante la anexión de un único carácter a una entrada de diccionario existente, cogerá muchas ocurrencias de una cadena repetida de caracteres antes de que dichas cadenas se encuentren en el diccionario. De acuerdo con esto, O'Brien preprocesa los datos usando una técnica de codificación de coordenada diferencial en la que se insertan las coordenadas diferenciales dentro del texto. La combinación resultante de texto y de coordenadas diferenciales para caracteres repetidos se comprime después usando la técnica de Lempel - Ziv.

En la Patente de los Estados Unidos número 5.049.881, Gibson y colaboradores describen un sistema de compresión de datos que crea sus propios punteros a partir de la secuencia de caracteres anteriormente procesada y enfatiza la maximización del producto de la velocidad de datos y la relación de compresión. De esta manera, los datos previamente introducidos se usan mientras el diccionario se combina con un algoritmo de cálculo de clave para encontrar candidatos para cadenas que coincidan sin el requisito de una tabla de coincidencia de cadenas.

En la Patente de los Estados Unidos número 5.058.137, Shah describe un descodificador Lempel - Ziv que tiene unas memorias para almacenar palabras de código y datos de manera independiente. Al producirse la recepción de una palabra de código, el descodificador almacena la palabra de código previamente recibida, aplica la palabra de código nuevamente recibida a la memoria de palabras de código para obtener la localización del último elemento de datos que es parte de los datos representados por la nueva palabra de código recibida y otra palabra de código asociada con el prefijo. Al producirse la finalización de la descodificación de la última palabra de código, el primer elemento de datos de la palabra descodificada se anexa a la siguiente palabra de código anteriormente recibida, y la combinación se almacena como el equivalente de una palabra de código que sea la siguiente después de que se haya recibido ya la palabra de código más alta. Se comparte al menos una memoria para su uso durante la codificación y la descodificación.

En la Patente de los Estados Unidos número 5.087.913, Eastman describe un algoritmo de Lempel - Ziv que usa una base de datos de árbol de búsqueda para permitir a las partes últimas de los datos ser descomprimidas sin tener que descomprimir todas las partes anteriores. La base de datos de árbol de búsqueda crece hasta un tamaño fijo predeterminado y no se permite que crezca más. El hecho de que la base de datos de árbol de búsqueda de compresión se establezca por adelantado a la descompresión permite la descompresión de partes de los datos sin descomprimir toda la parte anterior de los datos.

En la Patente de los Estados Unidos número 5.140.321, Jung detalla una modificación de Lempel - Ziv que permite una velocidad de compresión mejorada a costa de una reducción en la compresión. En lugar de intentar encontrar la

subcadena de coincidencia óptima en la parte entera comprimida de los datos, se explota el principio de localidad y solamente se examinan los datos comprimidos más recientes en una memoria intermedia de almacenamiento temporal de primero en entrar, primero en salir para encontrar una secuencia de coincidencia. Se usa una tabla de cálculo de clave para almacenar las cadenas que se hayan comprimido recientemente y para permitir la última recuperación de las cadenas de coincidencia.

En la Patente de los Estados Unidos, número 5.179.378, Ranganathan y colaboradores describen una implementación de Lempel - Ziv que usa una matriz sistólica de procesadores para mejorar el funcionamiento mediante la formación de palabras de código de longitud fija a partir un número variable de símbolos de datos.

En la Patente de los Estados Unidos número 5.262.776, Kutka describe una implementación del algoritmo de Lempel - Ziv que saca ventaja de la estructura de datos en árbol para evitar el paso de búsqueda que normalmente requiere el proceso de compresión. La secuencia de elementos en una secuencia primaria se convierte en elementos en un conjunto reducido de elementos usando secuencias de escape. Esta técnica es particularmente adecuada para la compresión de datos que representen los coeficientes de una transformada coseno discreto de una imagen.

En el documento de los Estados Unidos 5.561.421 se describe un sistema para muestrear un bloque de subcadena de datos de entrada que va a ser comprimido y para seleccionar un segmento de diccionario adecuado para comprimir cadenas con las características del bloque de entrada. Los segmentos de diccionario se usan durante el muestreo para constituir un diccionario de compresión construido en el sistema que se usa para comprimir el resto si se recibe la cadena de entrada como esta.

Además de lo anterior que incluye modificaciones a la técnica de compresión Lempel - Ziv, que se describen en la literatura de patentes, otras aparecen en artículos técnicos.

Por ejemplo, en “Algoritmo lineal para la compresión de datos a través de coincidencia de cadenas”, de Rodeh y colaboradores, se describe una modificación a la técnica LZ77 en la que el tamaño de la ventana no es fijo. Como resultado de esto, los punteros a las cadenas anteriores en la parte comprimida de los datos crecen en longitud y se codifican con un código de longitud variable.

En “Mejor compresión de texto OPM/L”, Bell describe una variante Lempel - Ziv a la que se hace referencia como LZSS en la que no toda la compresión se hace usando una combinación de un prefijo y una extensión. En lugar de esto, si el coste de transmitir un puntero es más alto que el de meramente transmitir un carácter o una secuencia de caracteres, entonces se transmite el carácter o la secuencia de caracteres. Se usa un árbol de búsqueda binario para encontrar la coincidencia de cadena más larga, sin un límite en la longitud de la coincidencia.

En “Una técnica para la compresión de datos de alto rendimiento”, Welch describe una modificación a LZ78 en la que solamente se usan los punteros a los datos anteriormente comprimidos, en lugar de una combinación de punteros y de caracteres. Se forma una tabla de cadenas que es una correspondencia de cadenas de caracteres de entrada con códigos de longitud fija. Para cada cadena en la tabla, su cadena prefijo también está en la tabla. La tabla de cadenas contiene cadenas que se han encontrado con anterioridad en el mensaje que se está comprimiendo. Consiste en una muestra de pasada de cadenas en el mensaje de forma que las cadenas disponibles reflejen las estadísticas del mensaje. Se hace referencia comúnmente a esta técnica como LZW. Usa un algoritmo de análisis gramatical “a conciencia” en el que la cadena de entrada se examina en serie de caracteres en una pasada, y la cadena de entrada más larga reconocida se analiza gramaticalmente cada vez. Las cadenas añadidas a la tabla de cadenas se determinan por medio de este análisis gramatical.

En “Variaciones sobre un tema por Ziv y Lempel”, Miller y Wagman describen otra variante de LZ78. En su versión, el diccionario se rellena por adelantado con todas las cadenas de longitud 1 (esto es, todos los caracteres del alfabeto sobre los que está teniendo lugar la compresión), lo que ayuda a reducir, pero no elimina el problema de iniciar con un diccionario desprovisto de entradas útiles. También, en lugar de reinicializar el diccionario de compresión cuando comienza a estar lleno, proponen borrar cadenas del diccionario que fueron las menos recientes en usarse. Sin embargo, la contribución mayor de su versión, a la que se hará referencia como LZMW, es que las extensiones nunca se transmiten. En lugar de esto, como el diccionario comienza con todas las cadenas de longitud 1, es posible codificar todos los datos usando el diccionario inicial. Sin embargo, esto daría como resultado una no compresión. En lugar de esto, el diccionario crece mediante la adición de entradas al diccionario que consisten en la concatenación de las dos anteriores coincidencias.

En la figura 3 se muestra una muestra de compresión LZMW. El alfabeto se supone que contiene solamente 3 caracteres (A, B, C) para propósitos de ilustración. Como se muestra, el diccionario de compresión contiene inicialmente los caracteres del alfabeto. El puntero al carácter en el que comenzará la compresión se coloca en el primer carácter de la secuencia del texto que se esté comprimiendo. Se encuentra la coincidencia más larga dentro del diccionario y se transmite un puntero a esta entrada. De manera alternativa, el puntero se puede almacenar localmente dependiendo de la aplicación del algoritmo de compresión. El puntero se mueve después el número de caracteres transmitidos. Normalmente, el diccionario se actualiza para que contenga la concatenación de las dos entradas de diccionario anteriormente transmitidas, pero como no hay anteriores transmisiones, este paso se omite. En la segunda iteración a través del ciclo de procesado, se encuentra la coincidencia más larga, se transmite su número de entrada de diccionario, el puntero se mueve el número apropiado de caracteres y se añade la concatenación de las dos entradas de diccionario anteriores

transmitidas al diccionario como una nueva entrada. Este proceso se repite después hasta que los datos que se vayan a comprimir se hayan agotado.

Algoritmos de compresión comúnmente usados usan también variaciones de los algoritmos LZ77 y LZ78. Por ejemplo, el algoritmo de compresión usado en "zip" comercialmente disponibles y en "gzip" gratuitamente disponibles es una variación de LZ77 que encuentra cadenas duplicadas en los datos de entrada. En "gzip", la segunda ocurrencia de una cadena se sustituye por un puntero a la cadena anterior en forma de un par (distancia, longitud). Cuando una cadena no ocurre en ningún lado en el número anterior de octetos dentro de una distancia designada, tal como 32 kbytes, se transmite como una secuencia de octetos literales. Los literales o las longitudes de coincidencia se comprimen con un árbol de Huffman y las distancias de coincidencia se comprimen con otro árbol. Los árboles se almacenan en un formato compacto al inicio de cada bloque. Los bloques pueden tener cualquier tamaño, excepto que los datos comprimidos para un bloque deben ajustarse en la memoria disponible. Un bloque se termina cuando "gzip" determina que debería ser útil iniciar otro bloque con árboles de refresco. Las cadenas duplicadas se encuentran usando una tabla de cálculo de clave. Todas las cadenas de entrada de longitud 3 se insertan en la tabla de cálculo de clave, y se calcula un índice de cálculo de clave para los siguientes tres octetos. Si la cadena de cálculo de clave para este índice no está vacía, todas las cadenas de la cadena se comparan con la cadena de entrada actual, y se selecciona la coincidencia más larga. Las cadenas de cálculo de clave se buscan, comenzando con las cadenas más recientes, para favorecer pequeñas distancias y de esta forma sacar ventaja de la codificación de Huffman. Las cadenas de cálculo de clave están simplemente enlazadas. No se borra nada de las cadenas de cálculo de clave; el algoritmo simplemente descarta las coincidencias que son demasiado antiguas. Para evitar una situación del peor caso, las cadenas de cálculo de clave muy largas se truncan de manera arbitraria a una cierta longitud como se determina por medio de una opción en tiempo de ejecución. Como resultado de esto, "gzip" no siempre encuentra la coincidencia más larga posible pero generalmente encuentra una coincidencia que es lo suficientemente larga.

Desafortunadamente, a pesar del gran número de variantes de Lempel - Ziv en la técnica anterior, ninguna se ocupa de manera adecuada del problema de que comenzar con un diccionario completamente desprovisto de palabras evita de manera virtual que pequeños ficheros sean comprimidos al completo y evita que largos ficheros sean comprimidos con un grado más alto de compresión. La Patente de los Estados Unidos número 4.814.746 y la Patente de los Estados Unidos número 4.876.541 y el trabajo de Miller y Wegman comienzan a ocuparse de este problema comenzando con un diccionario que contiene todos los caracteres que se encuentran en el conjunto de caracteres en el que se codificaron los datos. Esto resuelve el problema de tener que transmitir códigos de escape para indicar que lo que sigue no es un número de entrada de diccionario, sino un carácter.

Sin embargo, los presentes inventores han descubierto que una compresión más profunda se puede obtener observando que muchos documentos, en especial aquellos documentos de texto escritos en idiomas humanos naturales o en lenguajes de programación de ordenador, tales como el inglés o el lenguaje de programación C tienen un pequeño número de palabras que son de manera estadística extremadamente frecuentes y que deberían ser parte del diccionario de compresión. Zipf ha mostrado que esto es cierto para el lenguaje en un libro titulado *Comportamiento Humano y el Principio del Mínimo Esfuerzo*. De hecho, la frecuencia de palabras en inglés obedece a lo que ha venido a conocerse como una distribución Zipfiana. Esto es, el producto de la clasificación de una palabra y su frecuencia es aproximadamente constante. De esta manera, la segunda palabra más común aparecerá casi la mitad de veces que la palabra más frecuente. Esto implica que las palabras más comunes comprenderán una fracción grande de las ocurrencias totales de todas las palabras en un documento. Por ejemplo, en el Wall Street Journal, los datos recogidos como parte del proyecto de "treebank" *(un "treebank" es un cuerpo de texto en el que cada frase se ha anotado con su estructura sintáctica)*, descrito en un artículo de Marcus y colaboradores titulado *Building a Large Annotated Corpus of English: The Penn Treebank*, Computational Linguistics, volumen 19, número 2, páginas 313 a 330 (1993), las primeras diez "palabras" que eran ",", "the", ".", "of", "to", "a", "and", "in", "'s", "is" y "that" contaban ligeramente más del 25 por ciento de todas las palabras del cuerpo. Las 100 palabras más frecuentes solas contaban el 48,1 por ciento de todas las palabras del cuerpo, mientras que las 500 palabras más frecuentes contaban el 63,6 por ciento. Esto significa que las restantes 80901 palabras contaban el restante 37,4 por ciento de todas las palabras en el cuerpo. De esta manera, la palabra promedio encontrada en las quinientas palabras más repetidas es aproximadamente 275 veces más frecuente que la palabra promedio no encontrada en las quinientas palabras más repetidas.

Se desea de esta manera modificar las técnicas de compresión de texto de Lempel - Ziv descritas en la técnica anterior para tomar ventaja de esta observación de forma que se pueda permitir una compresión adicional de grandes documentos así como una compresión significativa de pequeños documentos.

Sumario de la invención

La presente invención se refiere a las técnicas de compresión de texto de Lempel - Ziv de la técnica anterior. En particular, la presente invención proporciona un diccionario de compresión y/o una ventana de compresión que contiene algunas de las palabras estadísticamente significativas que casi permitirán ciertamente conseguir la compresión antes dentro de un documento que si se requiriese software de compresión para volver a aprender la lista de palabras, secuencias de letras y frases extremadamente comunes en el documento que se vaya a comprimir. Por ejemplo, cuando se usen las técnicas de compresión basadas en LZ78, en las que las entradas son añadidas al diccionario de compresión mediante la extensión de las entradas anteriores en un sólo carácter, la invención hace innecesario coger varias repeticiones de palabras incluso extremadamente frecuentes antes de que se introduzcan en el diccionario de compresión. Esto también significa que el diccionario de compresión usado de acuerdo con la invención no contiene

necesariamente prefijos de las palabras largas, frecuentes que están inicialmente presentes en el diccionario, a menos que dichos prefijos sean por sí mismos frecuentes. Los prefijos raros ayudan al proceso de compresión poco, y su principal contribución en LZ78 es permitir la introducción de una palabra larga en el diccionario de compresión, un paso que es innecesario cuando se usan las técnicas de la invención, en el que la palabra larga y frecuente está en el diccionario en el comienzo. Mediante la omisión de los prefijos raros del diccionario se deja espacio para otras frases más útiles.

Mientras que las palabras más extremadamente frecuentes son cortas, los presentes inventores reconocen que el conocimiento del tipo de texto que se va a comprimir puede llevar a un beneficio mayor desde un diccionario prerrellenado. Si, por ejemplo, se está comprimiendo texto científico, entonces las muestras de textos similares se pueden usar para conseguir un perfil del tipo de texto que se está comprimiendo. En textos científicos, algunas frases largas pueden ser extremadamente comunes. Estas frases deberían ser aprendidas por el software de compresión y se deberían usar cuando se haga la primera referencia a las mismas en los datos que se vayan a comprimir. De esta forma, si por ejemplo, una palabra o una frase de longitud 10 fuese común, entonces incluso la primera ocurrencia de esa palabra se podría codificar usando una única referencia de diccionario, mientras que bajo el LZ78 convencional, dicha codificación no podría ser posible hasta que se produzca la undécima ocurrencia de la palabra. También se evitará que el diccionario contenga subcadenas de esta palabra o frase frecuente como sería el caso en LZ78 a menos que dichas subcadenas sean añadidas de manera independiente durante el proceso de compresión porque sean útiles fuera del contexto de la palabra o de la frase frecuente.

Cuando se comprime texto que está en el lenguaje de programación de ordenador, los presentes inventores han reconocido que potencialmente hay más incluso que ganar al comienzo de la compresión. El lenguaje de programación C por ejemplo, tiene un pequeño número de palabras que representan constructores en el lenguaje. Mientras que el usuario puede añadir a la lista de estas palabras que ocurren de manera frecuente mediante la creación de variables y de funciones, todos los programas en C harán uso de un subconjunto de este conjunto básico de palabras. Como resultado de esto, estas palabras en el diccionario antes del inicio de la compresión darán como resultado una mejor compresión, en especial para pequeños ficheros.

De esta manera, con el fin de sacar ventaja de la distribución de la frecuencia de palabras en idiomas humanos y otras clasificaciones de datos también, los presentes inventores proponen modificar los algoritmos de compresión estándar de Lempel - Ziv para incorporar un diccionario prerrellenado con palabras estadísticamente importantes y el alfabeto completo sobre los que está teniendo lugar la compresión. El proceso de rellenar el diccionario con palabras que significativas se maneja de cualquier número de maneras. Una posibilidad es para un usuario del software de compresión para que cree una lista de palabras frecuentemente usadas. Como este proceso tiene objeciones en la mayoría de las circunstancias, los procesos automáticos son usados de manera preferible de acuerdo con la invención. Sin embargo, el usuario del software de compresión puede crear diccionarios de compresión prerrellenados adicionales según desee.

En la reivindicación independiente 1 se define un procedimiento para comprimir de acuerdo con la invención. El aspecto de descompresión de la invención se define en la reivindicación independiente 16. Los aspectos del aparato se definen en las reivindicaciones independientes 18 a la 31. Realizaciones adicionales de la invención se especifican en las reivindicaciones dependientes anejas.

De acuerdo con la invención, se pueden probar de manera automática varios diccionarios de compresión prerrellenados al inicio del proceso de compresión sobre un pequeño subconjunto del documento entero que se vaya a comprimir, y el diccionario que consiga la compresión más alta sobre este subconjunto es el elegido para la compresión de todo el documento. Por supuesto, el usuario puede seleccionar también un diccionario de compresión particular según desee. De manera preferible, la identidad del diccionario prerrellenado que se esté usando para la compresión se almacena en los datos comprimidos de forma que el software de descompresión pueda identificar qué diccionario prerrellenado usar durante la descompresión.

El mismo diccionario prerrellenado debe encontrarse disponible tanto para el dispositivo de compresión en el momento de la compresión, como para el dispositivo de descompresión en el momento de la descompresión. Es posible transmitir dichos diccionarios desde un dispositivo a otro sobre una red de comunicaciones, según sea necesario, para hacer que los diccionarios apropiados se encuentren disponibles. Por ejemplo, se pueden almacenar una pluralidad de diccionarios de compresión de datos prerrellenados en servidores de comunicaciones de forma que cualquier texto compartido entre los servidores se pueda comprimir/descomprimir usando las técnicas de la invención. También, los diccionarios de compresión de datos prerrellenados son creados de manera preferible para diferentes géneros de datos de texto y son almacenados de manera jerárquica de forma que el diccionario de compresión de datos prerrellenado óptimo se pueda seleccionar para los datos que se vayan a transmitir. Con el fin de ahorrar espacio en memoria, cualquier entrada común entre la pluralidad de diccionarios de compresión de datos prerrellenados solamente necesita ser almacenada una vez en cada servidor. Por otra parte, un diccionario de compresión de datos prerrellenado usado para comprimir un documento particular puede ser el diccionario específico de datos formado durante la compresión de otro texto relacionado. De esta manera, la compresión de una pluralidad de documentos puede ser iterativa y basada en un único diccionario de compresión de datos prerrellenado original.

Como resultará patente a los expertos en la técnica, las técnicas de la invención se pueden usar junto con cualquiera de las variantes conocidas de los algoritmos de compresión Lempel - Ziv.

Breve descripción de los dibujos

Los objetos y ventajas anteriores y otros objetos y ventajas de la invención serán más aparentes y serán más fácilmente apreciados a partir de la siguiente descripción detallada de las realizaciones de ejemplo actualmente preferidas de la invención tomadas junto con los dibujos acompañantes, en los que:

Las figuras 1 (A) y 1 (B) juntas proporcionan un ejemplo de compresión de texto LZ77 de la técnica anterior.

La figura 2 es un ejemplo de compresión de texto LZ78 de la técnica anterior.

La figura 3 es un ejemplo de compresión de texto LZMW de la técnica anterior.

La figura 4 ilustra una configuración hardware de muestra para implementar la técnica de compresión Lempel - Ziv modificada de acuerdo con la invención.

La figura 5 ilustra las opciones para el uso de los datos comprimidos generados de acuerdo con la invención.

La figura 6 ilustra un diagrama de flujo del proceso de creación del diccionario de acuerdo con la invención, por lo que la secuencia seleccionada para la entrada dentro del diccionario es la secuencia que maximiza la compresión.

La figura 7 ilustra una realización preferida de un sistema de compresión LZ78 que usa uno o más diccionarios prerrellenados de acuerdo con las técnicas de la invención.

La figura 8 es un diagrama de flujo que ilustra una técnica de compresión LZ78 modificada de acuerdo con la invención.

La figura 9 ilustra una realización preferida de un sistema de descompresión LZ78 que usa uno o más diccionarios prerrellenados de acuerdo con las técnicas de la invención.

La figura 10 es un diagrama de flujo que ilustra una técnica de descompresión LZ78 modificada de acuerdo con la invención.

La figura 11 es un ejemplo de compresión de texto que usa la técnica LZ78 modificada de la invención.

La figura 12 es un diagrama de flujo que ilustra una técnica de compresión híbrida LZ77/LZ78 de acuerdo con la invención.

La figura 13 es un diagrama de flujo que ilustra una técnica de descompresión híbrida LZ77/LZ78 de acuerdo con la invención.

La figura 14 es un ejemplo de compresión de texto que usa una combinación de las técnicas híbridas LZ77/LZ78 de acuerdo con la invención.

Descripción detallada de las realizaciones actualmente preferidas

La presente invención será descrita con detalle a continuación con respecto a las figuras 4 a la 14. Los que sean expertos en la técnica apreciarán que la descripción que se da en este documento es para propósitos explicativos solamente y que no está destinada a limitar el alcance de la invención. De acuerdo con esto, el alcance de la invención solamente está limitado por el alcance de las reivindicaciones anejas.

La extensión a los algoritmos Lempel - Ziv de acuerdo con la invención se puede usar junto con todas las variantes conocidas de las técnicas de compresión Lempel - Ziv descritas en la literatura de patentes así como en la literatura de compresión de texto. Sin embargo, en las realizaciones preferidas en la actualidad, la presente invención se usa como una modificación a las técnicas de compresión LZ77 o LZ78 o como una modificación a la variante Lempel - Ziv que por sí sola es una modificación o una extensión de las técnicas LZ77 o LZ78. La extensión a los algoritmos Lempel - Ziv de acuerdo con la invención se implementa de manera preferible como parte de un paquete de compresión software o como un módulo que se usa junto con los paquetes software preexistentes que se beneficiarían de la compresión de datos, tales como un paquete de procesado de texto comercial. En la figura 4 se muestra una configuración de sistema de muestra.

La presente invención se implementa de manera preferible como software que contiene instrucciones para controlar un procesador de un ordenador 1 de un usuario. En una realización preferida, la versión ejecutable del software que implementa el algoritmo de compresión Lempel - Ziv se almacena en un dispositivo de almacenamiento de programa fijo tal como un disco duro 2 que puede ser leído por un procesador (CPU) 3 de ordenador 1 con lo que el programa de instrucciones almacenado en el mismo para implementar el algoritmo de compresión Lempel - Ziv puede ser ejecutado por el procesador 3 para realizar la compresión de datos deseada. Como se muestra en la figura 4, la CPU 3 incluye una RAM 4 para almacenar los datos durante la ejecución del algoritmo de compresión Lempel - Ziv. En una realización preferida, los datos que se vayan a comprimir se proporcionan en formato digital sobre un medio que pueda ser leído

por un ordenador tal como un disco flexible 5, un disco óptico 5' o un CD ROM 5", o se reciben a través de un módem 6 y se cargan en la RAM 6. Por supuesto, los datos se pueden almacenar en el disco duro del ordenador 2 antes de ser cargados en la RAM 4 o se pueden proporcionar directamente a través de una conexión de red. La entrada de los datos dentro del ordenador 1 desde el usuario se proporciona a través de un teclado 7 y/o un ratón 8, y cualquier imagen gráfica acompañante se visualiza en una pantalla de vídeo 9 de ordenador.

De acuerdo con la invención, los diccionarios de compresión prerrellenados se almacenan en el disco duro 2 junto con la versión ejecutable del software que implementa el algoritmo de compresión Lempel - Ziv, y se cargan en la RAM 4 de la CPU 3 durante la ejecución del algoritmo de compresión de datos de Lempel - Ziv. Los datos que se vayan a comprimir se pueden almacenar como ficheros en el disco duro antes de ser comprimidos y después son restaurados localmente o se transmiten tras su compresión. En otras palabras, si la compresión de datos se está realizando de manera estricta para ahorrar espacio de almacenamiento local, los resultados del proceso de compresión se escriben en el disco duro 2 o en otra memoria local. De manera alternativa, los datos de entrada y los diccionarios de compresión prerrellenados pueden ser proporcionados sobre un medio que pueda ser leído por un ordenador tal como un disco flexible 5, un disco óptico 5', un CD ROM 5", recibidos a través de módem 6, o proporcionados directamente a través de una conexión de red. Por otra parte, si la compresión de datos se está realizando para permitir una transmisión de datos más rápida a través de una conexión punto a punto (por ejemplo, a través de módem 6), entonces la CPU 2 se conecta al módem 6 o a algún otro dispositivo de transmisión, y los datos comprimidos se transmiten a través de ese dispositivo de transmisión. Además, los datos comprimidos pueden ser devueltos al medio original legible por parte de un ordenador (por ejemplo, disco flexible 5, disco óptico 5' o CD ROM 5").

De acuerdo con esto, durante el funcionamiento del algoritmo de compresión de acuerdo con la invención, la RAM 4 típicamente recibe el código ejecutable, uno o más diccionarios de compresión prerrellenados y los datos de entrada provenientes del disco duro 2, de algún otro elemento de memoria tal como un disco flexible 5, disco óptico 5' o CD ROM 5", o a través de una conexión por módem, y los datos comprimidos se restauran después en el disco duro 2 o en algún otro elemento de memoria tal como un disco flexible 5, un disco óptico 5' o CD ROM 5", o se transmiten a través de módem 6 a otro ordenador para su almacenamiento y/o descompresión. Además cuando se usen con variantes del algoritmo LZ78, la RAM 4 de manera preferible asignará espacio para un diccionarios específico de datos del tipo desarrollados de manera convencional durante el funcionamiento del algoritmo LZ78 y sus variantes.

La figura 5 ilustra las opciones para el uso de los datos comprimidos generados de acuerdo con la invención. Como se muestra en la figura 5, los datos de entrada que se van a comprimir se introducen en el paso 10, se comprimen en el paso 12 usando la variante preferida del algoritmo de compresión de Lempel - Ziv, y los datos comprimidos se almacenan de manera temporal en el paso 14. Si se determina en el paso 16 que se está realizando la compresión para ahorrar espacio de almacenamiento local en el disco duro 2, CD RM 5" o similares, el fichero de datos comprimido se almacena localmente en el paso 18 y se sale de la rutina en el paso 20. Sin embargo, si los datos de entrada están siendo comprimidos para acelerar la transmisión a otro ordenador, los datos comprimidos se transfieren en el paso 22 a un dispositivo de transferencia de datos tal como un módem 6 y se transmiten a una nueva localización en el paso 24. Los datos comprimidos transmitidos se descomprimen después en el paso 26 y se almacenan en su formato descomprimido original en el paso 28. Se sale de la rutina en el paso 20. Como se describirá más adelante, puede que sea necesario transmitir el diccionario de compresión prerrellenado con los datos comprimidos o al menos para identificar el diccionario de compresión prerrellenado usado para la compresión de forma que se pueda usar el mismo diccionario de compresión prerrellenado en el receptor durante el paso 26 para descomprimir los datos comprimidos. Por supuesto, los datos comprimidos recibidos no necesitan ser descomprimidos de manera inmediata, sino que se pueden almacenar en su forma comprimida en el receptor hasta que se necesiten.

El proceso de prerrellenar un diccionario de compresión con palabras que sean significativas para el algoritmo de Lempel - Ziv se maneja de cualquiera de entre un número de maneras de acuerdo con la invención. Una posibilidad es que un usuario del software de compresión cree una lista de palabras frecuentemente usadas. Como este proceso tiene objeciones en la mayoría de las circunstancias, son preferibles procesos automáticos usados de acuerdo con la invención. Como en la Patente número 4.386.416, una posibilidad para dicho proceso automático es restringir el diccionario de compresión prerrellenado a lo que se hace referencia normalmente como palabras - esto es, secuencias de caracteres que están delimitadas por espacios, tabulaciones o signos de puntuación. Sin embargo, se prefiere que las estadísticas sean recogidas a partir de una muestra representativa de texto para determinar las secuencias más frecuentes de caracteres de varias longitudes. El número de entradas del diccionario de cada longitud se puede determinar por medio de una función de la cantidad de compresión que resultará del uso de una entrada de longitud particular (esto es, el número de bits originalmente necesarios para codificar una cadena de longitud particular menos el número de bits necesarios para transmitir una referencia a una entrada de diccionario, que no variará con la longitud de la entrada de diccionario, sino que más bien será una cantidad fija determinada a partir del tamaño del diccionario) y la frecuencia observada de esa secuencia dentro de la muestra de texto utilizado.

Por ejemplo, supóngase que una referencia de diccionario cuando se realiza la compresión cuesta 12 bits (esto es, el diccionario contiene como mucho 2^{12} ó 4096 entradas) y también supóngase una codificación de 7 bits del juego de caracteres ASCII, una secuencia de 10 caracteres necesitaría 70 bits de espacio de almacenamiento sin compresión, mientras que una secuencia de 9 caracteres requeriría 63 bits de espacio de almacenamiento sin compresión. Si en la muestra de texto de la que se derivaron las estadísticas de frecuencia, una secuencia particular de longitud 10 ocurre 60 veces, entonces los ahorros que resultarían si esa secuencia estuviese en el diccionario, serían 60 * 70 = 4200 (el coste de transmitir los datos en bruto) menos 60 * 12 = 720 (el coste de transmitir los punteros) lo que supone un

ahorro de 3480 bits. Suponiendo que la cadena de longitud 9 fue la más frecuente y que ocurrió 65 veces, los ahorros de coste asociados con tener esa secuencia de longitud 9 en el diccionario serían (65 * 63) - (65 * 12) = 3315 bits de ahorro. De esta manera, la secuencia de longitud 10 sería preferida como una entrada de diccionario sobre la secuencia de longitud 9.

El contenido de un diccionario prerrellenado puede ser seleccionado de manera eficiente de la siguiente manera. El texto de todos los documentos de los que aprenderá el diccionario prerrellenado se debería cargar en una estructura de datos que permita que todas las subsecuencias del texto sean identificadas de una manera fácil. Por lo general, se necesitará elegir una longitud máxima para las cadenas en el diccionario para evitar que el coste de la computación sea demasiado alto. Una longitud de 10 ó 15 caracteres probablemente sería suficiente, pero se puede determinar una longitud máxima apropiada de manera experimental, si se desea. Sin embargo, como no hay una longitud máxima particular especificada en el diseño del algoritmo de la invención, la elección de una longitud máxima se deja al usuario del software de construcción del diccionario. Como resultado de esto, la longitud máxima elegida puede que no dé como resultado la compresión óptima. Esto es porque existe un equilibrio entre la compresión y el coste de generar el diccionario prerrellenado, y el usuario del software de construcción del diccionario debe elegir una longitud máxima que permitirá construir el diccionario prerrellenado en una cantidad de tiempo razonable en su ordenador.

Se usarán una estructura de datos en árbol en la que cada nodo contenga una cadena de caracteres y una cuenta para identificar las cadenas que ocurren con frecuencia. La raíz de este árbol contiene la cadena vacía y el número total de caracteres en el cuerpo. Cada nodo hijo extenderá la cadena creada mediante la concatenación de todos los caracteres contenidos por los nodos en el camino desde la raíz del árbol hasta el nodo actual. De esta manera, todas las secuencias de caracteres hasta una longitud particular se pueden almacenar en un árbol que no sea más profundo que la longitud máxima elegida. Este árbol podría crecer para ser muy grande, con las hojas del árbol numerando el tamaño del alfabeto elevado a la potencia de la profundidad del árbol. Sin embargo, debido a la dispersión de los datos, el árbol debería ser en la práctica mucho más pequeño. Como apreciarán los que sean expertos en la técnica, los datos están dispersos como resultado de las propiedades del lenguaje humano con lo que los lenguajes no contienen, o al menos no contienen con frecuencia, todos los posibles n-gramas. Por ejemplo, la secuencia de letras “zqwvx” es extraordinariamente rara y poco probable que ocurra en los datos de formación si los datos están en idioma inglés.

Una vez que se haya rellenado la estructura de datos del árbol, se identificarán las secuencias que maximizan la fórmula de los ahorros de costes esbozada anteriormente. Se manejan las cadenas con solapamiento, ya que todas las subsecuencias de una secuencia identificada necesitan tener su cuenta en la estructura de datos de árbol actualizada mediante la resta del número de ocurrencias de la secuencia elegida para su entrada en el diccionario. Esto se puede ilustrar por medio de un ejemplo. Supóngase que una secuencia “ABC” que ocurra 10 veces en los datos es elegida como una entrada de diccionario. La cuenta asociada con el nodo para “ABC” en el árbol se decrementaría entonces en 10, como la cuenta para “BC”, la cuenta para “A”, la cuenta para “B” y la cuenta para “C”. Incluso el nodo asociado con la propia secuencia “ABC” se decrementaría en 10, eliminándolo de esta manera del árbol y de una futura consideración como una entrada de diccionario.

La figura 6 ilustra un diagrama de flujo del proceso de creación del diccionario de acuerdo con una realización preferida de la invención, con lo que la secuencia seleccionada para su entrada dentro del diccionario es la secuencia que maximiza la compresión. Como se ilustra en la figura 6, el diccionario de compresión prerrellenado se crea a partir de una muestra de texto mediante el cálculo del número de ocurrencias (estadísticas de frecuencia) de secuencias particulares de caracteres en la muestra en el paso 30 y después determinando en el paso 32 la compresión para cada secuencia de caracteres dado un tamaño fijo de diccionario, y por lo tanto, una longitud fija de puntero de entrada de diccionario. La muestra de texto puede ser una parte del texto que se vaya a comprimir, un documento similar o una colección de documentos similares del mismo género. Como en el ejemplo anterior, la compresión se calcula como: el número de ocurrencias de una secuencia particular según se determina en el paso 30 multiplicado por la diferencia entre la longitud sin codificar de esa secuencia particular y la longitud de puntero de entrada de diccionario. La compresión resultante para cada secuencia de carácter se usa después para determinar qué secuencias maximizan la compresión. En el paso 34 se selecciona y se añade al diccionario de compresión la secuencia de caracteres que maximiza la compresión. Según sea necesario, se actualizan las cuentas en la estructura de árbol de datos según se ha descrito con anterioridad. Las cuentas de caracteres para todas las subcadenas de la secuencia de caracteres que se acaban de añadir al diccionario de compresión en el paso 34, incluyendo la cuenta para la propia secuencia de caracteres, se modifican después en el paso 36. Mediante la modificación así de la cuenta de caracteres de la propia cadena, se evita la posterior inclusión de esa misma cadena en el diccionario de compresión. Si se determina en el paso 38 que el diccionario de compresión no está lleno, el proceso de selección de aquellas secuencias que producen la máxima compresión continúa hasta que se determine en el paso 38 que el diccionario de compresión está lleno con aquellas secuencias que producen la máxima compresión. La rutina finaliza después en el paso 40. Por supuesto, un diccionario de compresión formado de esta manera contendrá por lo general secuencias cortas que ocurren de manera muy frecuente así como aquellas secuencias más largas que ocurren con una frecuencia suficiente como para conducir a una compresión sustancial.

El espacio de direcciones para los diccionarios de manera preferible será compartido, de esta manera no se necesita enviar de manera explícita ningún indicador que indique qué diccionario se está usando. El espacio de direcciones se puede dividir entre el diccionario prerrellenado y el diccionario específico o adaptativo convencional de datos de cualquier forma vista ajustado por el constructor del diccionario de compresión. También, el tamaño global del diccionario de compresión combinado puede ser determinado por el constructor del diccionario de compresión. Sin

embargo, el tamaño debería ser una potencia par de 2, ya que los números de entrada al diccionario de por sí serán codificados con una representación binaria.

Los expertos en la técnica apreciarán que una muestra grande representativa de texto no comprimido puede incluir documentos que caigan de manera natural en diferentes géneros de texto, cada uno con su propia terminología o propiedades estadísticas. Los documentos de un género dado constituyen de esta manera una pequeña muestra de texto específica para ese género. Cada una de dichas muestras se puede usar para crear un diccionario de compresión prerrellenado independiente confeccionado para su género. De esta manera, cualquier combinación de las técnicas anteriores o de otras técnicas para derivar un diccionario de compresión se pueden aplicar a varios tipos o géneros diferentes de datos tales como el texto en inglés, texto en francés, programas de ordenador escritos en C, programas de ordenador escritos en Pascal, ficheros de bases de datos, imágenes, etc. Una vez que se hayan descubierto las "palabras" más frecuentes para cada tipo de datos, se puede crear un diccionario para cada tipo de datos. Este diccionario, junto con un diccionario inicialmente vacío, al que se añadirán nuevas "palabras", se usarán entonces para realizar una compresión de Lempel - Ziv usando técnicas convencionales. Por ejemplo, si se estuviese comprimiendo el texto de un periódico en inglés, el diccionario sería prerrellenado con las secuencias de letras, palabras y/o frases más frecuentes en inglés encontradas en una muestra de artículos de periódicos.

Con el fin de comprimir un documento de un género en particular, el ordenador de compresión identifica en primer lugar el género al que el documento es el más similar por medios automáticos (por ejemplo, en base a palabras clave o procedimientos de agrupamiento) y selecciona el diccionario de compresión prerrellenado apropiado para ese género. La información que identifica el diccionario de compresión prerrellenado seleccionado se anexa después al fichero de datos comprimidos. En el momento de la descompresión, el ordenador de descompresión examina la información anexada con el fin de determinar qué diccionario prerrellenado usar cuando se realice la descompresión de los datos recibidos.

Para que funcione este procedimiento como se desea, la colección de documentos de muestra debe ser particionada inicialmente en múltiples géneros. Esto se puede hacer por medio de cualquiera de varios procedimientos. Por ejemplo, una persona o varias personas pueden decidir la partición manual en base a criterios subjetivos u objetivos. De manera alternativa, un ordenador puede explorar de manera automática cada uno de los documentos en busca de palabras o de patrones ortográficos estadísticos que indiquen la pertenencia a un género en particular, tal como los géneros de "texto en español" o "programas de ordenador compilados". Además, si no se sabe nada acerca de la colección de documentos de muestra, la colección puede ser particionada de manera automática en un conjunto razonable de géneros mediante el uso de procedimientos de agrupación bien conocidos en la literatura.

De esta forma, el usuario del software compresión puede crear según desee diccionarios prerrellenados adicionales. Si el usuario de manera rutinaria comprime datos que tienen un vocabulario estándar de por sí, tales como manuales de ordenador o documentos de empresa, ambos haciendo un uso frecuente de palabras que no se encuentran por lo común en otros formatos de texto, el usuario puede aplicar el proceso de creación del diccionario a un gran cuerpo de este tipo de texto y crear un diccionario personalizado para su uso con estos tipos de documentos. Obviamente, para que la descompresión sea posible, el software que realiza la descompresión debe tener también acceso a este diccionario. De manera similar, si los datos no están siendo comprimidos para propósitos de archivo, sino que se están comprimiendo para una transmisión en línea, el destinatario de los datos comprimidos debe tener acceso a este diccionario hecho a medida. De esta manera, el proceso de creación del diccionario no debería ser emprendido de una manera frecuente, ya que requeriría la distribución y el almacenamiento de un gran número de diccionarios y eliminaría o al menos reduciría el beneficio añadido de usar un diccionario más apropiado para propósitos de compresión.

Durante el proceso de compresión, el diccionario prerrellenado que es probable que consiga el nivel más alto de compresión, se puede determinar mediante la realización de una compresión sobre un subconjunto pequeño de todo el documento que se vaya a comprimir usando cada uno de los diccionarios prerrellenados. En otras palabras, los primeros N caracteres de datos del texto que se vaya a comprimir (N es arbitrario y se puede elegir por adelantado o se puede calcular como un porcentaje de la longitud del documento, sometido a un máximo de forma que la identificación del diccionario no consuma demasiado tiempo) se comprimen usando cada uno de los diccionarios prerrellenados supuestos. La compresión que usa cada uno de los diccionarios prerrellenados se calcula y el diccionario prerrellenado que maximice la compresión es el que se selecciona para la compresión de todo el texto. De esta manera, se puede usar el diccionario más apropiado para el texto particular que se vaya a comprimir. Además, el usuario puede elegir un diccionario de manera manual si está al tanto de que el documento que se va a comprimir es típico de la clase de documentos representados por un diccionario prerrellenado particular. Esto evitaría que el software experimente con varios diccionarios de compresión y daría como resultado un ahorro de tiempo durante el proceso de compresión. De manera adicional, la selección manual puede ser necesaria para permitir la transmisión de un documento a otro usuario que no posea todos los diccionarios prerrellenados que posee el remitente o en el caso de compresión para propósitos de archivo, de forma que el documento se pueda descomprimir en un momento posterior por medio de software que no esté al corriente de todos los diccionarios de compresión disponibles, tal como el software de un emplazamiento remoto.

Sin embargo, la opción de selección manual del diccionario prerrellenado no se encuentra dentro del alcance del la invención según se reivindica.

Además, si se determina durante la compresión del subconjunto de caracteres en los datos que se vayan a comprimir, que comenzando con un diccionario de compresión de datos completamente vacío se permitiría la mayoría de la compresión, entonces el diccionario de compresión prerrellenado se puede eliminar de la codificación de esos datos particulares.

Durante el proceso de descompresión, se hacen las mismas modificaciones al diccionario específico de datos que las que se hicieron al diccionario específico de datos construido durante la compresión. De esta manera, se pueden ampliar de una manera apropiada las referencias al diccionario específico de datos dentro de los datos comprimidos en el texto reconstituido creado por el sistema de descompresión de datos. La identidad del diccionario prerrellenado que se esté usando para la compresión se almacena de manera preferible en los datos comprimidos de forma que el módulo de descompresión sea capaz de identificar qué diccionario prerrellenado usar durante la descompresión. De manera similar, se puede almacenar con los datos comprimidos una indicación de qué variante del algoritmo de Lempel - Ziv usada para realizar la compresión de los datos, así como una indicación de qué política se usó durante la compresión para nivelar las entradas de los dos diccionarios. Además, se puede almacenar en los datos comprimidos una indicación de cómo se asigna el espacio de direcciones del diccionario a los diccionarios implicados en la compresión - esto es, cuáles son las direcciones de inicio y de finalización para el diccionario específico de datos así como las direcciones de inicio y de finalización para uno o más diccionarios prerrellenados usados en la compresión. También, si se transmiten los datos comprimidos a un emplazamiento remoto, puede que sea necesario enviar el diccionario de compresión con el fin de permitir la descompresión en el emplazamiento remoto. Sin embargo, esto puede frustrar el propósito de la compresión a menos que los ahorros de tiempo sean mayores que el tiempo que se necesita para transmitir el diccionario. Por otra parte, incluso si los ahorros de tiempo no sobrepasan el coste de transmisión del diccionario, podría ser inteligente el transmitir el diccionario si es probable que se transmitan otros documentos que usan este diccionario. De acuerdo con esto, cuando los datos comprimidos se vayan a transmitir a un emplazamiento remoto, por lo general es deseable usar solamente diccionarios de compresión prerrellenados que se encuentren disponibles en el emplazamiento de destino.

Sin embargo, si el remitente determina que el diccionario prerrellenado óptimo es el diccionario A, pero comprime con el diccionario B en su lugar porque el receptor no tiene el diccionario A, entonces el remitente podría transmitir también un mensaje al receptor sugiriéndole que el receptor consiga el diccionario A para su uso futuro. Una vez que el receptor haya recibido suficientes mensajes de este tipo, el receptor finalmente solicitará una copia del diccionario A al remitente o a otro servidor en el que se encuentre disponible el diccionario A. El receptor pone entonces la copia del diccionario A en un almacenamiento local a largo plazo o a medio plazo.

Como otra opción, el remitente puede comprimir el fichero de datos usando un diccionario prerrellenado A sin considerar si el receptor tiene el diccionario A. Entonces, siempre que el receptor descomprima el fichero la primera vez, necesitará una copia del diccionario A. Si el diccionario A no está en ese momento almacenado en el receptor, el receptor obtendrá una copia a través de la red. El receptor pone entonces una copia del diccionario A en un almacenamiento a largo plazo o a medio plazo para su uso en el futuro, además de usarlo para descomprimir el fichero.

Por otra parte, puede que ocurra que uno pueda desear comprimir una gran colección de documentos de una vez, por ejemplo, para propósitos de archivo. En lugar de comprimir los documentos por separado, es deseable explotar las similitudes entre los distintos documentos, no solamente las similitudes dentro de cada documento o su similitud con un cuerpo preseleccionado. En otras palabras, si algunos de los documentos están relacionados unos con otros de alguna manera, muchos de ellos contendrán cadenas similares que se pueden explotar para proporcionar más compresión.

Por ejemplo, supóngase que el documento A fue comprimido de acuerdo con la invención usando un diccionario D prerrellenado con palabras comunes del idioma inglés. Al final de la compresión del documento A, un diccionario ampliado D' permanece conteniendo las entradas de D así como cadenas que aparecieron en el documento A. Otro documento B se puede comprimir ahora usando el diccionario D' en lugar del diccionario prerrellenado D (o algún compromiso entre el diccionario D y el diccionario D') como el diccionario de inicio. Si el documento B es completamente similar al documento A, se ha descubierto que esta técnica proporciona una compresión superior.

Sin embargo, en el caso general en el que se desea comprimir un gran número de documentos, cualquier documento comprimido B tendría que especificar el diccionario de inicio para usar cuando se haga la descompresión. Puede especificar uno de los diccionarios prerrellenados almacenados o puede nombrar otro documento comprimido, el documento A, y especificar que el diccionario final que resulta de la descompresión del documento A será usado como el diccionario de inicio para la descompresión del documento B.

Las técnicas de agrupamiento se pueden usar para determinar qué documentos dependen de los diccionarios finales de otros documentos. Los documentos que se determinen a través del agrupamiento que son muy similares unos con otros se comprimen usando los diccionarios de compresión unos de otros. La información acerca de qué documentos dependen unos de otros se especifica en la cabecera del documento cuando se comprime de manera que el software de descompresión descomprima de manera correcta el documento una vez que todos los documentos de los que depende se hayan descomprimido.

De esta manera, en general, se pueden mejorar las velocidades de compresión para ciertos datos teniendo muchos diccionarios de compresión disponibles, cada uno de ellos confeccionado para un tipo de texto altamente específico.

Sin embargo, existe un coste asociado con el almacenamiento, y a veces con la transmisión, de tantos diccionarios. Este coste se puede minimizar mediante el reconocimiento de que los diferentes diccionarios codifican muchas de las mismas cadenas y de esta forma se solapan en contenidos hasta un grado sustancial. Esta característica de la codificación de diccionario se puede tomar como una ventaja, en que se pueden crear de manera simultánea diccionarios prerrellenados para una pluralidad de muestras de texto específico a partir de un único cuerpo grande de documentos. En este ejemplo, se crearía una representación combinada especial para los múltiples diccionarios que ocuparía menos espacio que el espacio necesario para almacenar de manera independiente todos los diccionarios específicos de género diferentes. Resumiendo, se ahorra espacio de memoria mediante la clasificación de una cadena que aparezca en múltiples diccionarios solamente una vez en lugar de clasificar de manera independiente la entrada en cada diccionario respectivo.

Los expertos en la técnica apreciarán que, hasta un cierto grado, la invención equilibra la eficiencia de la compresión con la compactación del diccionario. Para ilustrar esto, se puede definir un parámetro que indica la importancia relativa de la eficiencia de la compresión y la compactación del diccionario. Cuando el parámetro se fija a 1, se obtiene la máxima eficiencia de compresión. En otras palabras, cuando el parámetro se fija a 1, se construye de manera independiente un diccionario especializado para cada género y los diccionarios para cada género solamente se combinan para eliminar el almacenamiento redundante de duplicados como se ha descrito con anterioridad. Por otra parte, el se reduce el parámetro en alguna cantidad, entonces los diccionarios de género no se construyen completamente independientes; se construyen deliberadamente para mejorar su solapamiento. Si el parámetro se reduce hasta 0, entonces se obtiene el máximo solapamiento mediante el uso del mismo diccionario para cada género. La configuración del parámetro a 0 produce de esta manera el diccionario combinado más pequeño, pero no aprovecha en modo alguno la ventaja de las propiedades específicas del género. En la actualidad, los inventores creen que como mejor funcionará la técnica de la invención será cuando se hayan agrupado los muchos géneros especializados juntos dos o tres a la vez en "súper-géneros" jerárquicos análogos al sistema de clasificación "Decimal Dewey" para documentos de muestra. Dichas técnicas de clasificación jerárquica deben llevarse a cabo, en caso de que sean necesarias, por medio de procedimientos de agrupación automática bien conocidos.

Ahora se describe una realización preferida de la invención con respecto a las figuras 7 a la 11, como una extensión de LZ78. En el algoritmo LZ78 de la realización preferida, el texto se comprime como se muestra en la figura 2, usando un diccionario de compresión que se expande durante el proceso de compresión con lo que se transmiten la referencias al diccionario de compresión en lugar del texto original. A diferencia de LZ78 convencional, sin embargo, el diccionario de compresión usado de acuerdo con la invención no comienza con sólo la cadena vacía. En lugar de esto, se inicializa para que contenga no solamente la cadena vacía sino también el contenido de un diccionario prerrellenado o varios diccionarios prerrellenados creados de una de las maneras descritas con anterioridad. El diccionario prerrellenado también puede contener el contenido entero del alfabeto sobre el que se esté realizando la compresión. También, el diccionario de compresión puede ser considerado como dos diccionarios independientes que compartan un espacio de direcciones común o como un único diccionario más grande, dependiendo de las especificidades de la realización de la técnica. De esta manera, la técnica usada de acuerdo con la invención comienza con un diccionario prerrellenado o con varios diccionarios prerrellenados, así como el diccionario específico de datos convencional que contiene solamente la cadena vacía a la que se añaden entradas adicionales. Sin embargo, si todos los caracteres del alfabeto se han añadido al diccionario prerrellenado, como en LZMW, entonces el diccionario específico de datos debería estar vacío (ya que nunca es necesario transmitir la cadena vacía seguida por su extensión).

El espacio de diccionario prerrellenado puede ser considerado como dos diccionarios separados que comparten el mismo espacio de direcciones o como uno o más diccionarios prerrellenados independientes que puede o no estar concatenados para formar un diccionario prerrellenado mayor antes del inicio de la compresión. Cuando se encuentra una secuencia de caracteres que va a ser comprimida, se encuentra la coincidencia más larga (LM) dentro del diccionario prerrellenado o de los diccionarios prerrellenados o del diccionario específico de datos o en una tabla de consulta o usando una estructura de datos de árbol. Una vez que se ha encontrado esta coincidencia, entonces se transmite un número de entrada de diccionario y el carácter que sigue a esta secuencia. Después, se añade la concatenación de la entrada transmitida de diccionario y el carácter transmitido al diccionario específico de datos. Este proceso se repite hasta que se agoten los datos que se vayan a comprimir.

Como con LZ78, el diccionario prerrellenado tiene un tamaño fijo, que determina el tamaño del número de entrada de diccionario que se necesita transmitir para efectuar la compresión. Una vez que se ha llenado el diccionario prerrellenado, no se permiten entradas adicionales al mismo. Sin embargo, se pueden usar cualquiera de las modificaciones de manejo de diccionario esbozadas en la literatura. Por ejemplo, se podría usar un algoritmo usado menos recientemente para descartar entradas de diccionario cuando se necesiten añadir nuevas entradas. De manera alternativa, se podría supervisar el funcionamiento y el diccionario específico de datos podría ser reinicializado cuando la relación de compresión caiga por debajo de un cierto umbral o se desvíe de la relación de compresión conseguida para las secciones anteriores de los datos que se estén comprimiendo.

Las figuras 7 y 8 ilustran una realización preferida para realizar la compresión de datos LZ78 de acuerdo con la invención. Como se muestra en la figura 7, los datos de entrada que se vayan a comprimir se entregan a un módulo software 42 de compresión de datos que puede contener cualquiera de las variantes del algoritmo LZ78 conocidas. Como se ha hecho notar con anterioridad, el diccionario 44 puede ser considerado como que contiene dos diccionarios separados - un diccionario específico de datos LZ78 convencional 46 y un diccionario prerrellenado 48, respectivamente, ambos compartiendo el mismo bus de direcciones común 50. También, se puede usar una pluralidad de diccionarios

prerrellenados 52 con el fin de encontrar el diccionario prerrellenado que proporcione la mayor compresión para los datos de entrada particulares. Todos los elementos activos de la figura 7 de manera típica serían llevados dentro de la RAM 4 para el procesado por medio de la CPU 3 (figura 4) durante el funcionamiento.

La figura 8 es un diagrama de flujo de software para implementar la técnica de compresión LZ78 modificada en el módulo de compresión de datos LZ78 42 de acuerdo con la invención. Como se muestra, la rutina empieza en el paso 56 mediante la comprobación en el paso 58 de si el flujo de datos de entrada está vacío. Si está vacío, entonces todos los datos de entrada se han comprimido y la rutina sale en el paso 60. Sin embargo, si el flujo de datos no está vacío, y así existen más datos de entrada para comprimir, se busca en el diccionario específico de datos 46 en el paso 62 la coincidencia más larga(LM) con la secuencia de datos que sigue al puntero actual. La coincidencia más larga se almacena en el módulo de compresión de datos LZ78 42 como DSLM. Se busca después en diccionario prerrellenado 48 y/o en los diccionarios prerrellenados 52 en el paso 64 la coincidencia más larga (LM) con la secuencia de datos que sigue al puntero actual. La coincidencia más larga se almacena en el módulo de compresión de datos LZ78 42 como PFLM. En el paso 66, la CPU 3 determina entonces si la longitud de la coincidencia más larga encontrada en el diccionario específico de datos 46 es más larga que la longitud de la coincidencia más larga encontrada en el diccionario prerrellenado 48 (y en otros diccionarios prerrellenados 52 si se usan más de uno). Si la longitud de DSLM es mayor que la longitud PFML, entonces la coincidencia más larga se fija a DSLM en el paso 68; en caso contrario, la coincidencia más larga se fija a PFLM en el paso 70. (Si las longitudes DSLM y PFLM son las mismas, entonces se puede usar el mismo diccionario usado para la secuencia de datos de entrada anterior, aunque puede ser deseable asignar simplemente uno de los diccionarios cuando las longitudes sean las mismas para evitar la necesidad de un indicador para conservar la pista o cuál fue el último diccionario usado).

Una vez que se ha determinado la LM en los pasos 66 al 70, se identifica el carácter que sigue a la LM en el flujo de datos de entrada en el paso 72 y se almacena como una extensión (C) en el módulo de compresión de datos LZ78. El número de entrada de diccionario de LM (y qué diccionario) se transmiten/almacenan después como se desee en el paso 74. De manera similar, la extensión (C) se y transmite/almacena según se desee en el paso 74. Después, en el paso 76, se concatena la extensión (C) a la coincidencia más larga (LM) y se almacena en el diccionario específico de datos 46 a menos que ya esté lleno. Por supuesto, también se pueden usar las técnicas de actualización de la técnica primera para permitir insertar la última entrada en el diccionario específico de datos 46 en lugar de, por ejemplo, la entrada a la que se accedió menos recientemente. En el paso 78, el puntero actual a los datos de entrada se mueve entonces en la longitud de la coincidencia más larga más uno (por la extensión). El control vuelve entonces al paso 58 y el proceso de compresión se repite para la siguiente secuencia de datos de entrada hasta que se agote el flujo de datos y se haya completado el proceso de compresión.

El procedimiento de la figura 8 se puede modificar para tener en cuenta el ahorro de costes asociado con la compresión de una secuencia de datos usando el diccionario específico de datos con LZ78 o la ventana en LZ77 o haciendo referencia al diccionario prerrellenado. En LZ78, el coste de cada aproximación será probablemente idéntica, pero cuando se amplía LZ77 para incluir diccionarios prerrellenados de acuerdo con la invención, el coste de comprimir usando las dos aproximaciones puede diferenciarse y puede ser ventajoso elegir un procedimiento sobre el otro en el caso de que sea posible que la compresión use ambas técnicas.

La descompresión de una secuencia de datos comprimida LZ78 de acuerdo con la invención se realiza sustancialmente de la misma manera que con las técnicas de descompresión LZ78 convencionales. Todas las modificaciones al diccionario de compresión hechas durante la compresión se hacen también durante la descompresión. Esto limita algo la utilidad del algoritmo, como es el caso con LZ78 convencional, ya que la descompresión de una parte de datos requiere que la parte anterior sea descomprimida en su totalidad. Cuando se realice la descompresión, la lista de número de entrada de diccionario y pares de caracteres se procesa un par cada vez hasta que se agota la lista. Por lo general, se consulta cada entrada de diccionario dentro del diccionario y el texto al que se refiere se visualiza seguido por el carácter en su entrada de diccionario y par de caracteres. De esta manera, se consigue una compresión/descompresión sin pérdidas.

Las figuras 9 y 10 ilustran una realización preferida para realizar la descompresión de datos LZ78 de acuerdo con la invención. Como se muestra en la figura 9, los datos comprimidos se entregan a un módulo software 80 de descompresión de datos LZ78, que puede contener cualquiera de las variantes conocidas del algoritmo LZ78. Durante el funcionamiento, el módulo de descompresión de datos LZ78 80 consulta en el diccionario la entrada de los datos comprimidos de entrada en el diccionario 82, que incluye un diccionario prerrellenado 84 y un diccionario específico de datos convencional 86, usando el número de entrada de diccionario que identifica la LM. La entrada resultante de diccionario se almacena entonces en el módulo de descompresión de datos LZ78 80 como la entrada E. El carácter que sigue al número de entrada de diccionario en el flujo de datos de entrada comprimido se almacena entonces en el módulo de descompresión de datos LZ78 80 como C. C y E son concatenados después por la CPU 3 y se insertan dentro del diccionario específico de datos 86, a menos que esté lleno, y después se saca como los datos reconstituidos, descomprimidos (originales). Los datos concatenados se insertan en el diccionario específico de datos 86 usando las mismas convenciones para las que se insertaron los datos en el diccionario específico de datos 48 durante la compresión. En otras palabras, el código de diccionario asignado o reasignado a los datos concatenados se determina de la misma manera que se determinó el código de diccionario asignado durante la compresión.

Los expertos en la técnica apreciarán que el número de entrada de diccionario se asigna de manera diferente durante la descompresión dependiendo de la variación de la compresión usada. Si se usa una compresión convencional Lempel

- Ziv, entonces el número de entrada de diccionario asignado será el siguiente en la secuencia (es decir, el primero será numerado 0, el siguiente 1, el siguiente 2, etc.). Como la compresión Lempel -Ziv no hace ninguna provisión para realizar ninguna función especial en el diccionario, entonces el diccionario se reinicializará y los números de diccionario volverán a cero cuando el diccionario esté lleno, o no se permitirá que el diccionario crezca más. Por otra parte, si se usa más de un esquema complicado para gestionar el diccionario, como "menos recientemente usada" (LRU), entonces, el número de entrada de diccionario asignado será el mismo que el que se acaba de describir hasta que el diccionario se llene por completo. En este momento, el algoritmo LRU empezará a actuar y se asignará el número de entrada de diccionario de la entrada de diccionario menos recientemente usada. La entrada de diccionario anteriormente asociada con el número de entrada de diccionario será eliminada del diccionario.

Como con el diccionario prerrellenado 48 en el lado de compresión, el diccionario 82 puede ser considerado como dos diccionarios independientes que comparten el mismo bus de direcciones común 88. También, se puede usar una pluralidad de diccionarios prerrellenados 90 durante la descompresión si se usaron una pluralidad de tales diccionarios prerrellenados durante la compresión. Todos los elementos activos de la figura 9 típicamente se llevarán a la RAM 4 para el procesado por la CPU 3 (figura 4) durante el funcionamiento.

La figura 10 es un diagrama de flujo del software para implementar la técnica de descompresión LZ78 modificada de acuerdo con la invención. Como se muestra, la rutina comienza en el paso 94 con la comprobación en el paso 96 de si el flujo de datos comprimidos de entrada está vacío. Si está vacío, entonces se han descomprimido todos los datos de entrada y la rutina sale en el paso 98. Sin embargo, si el flujo de datos comprimidos no está vacío, y así existen más datos de entrada para descomprimir, se consulta en el paso 100 la entrada de diccionario correspondiente al puntero actual usando el número de entrada de diccionario. La correspondiente entrada de diccionario encontrada de esta forma se almacena después en el módulo de descompresión de datos LZ78 80 como E. Si se desea, los datos descomprimidos también se visualizan al usuario. En el paso 102, el carácter en el flujo de datos comprimidos de entrada que sigue al número de entrada de diccionario actual se almacena entonces en el módulo de descompresión de datos LZ78 80 como C. Si se desea, ese carácter también se visualiza al usuario. En el paso 104, el carácter C y la entrada de diccionario E se concatenan y se almacenan en el módulo de descompresión de datos LZ78 80 como ADD. Una vez más, si se desea, la concatenación resultante se visualiza también al usuario. Después, en el paso 106, la concatenación "ADD" se inserta en el diccionario específico de datos 84 a menos que ya esté lleno. Por supuesto, también se pueden usar las técnicas de actualización de la técnica anterior para permitir que la última entrada sea insertada en el diccionario específico de datos 80 en lugar de, por ejemplo, la entrada a la que se accedió menos recientemente. Finalmente, en el paso 108, el puntero actual a los datos de entrada se avanza entonces al inicio del siguiente número de entrada de diccionario en el flujo de datos de entrada comprimido de entrada. El control vuelve entonces al paso 96, y el proceso de descompresión se repite para la siguiente secuencia de datos de entrada hasta que se agote el flujo de datos comprimidos de entrada. El proceso de descompresión se completa entonces.

La figura 11 es un ejemplo de compresión de texto usando la realización LZ78 de la invención descrita con respecto a las figuras 7 a la 10. Como en el ejemplo de la figura 2, el alfabeto se supone que contiene solamente 3 caracteres (A, B, C) y los datos que se van a comprimir son los mismos que los de la figura 2 para propósitos de ilustración. Como se muestra, el diccionario de compresión contiene inicialmente los caracteres del alfabeto así como secuencias frecuentes, donde las secuencias frecuentes se determinaron a partir de una muestra de texto, de texto similar anteriormente almacenado, de la entrada de usuario o de cualquiera de otras técnicas basadas en la estadística que serían aparentes para los que sean expertos en la técnica. El puntero al carácter en el que comenzará la compresión se coloca en el primer carácter de la secuencia de texto que vaya a ser comprimido. Como en el ejemplo de la figura 2, se encuentra la coincidencia más larga dentro del diccionario y se transmite o se almacena localmente, dependiendo de la aplicación del algoritmo de compresión, un puntero a esta entrada. La coincidencia más larga dentro del diccionario se puede encontrar construyendo una estructura de árbol que contenga todas las entradas de diccionario idénticas al tipo de estructura de árbol anteriormente descrita para identificar cadenas que se vayan a insertar en el diccionario prerrellenado. Otra posibilidad es usar una tabla de consulta o una función de cálculo de clave. En cualquier caso, el puntero se mueve entonces el número de caracteres transmitido.

Como se muestra en la figura 11, el diccionario se actualiza para que contenga la concatenación de las dos entradas de diccionario anteriormente transmitidas. En el primer paso, no hubo transmisión previa, por lo que no se añade nada al diccionario. En la segunda iteración a través del ciclo de procesado, se encuentra la coincidencia más larga, se transmite su número de entrada de diccionario, se mueve el puntero el número apropiado de caracteres y se añade al diccionario como una nueva entrada la concatenación de las dos entradas de diccionario anteriores transmitidas. En este caso, se añade "ABC". Este proceso se repite entonces en el tercer paso, y se añade la nueva entrada "ABB" al diccionario. Este proceso se repite hasta que los datos que se vayan a comprimir se hayan agotado, que en este ejemplo son solamente tres pasos en contraposición a los 6 pasos usados en la técnica LZ78 de la técnica anterior de la figura 2 para la misma secuencia de entrada. De esta forma, un diccionario prerrellenado que contenga las secuencias que ocurren con frecuencia de acuerdo con la invención puede proporcionar una mejora sustancial en las eficiencias de codificación/descodificación de los algoritmos de compresión LZ78 convencionales.

Por supuesto, como apreciarán los que sean expertos en la técnica, un diccionario prerrellenado mejorará también el funcionamiento de compresión de las variantes de compresión de datos LZ77 también. Sin embargo, para comprender cómo se puede usar el diccionario prerrellenado de la invención con las variantes de compresión de datos LZ77, uno debe recordar las diferencias entre las técnicas de compresión LZ77 y LZ78 notadas anteriormente. Como se describe con respecto a la figura 1, la técnica LZ77, en lugar de construir un diccionario de manera explícita, como con LZ78,

retiene una “ventana” de texto que consiste en una parte de la parte ya comprimida del documento. En cada uno de los pasos, el sistema identifica la subcadena más larga de la ventana que aparece como un prefijo del texto restante que va a ser comprimido. El sistema codifica este prefijo mediante la transmisión de un puntero dentro de la ventana, que especifica o la posición en la ventana en la que comienza la copia del prefijo o la posición en la ventana en la que acaba la copia del prefijo, a la elección del usuario, y después el sistema transmite la longitud del prefijo y quizá una extensión. Sin embargo, si el prefijo seleccionado está vacío, el sistema en su lugar transmite un código de escape seguido por el siguiente carácter literal que se vaya a comprimir. La ventana se actualiza para incluir el texto nuevamente comprimido, y la compresión continúa con la parte del documento que sigue inmediatamente al prefijo. La técnica LZ77 no requiere que se transmita un carácter literal junto con cada prefijo que se transmita. Como resultado de esto, las variantes de la técnica LZ77 son por lo general más eficientes que la técnica LZ78 y se usan en muchos programas de compresión populares tales como gzip y PKZIP.

Se puede incorporar un diccionario prerrellenado de acuerdo con la invención en la técnica LZ77 de cualquiera de un número de maneras. Primera, y la más simple, se puede crear un texto estándar que contenga muchas cadenas comunes, y este texto se puede conservar anexado al comienzo de la ventana durante toda la compresión de cualquier documento. Se debería notar que el texto anexado al comienzo no es realmente comprimido y transmitido, sino que ya que está en la ventana, el software de compresión puede hacer referencia al mismo de la misma manera que hace referencia al texto que se comprimió y se transmitió recientemente. De esta manera, al comienzo del proceso de compresión, cuando no hay texto en cualquier otro caso disponible para servir como la ventana, el algoritmo de compresión puede transmitir aún punteros dentro del texto que contiene cadenas comunes. Esto mejora el funcionamiento de la compresión, especialmente al comienzo de la compresión, donde la técnica LZ77 estándar debe transmitir muchos caracteres literales u otra clase de cadenas debido a la carencia de texto suficiente en la ventana. A medida que progresa la compresión, el texto que contiene cadenas comunes puede ser opcionalmente acortado de manera gradual para permitir que haya más sitio en la ventana para el texto que proviene del documento que se está comprimiendo. La relación óptima de texto de diccionario prerrellenado respecto a texto comprimido se puede determinar de manera experimental y por lo general variará de un género de documento a otro. En LZ77, la ventana de texto de tamaño fijado se usa por lo general para dicha compresión.

Otra técnica preferida, pero más complicada, implica el tratamiento del conjunto de punteros usados por los algoritmos de compresión como punteros divididos, quizá de manera impar, en dos clases. Los punteros de la primera clase hacen referencia a las posiciones en la ventana, que consta de parte de una porción del documento ya comprimida, como en el estándar LZ77. Los punteros de la segunda clase hacen referencia a las entradas en un diccionario que listan las cadenas que ocurren de manera frecuente. Este procedimiento híbrido combina en efecto algunas de las propiedades de LZ77 y LZ78. En cada uno de los pasos, el sistema selecciona, codifica y transmite un prefijo del resto de los documentos por medio de uno de los dos procedimientos. Puede seguir el procedimiento LZ77, transmitiendo un puntero a una subcadena en la ventana, seguido por la longitud de esta subcadena, o cuando sea necesario, la transmisión de un código de escape seguido por un carácter literal. De manera alternativa, el sistema puede seguir el procedimiento LZ78, transmitiendo un puntero a la entrada más larga del diccionario que sea un prefijo del texto restante que vaya a ser comprimido. El último procedimiento se usa en cualquiera de los pasos en los que exista dicha entrada de diccionario, con tal de que el procedimiento consiga una mejor compresión (medida por la relación de bits transmitidos respecto a la longitud de texto) en ese paso que la compresión conseguida por el procedimiento LZ77. Una ventaja del procedimiento híbrido sobre el procedimiento LZ77 es que no se necesita transmitir ninguna longitud: cada entrada de diccionario tiene una longitud fija que está permanentemente almacenada (o en cualquier otro caso, registrada) en el diccionario. El procedimiento híbrido también es más eficiente que el procedimiento LZ78 en que las entradas de diccionario no están acompañadas por caracteres literales.

La figura 12 es un diagrama de flujo de software para implementar dicha técnica de compresión híbrida LZ77/LZ78. En una realización preferida en la actualidad, se implementa un sistema tal por medio de la modificación respectivamente del módulo de compresión de datos LZ78 42 de la figura 7 y el módulo de compresión de datos LZ78 80 de la figura 9 para que contengan el software de compresión y de descompresión LZ77. Como se muestra, la rutina híbrida resultante comienza en el paso 110 por medio de la comprobación en el paso 112 de si el flujo de datos de entrada está vacío. Si está vacío, entonces todos los datos de entrada se han comprimido y la rutina sale en el paso 114. Sin embargo, si el flujo de datos no está vacío, y de esta forma no hay más datos de entrada para comprimir, se busca en el diccionario prerrellenado en el paso 116 la coincidencia más larga (LM) con la secuencia de datos que sigue al puntero actual. La coincidencia más larga encontrada (y de manera opcional la extensión de carácter EXT) se almacena en el módulo de compresión de datos LZ77/LZ78 como PDLM (y EXT). Se busca entonces en la ventana anterior de texto la coincidencia más larga (LM) con la secuencia de datos que sigue al puntero actual. La coincidencia más larga encontrada se almacena en el módulo de compresión de datos LZ77/LZ78 como WLM. En el paso 120, la CPU 3 determina entonces si se encontró una coincidencia en el paso 118, y de haberse encontrado, se identifica de manera opcional en el paso 122 (si se desea la opción de extensión) el carácter siguiente a la coincidencia más larga WLM en el texto que se vaya a comprimir (EXT). Por otra parte, si la CPU 3 determina en el paso 120 que no se encontró ninguna coincidencia en WLM, se usa un código de escape en el paso 124 en lugar de un puntero y longitud para identificar el carácter literal EXT apuntado por el puntero actual.

En el paso 126, el coste de transmitir PDLM como un número de entrada de diccionario se compara con el coste de transmitir WLM como un puntero y una longitud, y de manera opcional, como un carácter de extensión EXT, o como un literal. De manera específica, en el paso 126 se determina qué técnica ahorra más en comparación con la transmisión de datos sin comprimir. Por lo general, esto se puede llevar a cabo mediante el cálculo de la relación

de compresión para cada técnica. En base a los resultados del paso 126, se transmite/se almacena según se desee en el paso 128 la técnica que ahorre más de PDLM y WLM (con o sin EXT). Finalmente, en el paso 130, el puntero actual a los datos de entrada se mueve en la longitud de la coincidencia más larga o la longitud de la coincidencia más larga más uno (si se usó una extensión EXT). El control vuelve después al paso 112 y se repite la compresión para la siguiente secuencia de datos de entrada en el puntero actual hasta que se agote el flujo de datos y se complete el proceso de compresión.

La figura 13 es un diagrama de flujo de software para implementar una técnica de descompresión híbrida LZ77/LZ78 de acuerdo con la invención. Como se muestra, la rutina comienza en el paso 132 por medio de la comprobación en el paso 134 de si el flujo de datos comprimidos de entrada está vacío. Si está vacío, entonces se han descomprimido todos los datos de entrada y la rutina sale en el paso 136. Sin embargo, si el flujo de datos comprimidos no está vacío, y de esta forma existen más datos de entrada para descomprimir, se comprueba la entrada correspondiente al puntero actual en el paso 138 para determinar si es una PDLM o una WLM. Por ejemplo, se puede determinar si la referencia es una referencia PDLM o una referencia WLM mediante la comprobación de cómo está dividido el espacio de direcciones. Si la entrada en el flujo de datos comprimidos es una PDLM, entonces se siguen las técnicas LZ78 descritas con respecto a la figura 10. En particular, la entrada de diccionario en el diccionario prerrellenado correspondiente al puntero actual se consulta en el paso 140 usando el número de entrada de diccionario. La correspondiente entrada de diccionario encontrada así se almacena después en un módulo de descompresión LZ77/LZ78 como E. Si se desea, también se visualizan al usuario los datos descomprimidos. En el paso 142, si se envió la extensión opcional EXT (como se determinó antes de ejecutar el algoritmo) el carácter en el flujo de datos comprimidos de entrada que sigue al número de entrada de diccionario actual se almacena entonces en el módulo de descompresión LZ77/LZ78 como C. Si se desea, ese carácter también se visualiza al usuario. En el paso 144, el carácter C y la entrada de diccionario E se concatenan y se almacenan en el módulo de descompresión de datos LZ77/LZ78 como ADD. Una vez más, si se desea, la concatenación resultante también se visualiza al usuario.

Por otra parte, si se determina en el paso 138 que la entrada en el flujo de datos comprimidos es una WLM, entonces la entrada en el puntero se usa en el paso 146 para consultar la entrada en la ventana anterior del texto apuntado por esa entrada. Este valor se almacena en el módulo de descompresión LZ77/LZ78 como W. Si se desea, los datos descomprimidos también se visualizan al usuario. En el paso 148, si se envió la extensión opcional EXT (como se determinó antes de le ejecución del algoritmo), el carácter en el flujo de datos de entrada comprimidos que sigue a la entrada actual se almacena entonces en el módulo de descompresión LZ77/LZ78 como C. Si se desea, ese carácter se visualiza también al usuario. En el paso 150, el carácter C y la entrada de diccionario E se concatenan y se almacenan en el módulo de descompresión de datos LZ77/LZ78 como ADD. Una vez más, si se desea, la concatenación resultante también se visualiza al usuario.

Al finalizar el paso 144 o el paso 150, el puntero actual al flujo de datos de entrada se avanza al comienzo de la siguiente entrada en el flujo de datos en el paso 152. El control vuelve entonces al paso 134 y el proceso de descompresión se repite para la siguiente secuencia de datos de entrada hasta que se haya agotado el flujo de datos comprimidos de entrada. El proceso de descompresión se completa entonces.

La figura 14 es un ejemplo de la compresión de texto usando la técnica híbrida de las figuras 12 y 13. En el ejemplo, se supone que el alfabeto contiene 7 caracteres (A, B, C, D, E, F, G), y los datos que se van a comprimir son "ABCBDEFGABC". Como se muestra, el diccionario de compresión contiene inicialmente las secuencias frecuentes "ABC" y "DEF", donde las secuencias frecuentes se determinaron usando cualquiera de las técnicas descritas con anterioridad o a partir de cualquier otra técnica basada en la estadística como sería aparente para los que sean expertos en la técnica. La figura 14 ilustra la manera en la que se consiguió la compresión, donde hay tres opciones: diccionario prerrellenado, referencia LZ77 y literal LZ77. En el ejemplo de la figura 14, no se usa carácter de extensión. Durante la compresión, se usa un bit para indicar si la siguiente información es un literal o si es una referencia de diccionario o una referencia LZ77. Las dos últimas opciones comparten un espacio común de direcciones, pero solamente se usa una longitud si se usa una referencia de estilo LZ77.

Como se ilustra en la figura 14, al inicio de la compresión, el puntero al carácter en el que comenzará la compresión se coloca en el primer carácter de la secuencia de texto que se vaya a comprimir. En el paso 0, la coincidencia más larga se encuentra en el diccionario de compresión, y se transmite o se almacena localmente dependiendo de la aplicación del algoritmo de compresión, un puntero a esta entrada (D0). El puntero se mueve entonces en el número de caracteres transmitidos. En el paso 1, se transmite una referencia LZ77 ya que ni "B" ni "BD" están en el diccionario de compresión. El indicador indica que se trasmitió una referencia LZ77 (2, 1), donde "2" es un puntero a la segunda entrada en la ventana actual, y "1" es la longitud. El puntero se mueve después un carácter antes del paso 2. En el paso 2, se encuentra la coincidencia más larga en el diccionario de compresión, y se transmite o se almacena localmente dependiendo de la aplicación del algoritmo de compresión, un puntero a esta entrada (D1). El puntero se mueve entonces el número de caracteres transmitidos antes del paso 3. En el paso 3, se envía un literal LZ77 ya que el carácter "G" no está en la ventana de compresión o en el diccionario de compresión. El puntero se mueve entonces un carácter antes del paso 4. En el paso 4, se encuentra la coincidencia más larga en el diccionario de compresión, y se transmite o se almacena localmente dependiendo de la aplicación del algoritmo de compresión, un puntero a esta entrada (D0). El puntero se mueve entonces el número de caracteres transmitidos. En el paso 4, se usó la entrada de diccionario en lugar de la entrada de ventana ya que se supuso que era de un coste más bajo que la referencia de puntero LZ77. Como el puntero está al final del texto que se va a comprimir, se completa el proceso de codificación.

Cualquiera de las mejoras del estándar LZ77, tal como la aplicación de alguna forma de compresión a los literales, longitudes y direcciones que se están transmitiendo, la regeneración del diccionario o la ventana de forma que no se necesitan transmitir literales y similares, pueden ser igualmente bien aplicadas a cualquiera de las realizaciones anteriormente mencionadas que sean extensiones de la técnica de compresión LZ77. También como se ha descrito con anterioridad, se pueden usar diferentes diccionarios (o diferentes textos para anexar al comienzo de la ventana) para diferentes géneros de los datos que se estén transmitiendo. Un número de código inicial enviado al comienzo de cada mensaje indicaría entonces qué diccionario o texto anexado al comienzo se estaba usando (por ejemplo, uno para prosa en inglés, para software de ordenador o para formatos de transacción de empresa), o se transmitiría un código especial indicando que el diccionario o el texto anexado al comienzo se ha derivado de manera específica a partir de texto anteriormente transmitido.

En una implementación de muestra de las técnicas de la invención, un conjunto de diccionarios de compresión de datos prerrellenados se distribuye a un número de servidores que de manera regular intercambian documentos comprimidos entre ellos. Como se ha descrito con anterioridad, los diccionarios se pueden combinar antes de la distribución para ahorrar costes de transmisión. De manera periódica, se distribuye un nuevo conjunto de diccionarios prerrellenados para suplementar el viejo conjunto de diccionarios prerrellenados. El nuevo conjunto podría incluir diccionarios para nuevos géneros, así como más diccionarios hasta la fecha para viejos géneros que reflejen los cambios en la terminología. De manera ideal, los nuevos diccionarios están construidos de forma que se solapen sustancialmente con los viejos diccionarios, con lo que la representación combinada para todos los diccionarios nuevos y viejos no difiera mucho de la representación combinada para los diccionarios viejos solos. Se distribuyen entonces un fichero que lista los cambios entre las dos representaciones de un diccionario prerrellenado. Al recibir este fichero, cada uno de los servidores modifica su representación de diccionario combinado con el fin de añadir los nuevos diccionarios. Los viejos diccionarios se pueden retener ahora en la representación combinada o ser borrados, dependiendo de si se necesitarán o no con el fin de descomprimir ficheros antiguos. Dicha técnica llevará a una transmisión de datos y a unos costes de almacenamiento reducidos de manera sustancial, en particular entre dos servidores que a menudo intercambien ficheros del mismo género.

Por ejemplo, se realizó una simulación de la invención usando la mitad de un cuerpo de datos, donde el diccionario prerrellenado era aprendido a partir de la otra mitad. Para propósitos de comparación, los datos se descomprimieron usando “gzip”, una simulación de “gzip” y una simulación de “gzip” con un diccionario prerrellenado de acuerdo con la invención. Se usó una simulación ya que “gzip” se debería haber modificado para ser usado con un diccionario prerrellenado de acuerdo con la invención. La simulación de “gzip” se probó sin el diccionario prerrellenado meramente para verificar que la simulación se aproximaba al funcionamiento de “gzip” antes de la adición del diccionario prerrellenado. Se obtuvieron los siguientes resultados:

Porcentaje de compresión con gzip:	50,6
Porcentaje de compresión con simulación:	50,7
Porcentaje de compresión con simulación usando diccionario:	62,8

Los expertos en la técnica apreciarán que una disminución de 12 puntos porcentuales en el tamaño del fichero es completamente sustancial. En este ejemplo, ¡eso equivale a aproximadamente un ahorro del 25% de memoria!

Aunque se han descrito con anterioridad numerosas realizaciones de la invención y numerosas extensiones del concepto inventivo, los expertos en la técnica apreciarán fácilmente que son posibles muchas modificaciones adicionales en la realización de ejemplo sin apartarse de manera material de las enseñanzas novedosas y de las ventajas de la invención.

Por ejemplo, como se ha hecho notar con anterioridad, los diccionarios prerrellenados de la invención se pueden formar de cualquiera de un número de maneras y se pueden usar con cualquiera de un número de variaciones de la técnica de compresión de Lempel - Ziv básica. Además, se puede optimizar el funcionamiento de la compresión mediante la supervisión o el seguimiento de manera independiente del funcionamiento de compresión del diccionario de compresión de datos prerrellenado y del diccionario de compresión específico de datos. Por ejemplo, si el diccionario de compresión de datos prerrellenado no proporciona una mejor compresión de datos que el diccionario de compresión específico de datos, entonces en el caso de que el diccionario específico de datos pase a estar lleno y necesite ser reinicializado, el diccionario de compresión de datos prerrellenado se sustituye por el diccionario específico de datos y se reinicializa el diccionario específico de datos. Por otra parte, en el caso de que el diccionario de compresión de datos prerrellenado sea más útil para la compresión, se retiene y se reinicializa el diccionario de compresión específico de datos. Para determinar qué diccionario proporciona una mejor compresión que otro diccionario, se mantienen los ahorros asociados con cada uno de los diccionarios mientras se produce la compresión. La longitud de cada una de las cadenas comprimidas que usan cada uno de los diccionarios también se conserva, ya que es la cantidad de datos resultante proveniente de la parte de la compresión efectuada por cada uno de los diccionarios. Se elige un umbral por debajo del cual se dice que el diccionario prerrellenado está produciendo una mejora tan pequeña que sería mejor permitir que el diccionario específico de datos crezca más. Esto se hace tanto d manera global (para la totalidad del documento comprimido hasta este punto) así como de manera local manteniendo varios conjuntos diferentes de estos datos. En efecto, este proceso permite que todo el espacio de direcciones proporcionado para el diccionario de compresión de datos prerrellenado y para el diccionario de compresión específico de datos sea utilizado por el dic-

cionario específico de datos cuando el diccionario prerrellenado no esté proporcionando una mejora en el proceso de compresión.

Además, un procedimiento usado menos recientemente (LRU) del tipo descrito por Storer en la Patente de los Estados Unidos número 4.876.541 hace innecesario el conservar los dos diccionarios independientes, porque la entrada usada menos recientemente siempre es descartada con independencia de qué diccionario provenga. Se debería notar que, en este contexto, "usado" incluye la lectura o la escritura de un código como una parte de un flujo de datos comprimidos, pero la entrada del código dentro del diccionario en la primera aparición de su cadena se puede contar o no como una "utilización".

Además, se pueden usar múltiples diccionarios de compresión de datos prerrellenados en el proceso de compresión a expensas de reducir el tamaño del diccionario específico de datos. En el caso más extremo, el diccionario específico de datos se podría eliminar por completo y el espacio de direcciones entero normalmente compartido por el diccionario prerrellenado y el diccionario específico de datos sería compartido por los diccionarios prerrellenados pertinentes. Por supuesto, en este caso, todas las secuencias de caracteres, palabras y frases en el texto que se vaya a comprimir necesitarían estar presentes en uno o en más de los diccionarios de compresión de datos prerrellenados, lo que es completamente posible cuando el diccionario o los diccionarios prerrellenados son los suficientemente grandes como para contener todas las combinaciones de caracteres que es probable que ocurran en el texto que se vaya a comprimir. Además, si no se encuentran todas las combinaciones de caracteres en el diccionario de compresión de datos prerrellenado, las combinaciones no encontradas se podrán almacenar simplemente sin comprimir y/o el diccionario prerrellenado se podría actualizar durante la compresión para incluir esa combinación de caracteres. Por otra parte, se podría usar el algoritmo para determinar si el conjunto completo de caracteres se encuentra presente en el diccionario de compresión de datos prerrellenado creado por el software descrito con respecto a la figura 6, que realiza el análisis de la frecuencia de la colección de documentos a partir de los cuales se forma el diccionario prerrellenado. En el caso de que no estén presentes todos los caracteres, aquellos caracteres que se encuentren ausentes serán insertados en el diccionario de compresión de datos prerrellenado, garantizando de esta manera que no se necesitan enviar secuencias de escape para indicar que el siguiente carácter está ausente del diccionario.

Como otra modificación a las realizaciones preferidas de la invención, las direcciones de palabra de código de las entradas de diccionario se pueden codificar a sí mismas usando una técnica tal como la codificación de Hufman de forma que se puedan representar las direcciones más frecuentemente usadas usando menos bits. De manera similar, los caracteres de extensión se pueden codificar usando una codificación de longitud variable, tal como la codificación de Hufman, para mejorar el funcionamiento. En tal caso, no es necesario limitar el tamaño del diccionario específico de datos, y así es innecesario descargar el diccionario específico de datos, para que las palabras de código Lempel - Ziv sean sólo enteros. Estos enteros se codifican después por medio de secuencias binarias de longitud variable. Además, si la distribución de estas palabras de código enteros varía en todo el documento, de forma que su entropía local sea inferior que su entropía global, entonces se deberían usar versiones adaptativas del esquema de codificación de longitud variable, tal como una codificación de Hufman adaptativa o una codificación aritmética adaptativa.

De acuerdo con esta técnica, se conoce el tamaño del código que se está transmitiendo usando la codificación para las palabras de código y se basa en el número de palabras de código en el diccionario. Se puede mantener una lista de palabras de código usadas recientemente y en el momento justo en el que las palabras de código recientemente usadas formen un subconjunto pequeño del espacio entero de las palabras de código implicadas en el esquema de compresión de palabras de código, se puede retomar la compresión de estas palabras de código. Esto es, se puede crear un nuevo diccionario para comprimir palabras de código. Este nuevo diccionario se podrían construir a partir de las palabras de código recientemente usadas bajo la suposición de que las palabras de código recientemente usadas son más probables de ser usadas de nuevo de manera local, mientras que las palabras de código que no se han usado recientemente son menos probables de ser usadas de nuevo.

REIVINDICACIONES

1. Un procedimiento de compresión de datos en un sistema de compresión de datos para comprimir un texto que contenga una secuencia de caracteres, **caracterizado** porque hace que el sistema de compresión de datos lleve a cabo los pasos de:

(a) determinar mediante la compresión de un subconjunto de la mencionada secuencia de caracteres usando cada uno de una pluralidad de diccionarios de compresión de datos prerrellenados, un diccionario de compresión de datos prerrellenado que contenga las combinaciones predeterminadas de caracteres que ocurran con probabilidad de manera frecuente y que sea representativo de un género de texto, dicho diccionario de la mencionada pluralidad de diccionarios de compresión de datos prerrellenados proporciona la mayor compresión para el mencionado subconjunto de dicha secuencia de caracteres y seleccionando el diccionario determinado de la mencionada pluralidad de diccionarios de compresión de datos prerrellenados para su uso en la compresión de la mencionada secuencia de caracteres;

(b) inicializar un puntero a un primer carácter en la mencionada secuencia de caracteres para su compresión;

(c) comparar los caracteres que comienzan en el mencionado puntero con las secuencias de caracteres almacenadas en el mencionado diccionario de compresión de datos prerrellenado y determinar una coincidencia más larga de los mencionados caracteres que comienzan en dicho puntero con las mencionadas secuencias de caracteres almacenadas en el mencionado diccionario de compresión de datos prerrellenado;

(d) almacenar un puntero de diccionario a la mencionada coincidencia más larga en el mencionado diccionario de compresión de datos prerrellenado en una memoria [2, 5] como una representación comprimida de los mencionados caracteres que constituyen la mencionada coincidencia más larga;

(e) mover el mencionado puntero a un carácter en la mencionada secuencia de caracteres que sigue a la mencionada coincidencia más larga; y

(f) repetir los pasos c al e para todos los caracteres de la mencionada secuencia de caracteres que se vayan a comprimir.

2. Un procedimiento de compresión de datos como en la reivindicación 1, comprendiendo de manera adicional el paso de crear el mencionado diccionario de compresión de datos prerrellenado antes de dicho paso de determinación mediante la realización de los pasos de:

analizar las secuencias de caracteres en al menos una secuencia de caracteres de muestra que sea representativa de la secuencia de caracteres que se vayan a comprimir para determinar la frecuencia de la ocurrencia de las mencionadas secuencias de caracteres en al menos una de las mencionadas secuencias de caracteres; y

seleccionar como dichas combinaciones predeterminadas, aquellas secuencias de caracteres en al menos una de las mencionadas secuencias de caracteres de muestra que ocurran más frecuentemente en al menos una de las mencionadas secuencias de caracteres de muestra.

3. Un procedimiento de compresión de datos como en la reivindicación 2, en el que el paso de análisis comprende los pasos de determinar un número de ocurrencias de cada una de las secuencias de caracteres en el menos una de las mencionadas secuencias de caracteres de muestra y calcular para cada secuencia de caracteres un producto del número de ocurrencias de la mencionada secuencia de caracteres con la diferencia en la longitud binaria de cada una de las mencionadas secuencias de caracteres y una longitud binaria del mencionado puntero de diccionario.

4. Un procedimiento de compresión de datos como en la reivindicación 3, en el que el mencionado paso de selección de las combinaciones predeterminadas comprende el paso de seleccionar para su almacenamiento en el mencionado diccionario de compresión de datos prerrellenado aquellas secuencias de datos que tengan el producto mayor determinado en el mencionado paso de cálculo de producto.

5. Un procedimiento de compresión de datos como en la reivindicación 4, comprendiendo de manera adicional los pasos de determinar si se encuentra presente un juego completo de caracteres en el mencionado diccionario de compresión de datos prerrellenado, y, en el caso de que todos los caracteres del mencionado juego de caracteres completo no se encuentre presente en el mencionado diccionario de compresión de datos prerrellenado, insertar aquellos caracteres del mencionado juego de caracteres que no se encuentren en el mencionado diccionario de compresión de datos prerrellenado en el mencionado diccionario de compresión de datos prerrellenado como nuevas entradas de diccionario en lugar de entradas de diccionario que tengan el producto más pequeño determinado en el mencionado paso de cálculo de producto.

6. Un procedimiento de compresión de datos como en la reivindicación 1, en el que el mencionado paso de determinar qué diccionario de compresión de datos prerrellenado proporciona la compresión mayor comprende los pasos

de realizar los pasos b - e para un subconjunto de caracteres de la mencionada secuencia de caracteres de datos que se vayan a comprimir para una pluralidad de diferentes diccionarios de compresión de datos prerrellenados para determinar cuál de la mencionada pluralidad de diferentes diccionarios de compresión de datos prerrellenados proporciona la compresión mayor para el mencionado subconjunto de caracteres, y el mencionado paso de selección comprende la selección del mencionado diccionario de compresión de datos prerrellenado como el mencionado diccionario de compresión de datos prerrellenado para su uso en la compresión de la mencionada secuencia de caracteres.

7. Un procedimiento de compresión de datos como en el paso 6, en el que el mencionado paso de seleccionar el mencionado diccionario de compresión de datos prerrellenado comprende el paso de seleccionar un diccionario de compresión de datos vacío en el caso de que se determine mientras se estén realizando los pasos b - e para el mencionado subconjunto de caracteres de dicha secuencia de caracteres de datos que se vayan a comprimir que el comenzar con un diccionario de compresión de datos vacío permitiría la mayor compresión de la mencionada secuencia de caracteres de datos.

8. Un procedimiento de compresión de datos como en la reivindicación 1, que comprende el paso adicional de (g) transmitir a un destinatario el mencionado puntero de diccionario como la mencionada representación comprimida de dichos caracteres que constituyen la mencionada coincidencia más larga.

9. Un procedimiento de compresión de datos como en la reivindicación 8, en el que el mencionado paso de seleccionar dicho diccionario de compresión de datos prerrellenado comprende los pasos de seleccionar un diccionario de compresión de datos prerrellenado de entre una pluralidad de diccionarios de compresión de datos prerrellenados en base a si la mencionada pluralidad de diccionarios de compresión de datos prerrellenados se encuentran disponibles o no para el mencionado destinatario de los datos transmitidos en el paso g.

10. Un procedimiento de compresión de datos como en la reivindicación 1, en el que el mencionado paso de seleccionar un diccionario de compresión de datos prerrellenado comprende el paso de habilitar la selección de múltiples diccionarios de compresión de datos prerrellenados para su uso durante la compresión de la mencionada secuencia de caracteres que se vayan a comprimir.

11. Un procedimiento de compresión de datos como en la reivindicación 1, comprendiendo el paso adicional de codificar el mencionado puntero de diccionario de forma que el mencionado puntero de diccionario pueda ser representado usando menos bits de dirección.

12. Un procedimiento de compresión de datos como en la reivindicación 1, comprendiendo el paso adicional de almacenar con la mencionada representación comprimida de dichos caracteres que constituyen la mencionada coincidencia más larga, una referencia a un diccionario de compresión de datos prerrellenado para su uso durante un proceso de descompresión.

13. Un procedimiento de compresión de datos como en la reivindicación 1, comprendiendo el paso adicional de almacenar la pluralidad de diccionarios de compresión de datos prerrellenados que contienen diferentes géneros de datos de texto, en el que la mencionada selección de dicho diccionario de compresión de datos prerrellenado comprende el paso de seleccionar el mencionado diccionario de compresión de datos prerrellenado a partir de la mencionada pluralidad de diccionarios de compresión de datos prerrellenados con lo que el diccionario de compresión de datos prerrellenado contiene datos provenientes del género más similar a la mencionada secuencia de caracteres que se vaya a comprimir.

14. Un procedimiento de compresión de datos como en la reivindicación 13, en el que el mencionado paso de almacenar la mencionada pluralidad de diccionarios de compresión de datos prerrellenados comprende el paso de disponer la mencionada pluralidad de diccionarios de compresión de datos prerrellenados de manera jerárquica por género de datos de texto contenidos dentro de los respectivos diccionarios de compresión de datos prerrellenados.

15. Un procedimiento de compresión de datos como en la reivindicación 13, en el que el mencionado paso de almacenar la mencionada pluralidad de diccionarios de compresión de datos prerrellenados comprende el paso de almacenar entradas comunes en la mencionada pluralidad de diccionarios de compresión de datos prerrellenados solamente una vez en un medio de almacenamiento con lo que las mencionadas entradas comunes son compartidas por la mencionada pluralidad de diccionarios de compresión de datos prerrellenados.

16. Un procedimiento de descompresión de una representación comprimida de una secuencia de caracteres, resultante del procedimiento de compresión de datos de la reivindicación 1, dicha representación comprimida comprendiendo punteros de diccionario a las correspondientes coincidencias más largas de secuencias de caracteres que comienzan en un puntero a caracteres particulares dentro de la mencionada secuencia de caracteres con secuencias de caracteres almacenados en el diccionario de compresión de datos prerrellenado seleccionado, dicho procedimiento de descompresión comprendiendo los pasos de:

(a) inicializar un puntero a un primer puntero de diccionario en la mencionada representación comprimida de dichos caracteres;

(b) recuperar una entrada de diccionario del mencionado diccionario de compresión de datos prerrellenado usando el mencionado puntero de diccionario apuntado por dicho puntero;

(c) almacenar la mencionada entrada de diccionario como una representación descomprimida de los caracteres que constituyen la mencionada coincidencia más larga;

(d) mover el mencionado puntero a un siguiente puntero de diccionario en la mencionada representación comprimida de los mencionados caracteres; y

(e) repetir los pasos b - d para todos los punteros de diccionario en la mencionada representación comprimida de dichos caracteres hasta que se hayan descomprimido todos los caracteres comprimidos en la mencionada secuencia de caracteres.

17. Un procedimiento de descompresión de datos como en la reivindicación 16, comprendiendo el paso adicional de extraer de la mencionada representación comprimida de dichos caracteres al menos una de: (1) la identidad del diccionario de compresión de datos prerrellenado para usar durante la descompresión, y (2) una indicación de una variante del algoritmo de Lempel - Ziv para ser usada para descomprimir la mencionada representación de dichos caracteres.

18. Un sistema de compresión de datos [1] para comprimir un texto que contenga una secuencia de caracteres que comprende:

una pluralidad de diccionarios de compresión de datos prerrellenados disponibles que contengan combinaciones predeterminadas de caracteres que son probables que ocurran de manera frecuente en una secuencia de caracteres del mencionado texto y siendo representativas de un género de texto;

una memoria [2, 5] que almacene el mencionado texto después de que haya sido descomprimido; y

comprendiendo además un medio de compresión para realizar los pasos de:

(a) determinar mediante la compresión de un subconjunto de la mencionada secuencia de caracteres usando cada uno de la mencionada pluralidad de diccionarios de compresión de datos prerrellenados, dicho diccionario de la mencionada pluralidad de diccionarios de compresión de datos prerrellenados proporciona la mayor compresión para el mencionado subconjunto de dicha secuencia de caracteres y seleccionar el diccionario determinado de la mencionada pluralidad de diccionarios de compresión de datos prerrellenados para su uso en la compresión de la mencionada secuencia de caracteres;

(b) inicializar un puntero a un primer carácter en la mencionada secuencia de caracteres para compresión,

(c) comparar los caracteres que comienzan en el mencionado puntero con secuencias de caracteres almacenados en el mencionado diccionario de compresión de datos prerrellenado y determinar la coincidencia más larga de los mencionados caracteres que comienza en el mencionado puntero con dichas secuencias de caracteres almacenadas en el mencionado diccionario de compresión de datos prerrellenado,

(d) almacenar un puntero de diccionario a la mencionada coincidencia más larga en el mencionado diccionario de compresión de datos prerrellenado en la mencionada memoria [2, 5] como una representación comprimida de los mencionados caracteres que constituyen la coincidencia más larga,

(e) mover el mencionado puntero a un carácter en la mencionada secuencia de caracteres que sigue a la mencionada coincidencia más larga, y

(f) repetir los pasos b - d para todos los caracteres en la mencionada secuencia de caracteres que se vaya a comprimir.

19. Un sistema de compresión de datos [1] como en la reivindicación 18, comprendiendo de manera adicional un medio para crear el mencionado diccionario de compresión de datos prerrellenado, dicho medio de creación del diccionario de compresión de datos prerrellenado insertando dentro del mencionado diccionario de compresión de datos prerrellenado aquellas combinaciones de caracteres que ocurren con más frecuencia en al menos una secuencia de caracteres de muestra que sea representativa de la secuencia de caracteres que se vaya a comprimir.

20. Un sistema de compresión de datos [1] como en la reivindicación 19, en el que el mencionado medio de creación del diccionario de compresión de datos prerrellenado inserta en el mencionado diccionario de compresión de datos prerrellenado aquellas combinaciones de caracteres en al menos una secuencia de caracteres de muestra que sea representativa de la secuencia de caracteres que se vaya a comprimir que tenga el producto más grande de (a) un número de ocurrencias de cada una de las combinaciones de caracteres en al menos una de las mencionadas secuencias

de caracteres de muestra con (b) una diferencia en la longitud binaria de cada una de las mencionadas combinaciones de caracteres y una longitud binaria del mencionado puntero de diccionario.

21. Un sistema de compresión de datos [1] como en la reivindicación 20, en el que el mencionado medio de creación del diccionario de compresión de datos prerrellenado determina si está presente un juego completo de caracteres de la mencionada secuencia de caracteres que se va a comprimir en el mencionado diccionario de compresión de datos prerrellenado, y, en el caso de que todos los caracteres del mencionado juego de caracteres completo no se encuentren presentes en la mencionada secuencia de caracteres que se vaya a comprimir, inserta aquellos caracteres en el mencionado juego de caracteres que se encuentren ausentes en el mencionado diccionario de compresión de datos prerrellenado dentro del mencionado diccionario de compresión de datos prerrellenado como nuevas entradas de diccionario en lugar de las entradas de diccionario que tengan el producto más pequeño determinado por el mencionado medio de creación del diccionario de compresión de datos prerrellenado.

22. Un sistema de compresión de datos [1] como en la reivindicación 18, en el que el mencionado diccionario de compresión de datos prerrellenado es uno de un número de diccionarios de compresión de datos prerrellenados, el mencionado diccionario de compresión de datos prerrellenado conteniendo aquellas combinaciones de caracteres que proporcionan más compresión de un subconjunto de caracteres de la mencionada secuencia de caracteres de datos que se vayan a comprimir que la compresión proporcionada al mencionado subconjunto de caracteres de la mencionada secuencia de caracteres de datos por cada uno del resto del mencionado número de diccionarios de compresión de datos prerrellenados.

23. Un sistema de compresión de datos [1] como en la reivindicación 18, en el que la mencionada memoria [2, 5] está en una localización remota del mencionado medio de compresión, comprendiendo de manera adicional un medio para transmitir el mencionado puntero de diccionario como dicha representación comprimida de los mencionados caracteres que constituyen dicha coincidencia más larga desde dicho medio de compresión a la mencionada memoria [2, 5].

24. Un sistema de compresión de datos [1] como en la reivindicación 18, en el que el mencionado diccionario de compresión de datos prerrellenado es uno de un número de diccionarios de compresión de datos prerrellenados que también se encuentra disponible en la localización remota.

25. Un sistema de compresión de datos [1] como en la reivindicación 18, en el que el mencionado medio de compresión comprende un medio para codificar el mencionado puntero de diccionario de forma que el mencionado puntero de diccionario pueda ser representado usando menos bits de dirección.

26. Un sistema de compresión de datos [1] como en la reivindicación 18, en el que el mencionado medio de compresión almacena en la mencionada memoria [2, 5] con dicha representación comprimida de los mencionados caracteres que constituyen la mencionada coincidencia más larga, una referencia a un diccionario de compresión de datos prerrellenado que se va a usar durante el proceso de descompresión.

27. Un sistema de compresión de datos [1] como en la reivindicación 18, en el que la mencionada memoria [2, 5] comprende al menos uno de entre un disco duro, una RAM, un CD ROM, un disco flexible y un disco óptico.

28. Un sistema de compresión de datos [1] como en la reivindicación 18, comprendiendo de manera adicional una memoria de diccionario [2, 5] para almacenar una pluralidad de diccionarios de compresión de datos prerrellenados que contienen diferentes géneros de datos de texto, con lo que el diccionario de compresión de datos prerrellenado usado para comprimir el mencionado texto contiene datos del género más similar a la mencionada secuencia de caracteres que se vaya a comprimir.

29. Un sistema de compresión de datos [1] como en la reivindicación 18, en el que la mencionada pluralidad de diccionarios de compresión de datos prerrellenados están dispuestos en la mencionada memoria [2, 5] de manera jerárquica por género de datos de texto contenidos dentro de los respectivos diccionarios de compresión de datos prerrellenados.

30. Un sistema de compresión de datos [1] como en la reivindicación 18, en el que la mencionada pluralidad de diccionarios de compresión de datos prerrellenados se almacenan en la mencionada memoria [2, 5] de forma que las entradas comunes en la mencionada pluralidad de diccionarios de compresión de datos prerrellenados se almacenan en dicha memoria [2, 5] solamente una vez y son compartidas por la mencionada pluralidad de diccionarios de compresión de datos prerrellenados.

31. Un sistema de descompresión de datos que descomprime una representación comprimida de una secuencia de caracteres, resultado de una compresión de datos llevada a cabo por medio de un sistema de compresión de datos de acuerdo con la reivindicación 18, la mencionada representación comprendiendo punteros de diccionario a las respectivas coincidencias más largas de secuencias de caracteres que comienzan en un puntero a caracteres particulares dentro de la mencionada secuencia de caracteres con secuencias de caracteres almacenadas en el diccionario de compresión de datos prerrellenado seleccionado, comprendiendo el mencionado sistema de descompresión:

una memoria de datos comprimidos para almacenar la mencionada representación comprimida de la mencionada secuencia de caracteres;

una memoria de datos descomprimidos para almacenar la mencionada secuencia de caracteres después de la descompresión;

y

un medio de descompresión para realizar los pasos de: (a) inicializar un puntero a un primer puntero de diccionario en la mencionada representación comprimida de los mencionados caracteres en la mencionada memoria de datos comprimidos, (b) recuperar una entrada de diccionario de un diccionario de compresión de datos prerrellenado usando el mencionado puntero de diccionario apuntado por el mencionado puntero, (c) almacenar la mencionada entrada de diccionario en la mencionada memoria de datos descomprimidos como una representación descomprimida de los caracteres que constituyen la mencionada coincidencia más larga, (d) mover el mencionado puntero al siguiente puntero de diccionario en la mencionada representación comprimida de dichos caracteres, y (e) repetir los pasos b - d para todos los punteros de diccionario en la mencionada representación comprimida de los mencionados caracteres en dicha memoria de datos comprimidos hasta que se hayan descomprimido todos los caracteres comprimidos de la mencionada secuencia de caracteres.

32. Un sistema de descompresión de datos como en la reivindicación 31, en el que el mencionado medio de descompresión comprende un medio para extraer de la mencionada representación comprimida de dichos caracteres en la mencionada memoria de datos comprimidos al menos una de: (1) la identidad del diccionario de compresión de datos prerrellenado para su uso durante la descompresión por el mencionado medio de descompresión, y (2) una indicación de una variante del algoritmo de Lempel - Ziv que se vaya a usar para descomprimir la mencionada representación comprimida de dichos caracteres en la mencionada memoria de datos comprimidos.

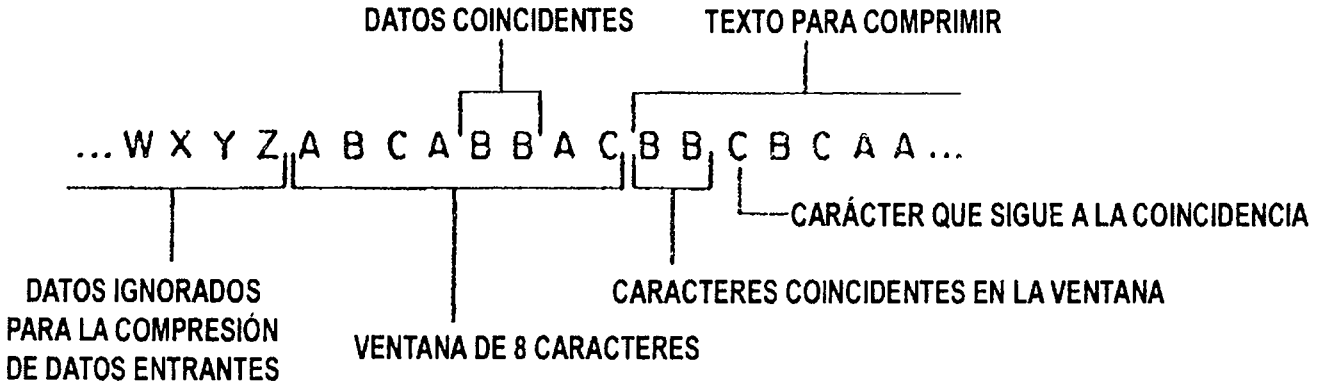

Fig. 1(A) COMPRESIÓN DE TEXTO LZ77 DE LA TÉCNICA ANTERIOR

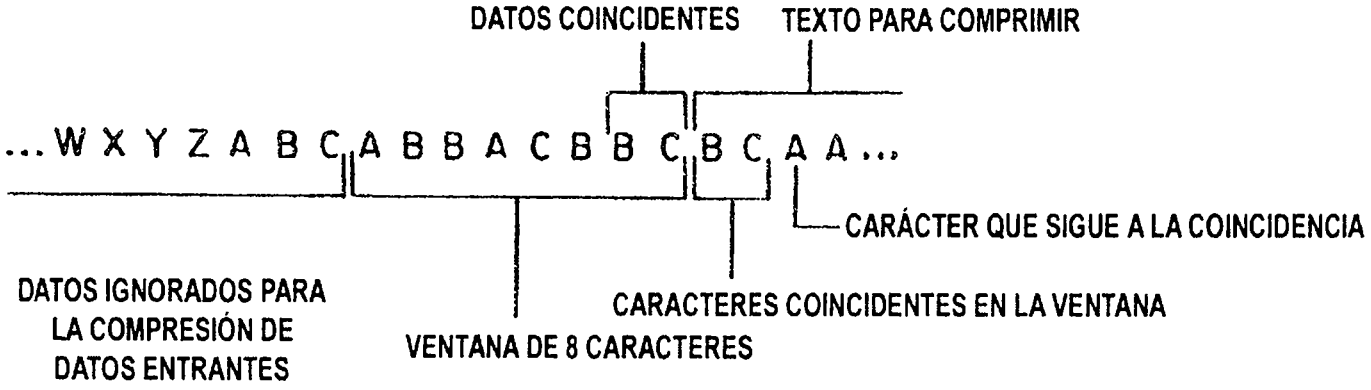

Fig. 1(B) COMPRESIÓN DE TEXTO LZ77 DE LA TÉCNICA ANTERIOR

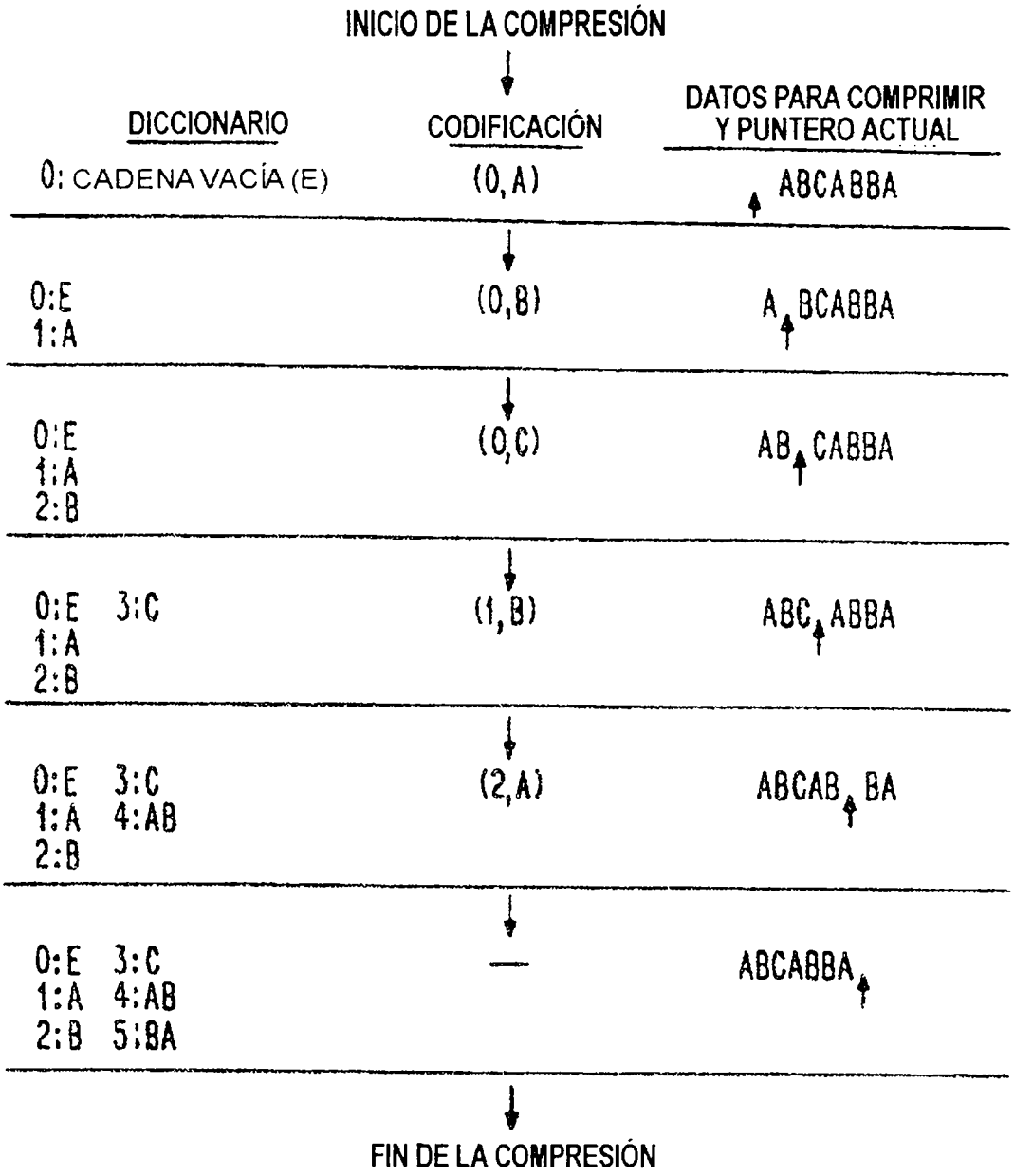

Fig. 2 COMPRESIÓN DE TEXTO LZ78 DE LA TÉCNICA ANTERIOR

INICIO DE LA COMPRESIÓN :

DICCIONARIO	CODIFICACIÓN	DATOS PARA COMPRIMIR Y PUNTERO ACTUAL
0:A 1:B 2:C	(0)	↑ABCABBA
AS ABOVE	(1)	A↑BCABBA
0:A 1:B 2:C 3:AB	(2)	AB↑CABBA
0:A 4:BC 1:B 2:C 3:AB	(3)	ABC↑ABBA
0:A 4:BC 1:B 5:CAB 2:C 3:AB	(1)	ABCAB↑BA
0:A 4:BC 1:B 5:CAB 2:C 6:ABB 3:AB	(0)	ABCABB↑A
0:A 4:BC 1:B 5:CAB 2:C 6:ABB 3:AB 7:BA	—	ABCABBA↑

FIN DE LA COMPRESIÓN

Fig. 3 COMPRESIÓN DE TEXTO L_MW DE LA TÉCNICA ANTERIOR

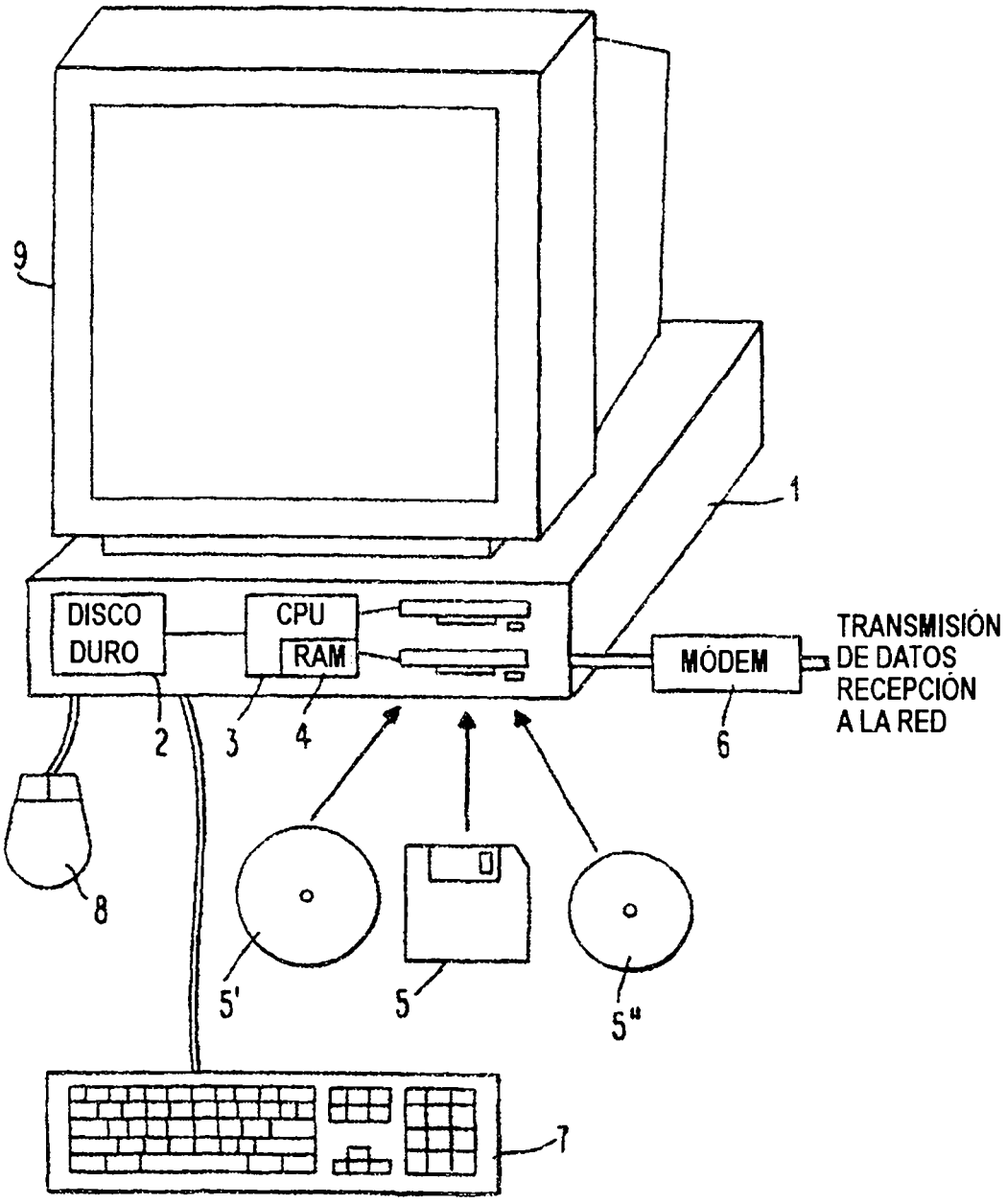
9
1
DISCO
DURO
CPU
RAM
MÓDEM
TRANSMISIÓN
DE DATOS
RECEPCIÓN
A LA RED
2
3
4
6
8
5'
5
5"
7

Fig. 4

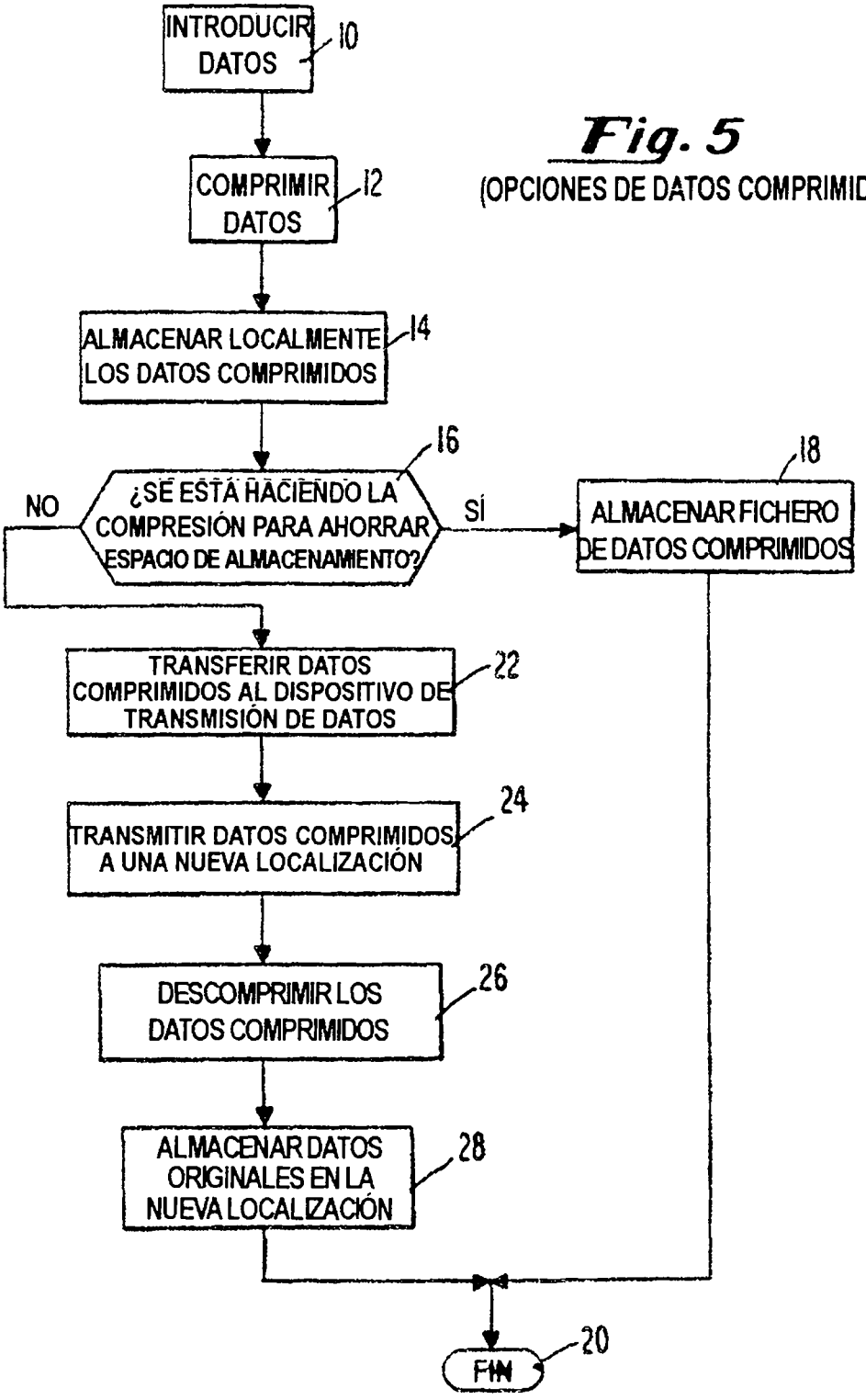

Fig. 5

(OPCIONES DE DATOS COMPRIMIDOS)

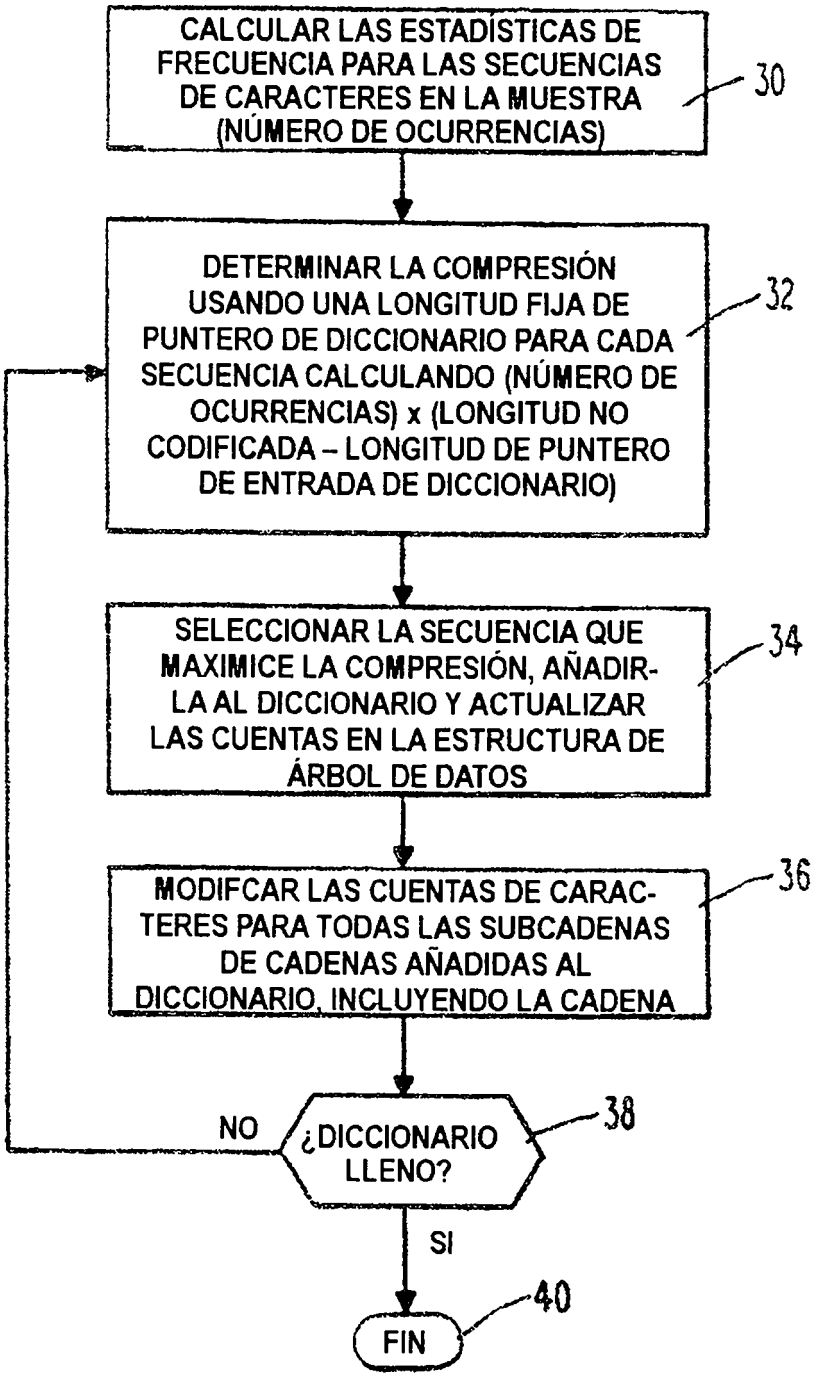

Fig. 6

(CREACIÓN DEL DICCIONARIO)

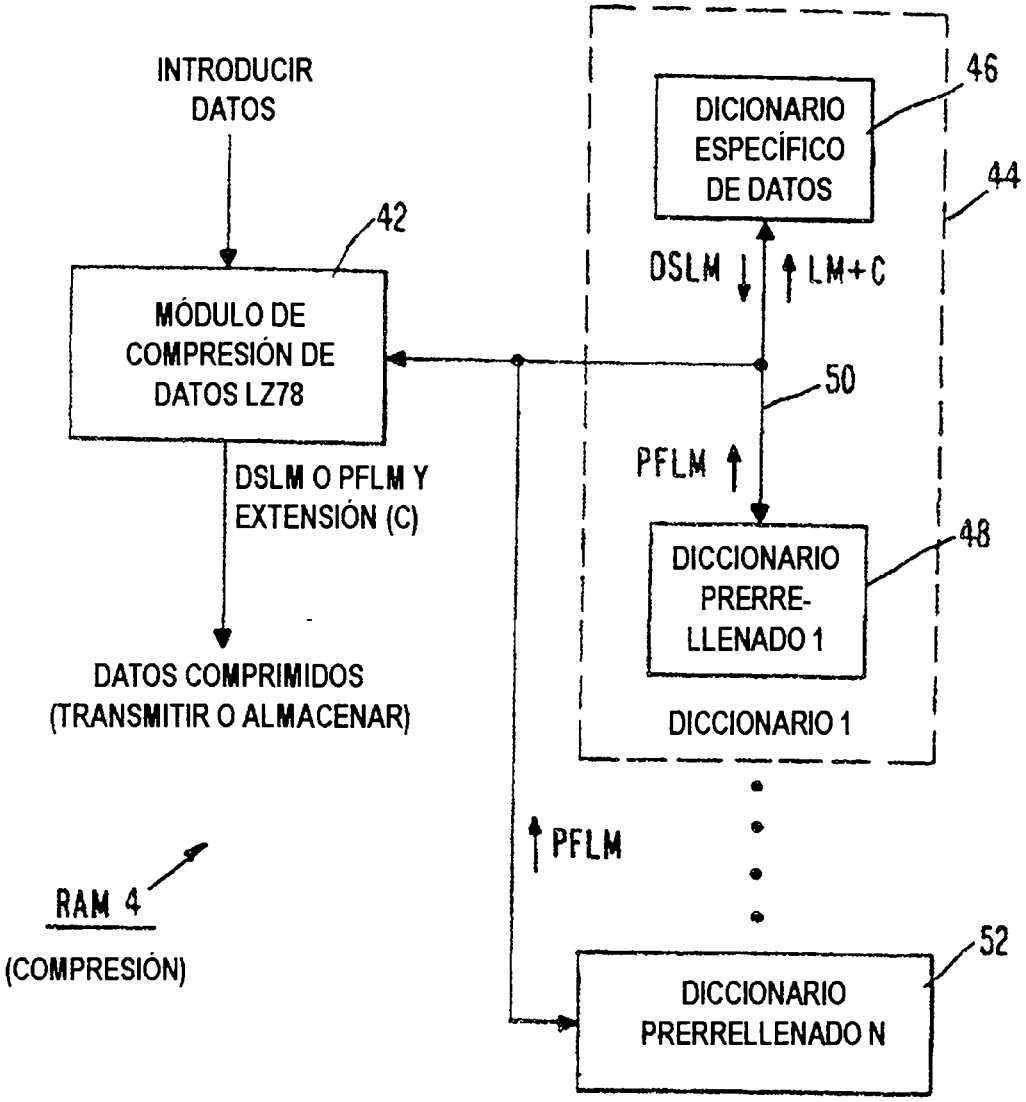
INTRODUCIR DATOS
42
MÓDULO DE COMPRESIÓN DE DATOS LZ78
DSLM O PFLM Y EXTENSIÓN (C)
DATOS COMPRIMIDOS (TRANSMITIR O ALMACENAR)
RAM 4
(COMPRESIÓN)
46
DICIONARIO ESPECÍFICO DE DATOS
44
DSLM
LM+C
50
PFLM
48
DICCIONARIO PRERRE-LLENADO 1
DICCIONARIO 1
PFLM
52
DICCIONARIO PRERRELLENADO N

Fig. 7

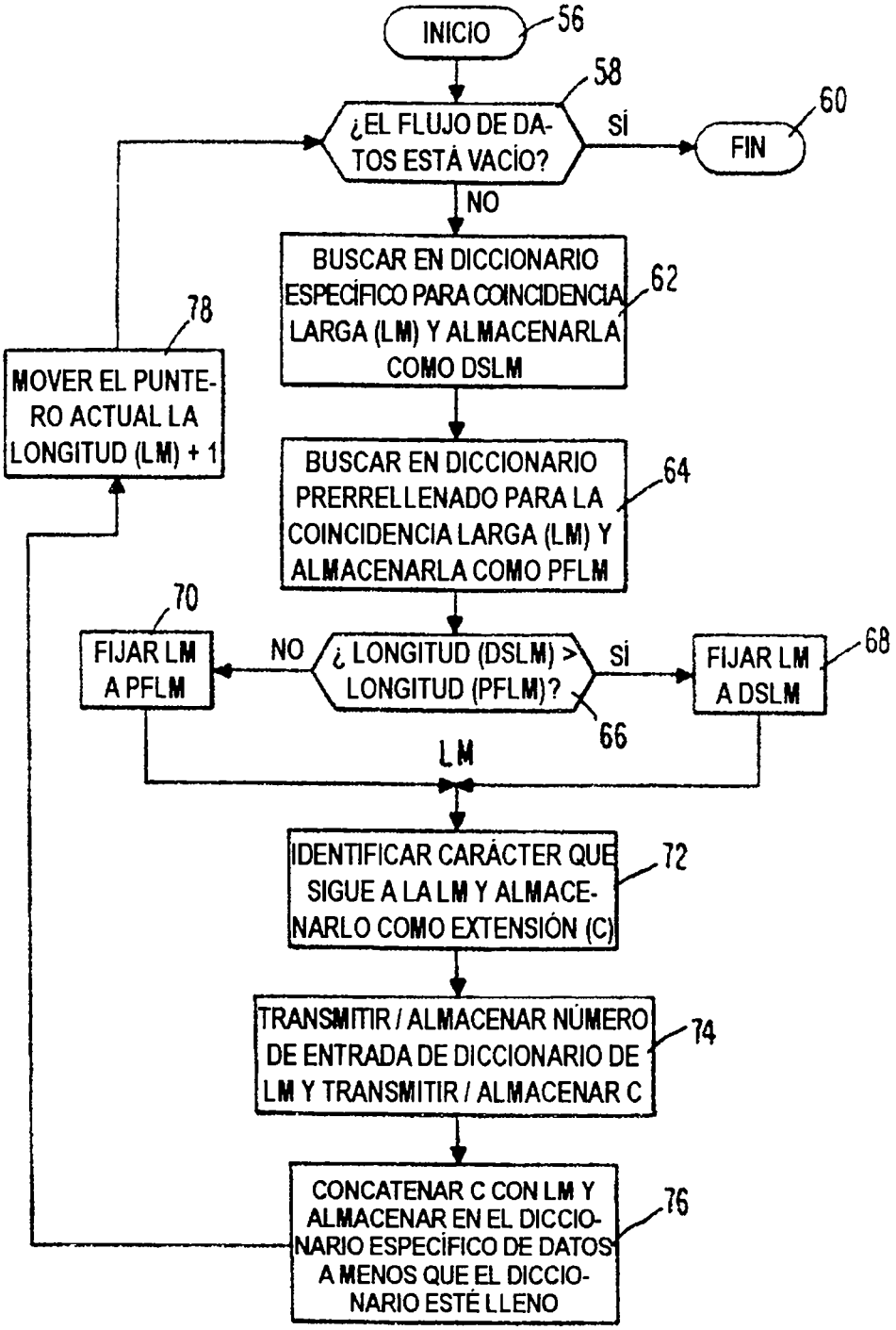

Fig. 8

(COMPRESIÓN LZ78 USANDO DICCIONARIO PRERRELLENADO)

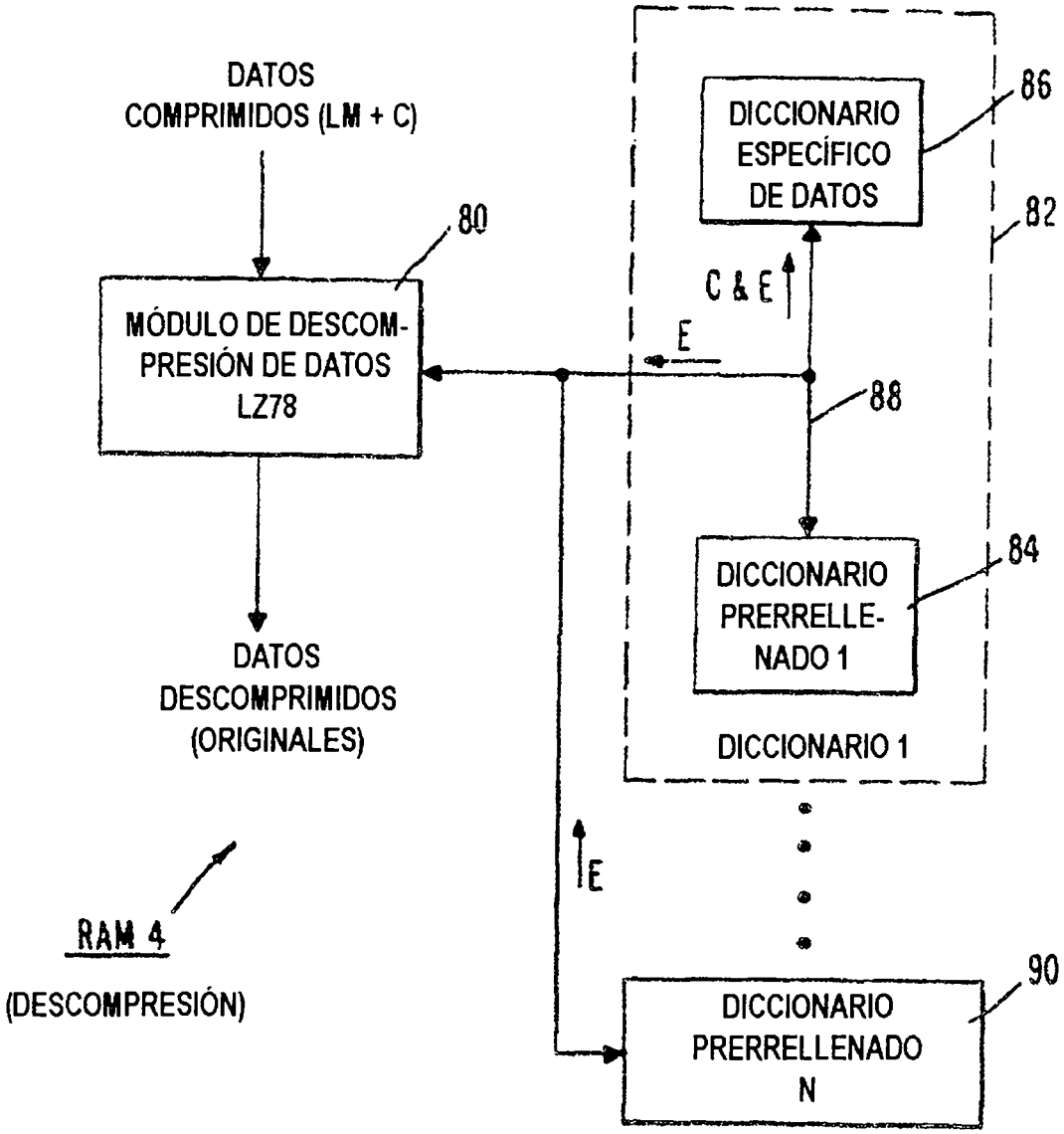
DATOS
COMPRIMIDOS (LM + C)
80
MÓDULO DE DESCOM-
PRESIÓN DE DATOS
LZ78
DATOS
DESCOMPRIMIDOS
(ORIGINALES)
RAM 4
(DESCOMPRESIÓN)
DICCIONARIO
ESPECÍFICO
DE DATOS
86
82
C & E
E
88
DICCIONARIO
PRERRELLE-
NADO 1
84
DICCIONARIO 1
E
DICCIONARIO
PRERRELLENADO
N
90

Fig. 9

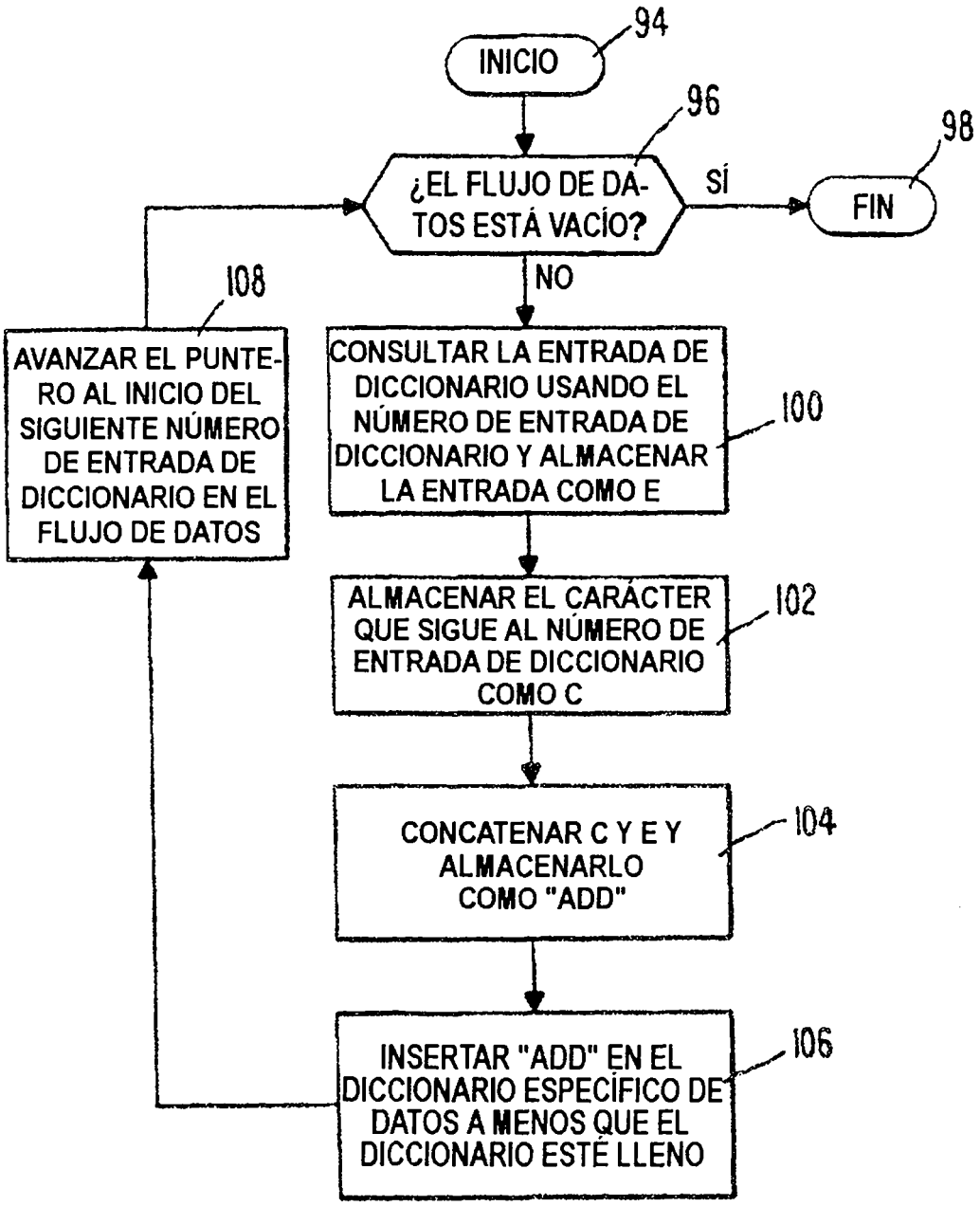

Fig. 10

(DESCOMPRESIÓN LZ78 USANDO DICCIONARIO PRERRELLENADO)

EL DICCIONARIO CONTIENE INICIALMENTE
(CONTENIDO DEL ALFABETO Y SECUENCIAS FRECUENTES)

CONTENIDO DEL ALFABETO { 0:A, 1:B, 2:C } { 3:AB, 4:CC, 5:BB } SECUENCIAS FRECUENTES

INICIO DE LA COMPRESIÓN

DICCIONARIO	CODIFICACIÓN	DATOS PARA SER COMPRIMIDOS Y PUNTERO ACTUAL
AS ABOVE	(3,C)	ABCABBA
0:A 4:CC 1:B 5:BB 2:C 6:ABC 3:AB	(3,B)	ABCABBA
0:A 4:CC 1:B 5:BB 2:C 6:ABC 3:AB 7:ABB	(0,-)	ABCABBA
COMO ANTERIORMENTE		ABCABBA

FIN DE LA COMPRESIÓN

Fig. 11 COMPRESIÓN DE TEXTO LZ78 CON DICCIONARIO PRERRELLENADO

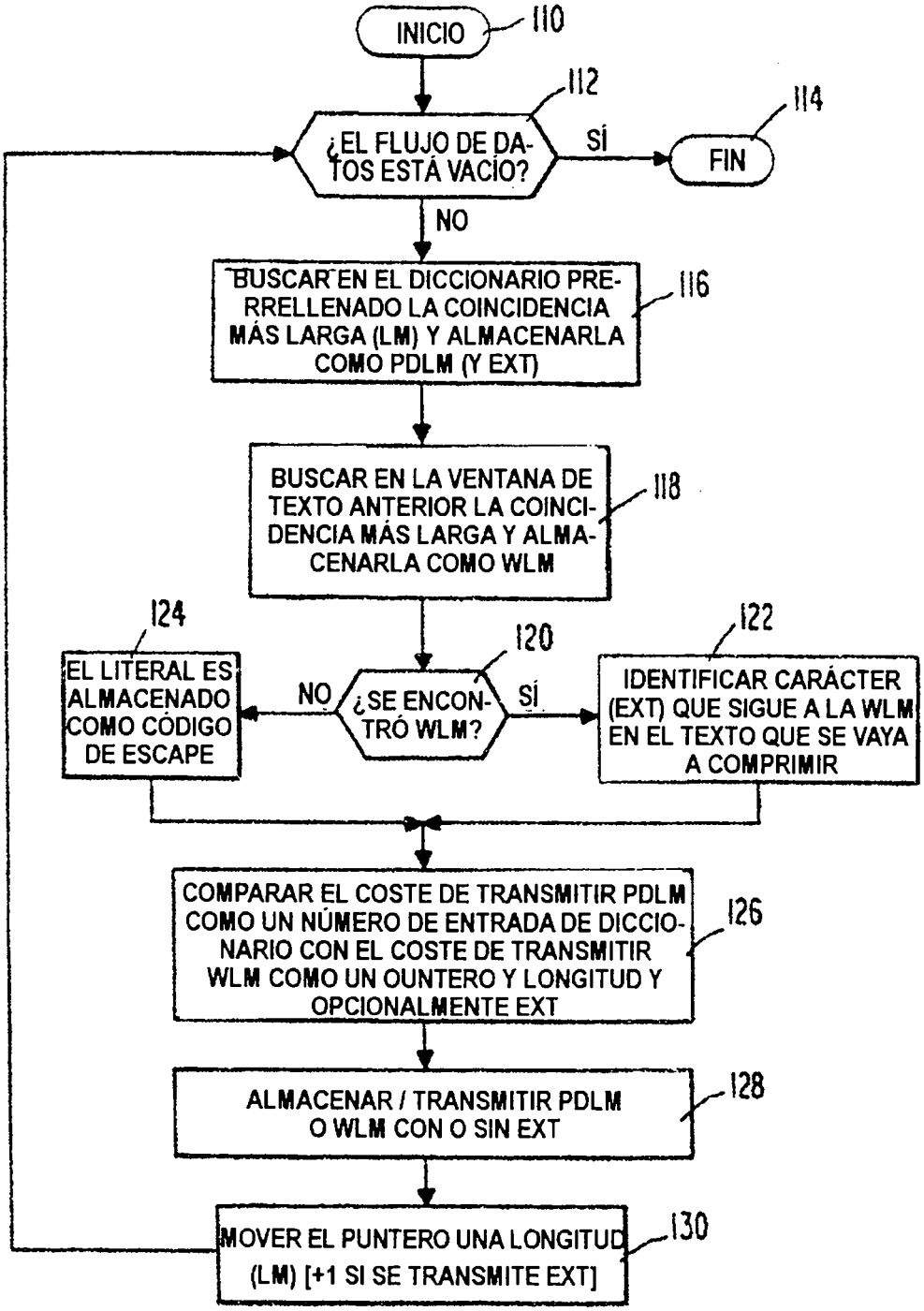

Fig. 12

(COMPRESIÓN HÍBRIDA LZ77 / LZ78 USANDO DICCIONARIO PRERRELLENADO)

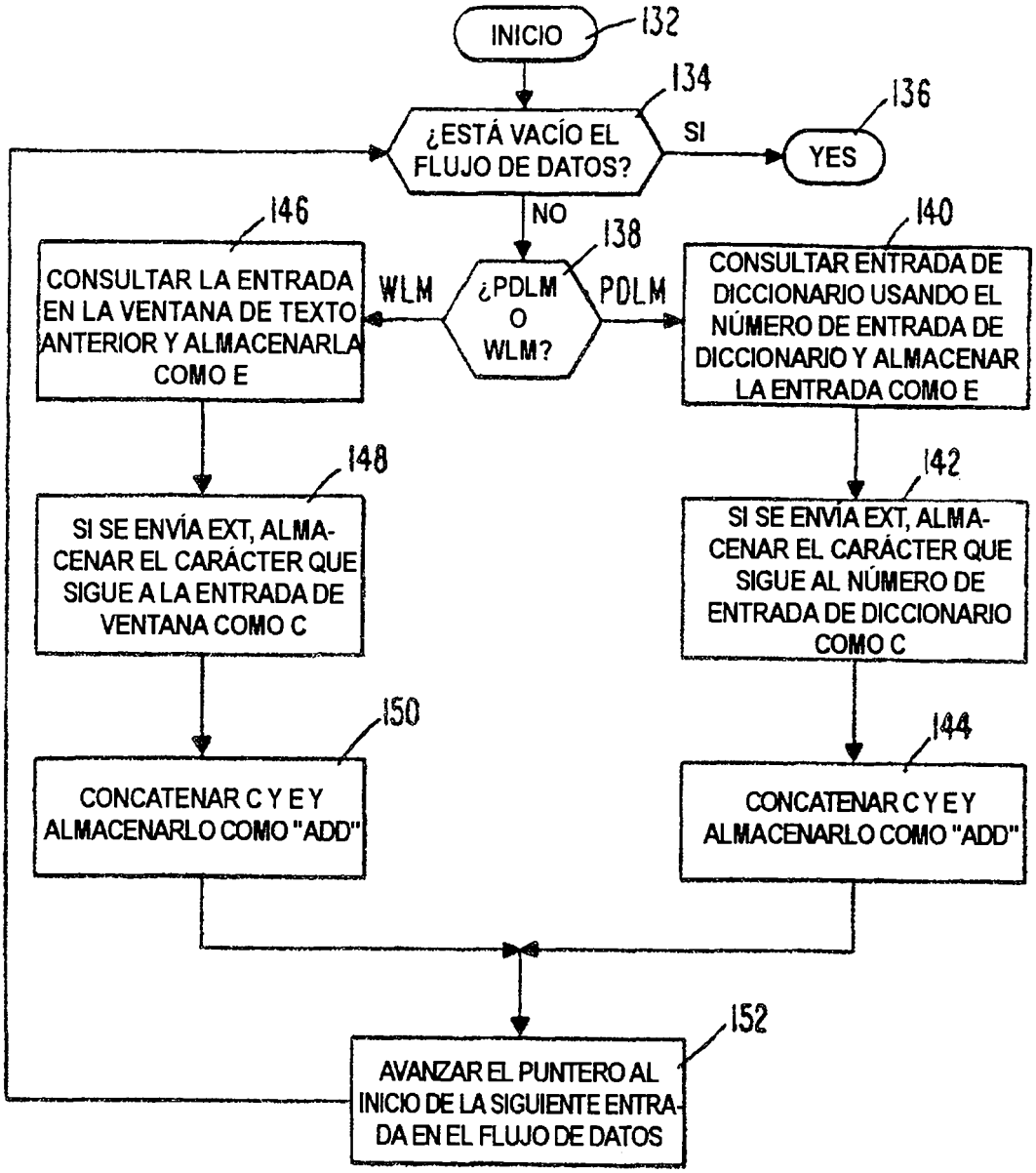

Fig. 13

(DESCOMPRESIÓN HÍBRIDA LZ77 / LZ78
USANDO DICCIONARIO PRERRELLENADO)

EL DICCIONARIO PRERRELLENADO CONTIENE INICIALMENTE
(CONTENIDO DEL ALFABETO Y SECUENCIAS FRECUENTES)

NÚMERO DE ENTRADA	ENTRADA
D0:	ABC
D1:	DEF
⋮	⋮

TEXTO QUE SE VA A COMPRIMIR:

ABCBDEFGABC

INICIO DE LA COMPRESIÓN:

PASO	CODIFICACIÓN	PROCEDIMIENTO	DATOS QUE VAN A SER COMPRIMIDOS Y PUNTERO ACTUAL
0	DØ	DICCIONARIO	ABCBDEFGABC
1	⟨2,1⟩	REFERENCIA LZ77	ABCBDEFGABC
2	D1	DICCIONARIO	ABCBDEFGABC
3	G	LITERAL LZ77	ABCBDEFGABC
4	DØ	DICCIONARIO	ABCBDEFGABC
COMPLETADA			ABCBDEFGABC

FIN DE LA COMPRESIÓN

Fig. 14

COMPRESIÓN DE TEXTO LEMPEL – Ziv
USANDO UNA COMPRESIÓN HÍBRIDA LZ77
/ LZ78 CON UN DICCIONARIO PRERRELLENADO