UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T V.44

(11/2000)

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

SERIE V: COMUNICACIÓN DE DATOS POR LA RED TELEFÓNICA

Control de errores

Procedimientos de compresión de datos

Recomendación UIT-T V.44

(Anteriormente Recomendación del CCITT)

RECOMENDACIONES UIT-T DE LA SERIE V

COMUNICACIÓN DE DATOS POR LA RED TELEFÓNICA

Generalidades	V.1–V.9
Interfaces y módems para la banda vocal	V.10–V.34
Módems de banda ancha	V.35–V.39
Control de errores	**V.40–V.49**
Calidad de transmisión y mantenimiento	V.50–V.59
Transmisión simultánea de datos y de otras señales	V.60–V.99
Interfuncionamiento con otras redes	V.100–V.199
Especificaciones de la capa interfaz para comunicaciones de datos	V.200–V.249
Procedimientos de control	V.250–V.299
Módems en circuitos digitales	V.300–V.399

Para más información, véase la Lista de Recomendaciones del UIT-T.

Recomendación UIT-T V.44

Procedimientos de compresión de datos

Resumen

La presente Recomendación describe un algoritmo de compresión de datos en los equipos de terminación del circuito de datos, con el cual se logra una calidad de funcionamiento mejor que con el procedimiento de la Recomendación V.42 *bis* en muchos tipos de datos. Además del método de trenes normales, el algoritmo tiene un método que puede comprimir eficazmente datos ya contenidos en paquetes.

Orígenes

La Recomendación UIT-T V.44, preparada por la Comisión de Estudio 16 (2001-2004) del UIT-T, fue aprobada por el procedimiento de la Resolución 1 de la AMNT el 17 de noviembre de 2000.

PREFACIO

La UIT (Unión Internacional de Telecomunicaciones) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones. El UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) es un órgano permanente de la UIT. Este órgano estudia los aspectos técnicos, de explotación y tarifarios y publica Recomendaciones sobre los mismos, con miras a la normalización de las telecomunicaciones en el plano mundial.

La Asamblea Mundial de Normalización de las Telecomunicaciones (AMNT), que se celebra cada cuatro años, establece los temas que han de estudiar las Comisiones de Estudio del UIT-T, que a su vez producen Recomendaciones sobre dichos temas.

La aprobación de Recomendaciones por los Miembros del UIT-T es el objeto del procedimiento establecido en la Resolución 1 de la AMNT.

En ciertos sectores de la tecnología de la información que corresponden a la esfera de competencia del UIT-T, se preparan las normas necesarias en colaboración con la ISO y la CEI.

NOTA

En esta Recomendación, la expresión "Administración" se utiliza para designar, en forma abreviada, tanto una administración de telecomunicaciones como una empresa de explotación reconocida de telecomunicaciones.

PROPIEDAD INTELECTUAL

La UIT señala a la atención la posibilidad de que la utilización o aplicación de la presente Recomendación suponga el empleo de un derecho de propiedad intelectual reivindicado. La UIT no adopta ninguna posición en cuanto a la demostración, validez o aplicabilidad de los derechos de propiedad intelectual reivindicados, ya sea por los miembros de la UIT o por terceros ajenos al proceso de elaboración de Recomendaciones.

En la fecha de aprobación de la presente Recomendación, la UIT ha recibido notificación de propiedad intelectual, protegida por patente, que puede ser necesaria para aplicar esta Recomendación. Sin embargo, debe señalarse a los usuarios que puede que esta información no se encuentre totalmente actualizada al respecto, por lo que se les insta encarecidamente a consultar la base de datos sobre patentes de la TSB.

ÍNDICE

Página

Página

Recomendación UIT-T V.44

Procedimientos de compresión de datos

1 Alcance

1.1 Consideraciones generales

La presente Recomendación describe un procedimiento de compresión de datos sin pérdidas para uso con los equipos de terminación del circuito de datos (DCE, *data circuit-terminating equipment*) de la serie V.

Las principales características del procedimiento de compresión de datos son:

a) un procedimiento de compresión basado en un algoritmo que codifica cadenas de caracteres recibidos del equipo terminal de datos (DTE, *data terminal equipment*) como códigos binarios de longitud variable;

b) un procedimiento de decodificación que recupera las cadenas de caracteres de los códigos binarios recibidos de longitud variable;

c) un mecanismo de construcción de cadenas que amplía rápidamente las cadenas existentes;

d) un modo de funcionamiento transparente invocado automáticamente cuando se detectan datos que no son comprimibles.

El anexo B describe la implementación de este procedimiento de compresión de datos a redes de paquetes.

En la cláusula 8 figura un resumen de los conjuntos de parámetros utilizados en la presente Recomendación.

En la cláusula I.7 se resumen las principales diferencias entre los algoritmos de UIT-T V.44 y V.42 *bis*.

La presente Recomendación contiene ejemplos ilustrativos; cuando aparezcan contradicciones entre un ejemplo y el texto normativo, este último tendrá precedencia.

1.2 Requisitos de los procedimientos de corrección de errores

Para el funcionamiento correcto de la función de compresión de datos, es necesario aplicar un procedimiento de corrección de errores entre las dos entidades que utilizan la presente Recomendación. En el caso de las Recomendaciones de la serie V, se deben implementar los procedimientos de corrección de errores del procedimiento de acceso al enlace para módems (LAPM, *link access procedure for modems*) definido en UIT-T V.42 o los procedimientos de corrección de errores de UIT-T V.76 o V.120.

NOTA – Los bits erróneos no detectados originarán el funcionamiento defectuoso de la función de compresión de datos. La utilización de una secuencia de verificación de trama (FCS, *frame check sequence*) de 32 bits, definida en ISO/CEI 13239, reduce fundamentalmente la posibilidad de estos errores, y puede ser conveniente en entornos con degradaciones importantes. Esta secuencia de verificación de trama de 32 bits es una opción en el LAPM de UIT-T V.42.

1.3 DCE que emplea compresión de datos

La función de compresión de datos se puede utilizar con un DCE con corrección de errores, como se muestra en la figura 1 y se describe en la cláusula 5. Los elementos de un DCE de la serie V con corrección de errores se especifican en UIT-T V.42.

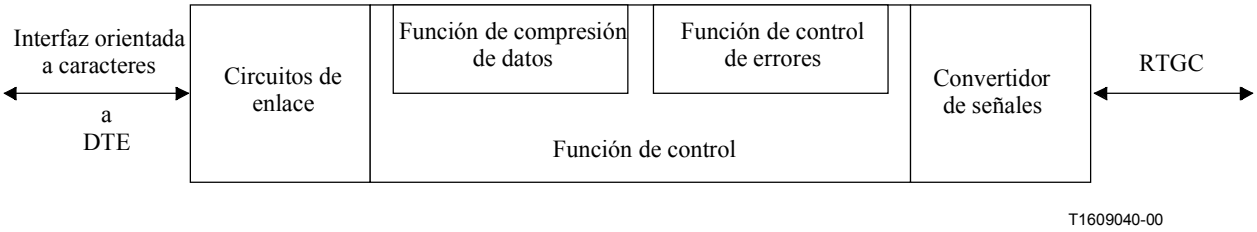

Figura 1/V.44 – DCE que emplea compresión de datos y control de errores

2 Referencias

Las siguientes Recomendaciones del UIT-T y otras referencias contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de la presente Recomendación. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todas las Recomendaciones y otras referencias son objeto de revisiones por lo que se preconiza que los usuarios de esta Recomendación investiguen la posibilidad de aplicar las ediciones más recientes de las Recomendaciones y otras referencias citadas a continuación. Se publica periódicamente una lista de las Recomendaciones UIT-T actualmente vigentes.

– UIT-T V.42 (1996) – *Procedimientos de corrección de errores para los equipos de terminación del circuito de datos que utilizan la conversión de modo asíncrono a modo síncrono.*

– UIT-T V.42 *bis* (1990) – *Procedimientos de compresión de datos para los equipos de terminación del circuito de datos que utilizan procedimientos de corrección de errores.*

– UIT-T V.120 (1996) – *Soporte proporcionado por una red digital de servicios integrados a equipos terminales de datos con interfaces del tipo serie V con multiplexión estadística.*

– ISO/CEI 13239:2000, *Information technology – Telecommunications and information exchange between systems – High-level data link control (HDLC) procedures*

3 Definiciones

En esta Recomendación se definen los términos siguientes.

3.1 alfabeto: Conjunto de todos los caracteres posibles. En la presente Recomendación, los caracteres son contiguos de 0 a $N_4 – 1$ (véase la cláusula 8).

3.2 anexar: Crear una nueva cadena que utiliza un carácter no concordado. Si el carácter no concordado sigue a una concordancia de cadena, la nueva cadena consiste en la concordancia de cadena con el carácter no concordado añadido al final; si el carácter no concordado sigue a un carácter, la nueva cadena consiste en ambos caracteres.

3.3 carácter: Un elemento de datos que es la entrada del DTE al codificador y que utiliza un número de bits predefinidos N_3 (véase la cláusula 8).

3.4 código: Una secuencia de bits que es la salida del codificador que representa control o información. Los tipos de códigos definidos son: códigos de control, ordinales, palabras de código y longitudes de extensión de cadena.

3.5 prefijo de código: Uno o dos bits que preceden a un código y que indican el tipo de código que sigue. Se definen en 6.6.3.

3.6 palabra de código: Un número binario en la gama 4 a $N_2 – 1$ que representa una cadena de caracteres consecutivos. En el codificador, corresponde a un segmento de cadena particular. En el decodificador corresponde a una cadena completa (véase la cláusula 8).

3.7 modo con compresión: Modo de funcionamiento en el cual los datos del DTE se transmiten como códigos binarios.

3.8 funcionamiento con compresión: Estado del DCE en el cual la función de compresión de datos está activa. El funcionamiento con compresión tiene dos modos: modo con compresión y modo transparente. Las transiciones entre estos modos pueden ser automáticas, y estar basadas en el contenido de la entrada de datos del DTE (véase 6.5).

3.9 código de control: Número binario en la gama 0 a 3, reservado para uso en la señalización de control de información de DCE a DCE relacionada con la función de compresión mientras se funciona en el modo con compresión (véase 7.2).

3.10 decodificador: Subfunción de compresión de datos que descomprime la salida de un codificador.

3.11 diccionario: Estructura de datos que representa las cadenas creada durante la codificación o decodificación. Para el codificador, consiste en la matriz de raíces, el árbol de nodos y la historia; para el decodificador, consiste en el conjunto de cadenas y la historia.

3.12 codificador: Subfunción de compresión de datos que comprime datos.

3.13 ESCAPE: En el modo transparente, la indicación del comienzo de una secuencia de instrucciones al decodificador.

3.14 extender: Crear una nueva cadena, que consiste en una concordancia de cadena con uno o más caracteres añadidos al final.

3.15 evacuación: El codificador completa el procedimiento de caracteres recibidos hasta ese punto, transfiere cualesquiera códigos resultantes, actualiza el diccionario según proceda y establece la alineación de octetos. El decodificador establece subsiguientemente la alineación de octetos.

3.16 historia: La memoria tampón de historia, denominada en este documento "la historia", es la estructura de datos que contiene los caracteres introducidos en el codificador (o equivalentemente, decodificados por el decodificador) desde la reinicialización más reciente de las estructuras de datos. Los caracteres entran en la historia en el orden en que son recibidos (o decodificados).

3.17 método multipaquetes: Compresión de datos que están en paquetes (o en tramas) de modo que ambos extremos de la transmisión de datos puedan identificar los límites del paquete (o trama), y varios paquetes, o porciones de paquetes, son procesados como continuación. El método multipaquete se describe en el anexo B.

3.18 nodo: Punto situado en una estructura de árbol que representa un segmento de cadena. Corresponde a una palabra de código en el codificador.

3.19 árbol de nodos: Estructura de datos en el codificador que representa el conjunto de segmentos de cadena y sus interrelaciones. Combinado con la matriz de raíces, representa las estructuras de árbol del codificador. Una palabra de código corresponde con un determinado segmento de cadena y con un determinado nodo.

3.20 valor ordinal: El valor ordinal (u ordinal, para abreviar) de un carácter es el equivalente numérico del carácter.

3.21 método paquetes: Compresión de datos que están en paquetes (o en tramas) de modo que ambos extremos de la transmisión de datos puedan identificar los límites de los paquetes (o tramas), y cada paquete es procesado separadamente. El método paquetes se describe en el anexo B.

3.22 modo parámetros: Modo de funcionamiento en el cual los parámetros de compresión son transferidos entre el par de funciones de compresión de datos.

3.23 recepción: En la negociación de parámetros, el sentido correspondiente al decodificador de una entidad.

3.24 raíz: Punto en la base de una estructura de árbol que representa, dentro del contexto de la presente Recomendación, el primer carácter de una cadena (véanse la figura 2 y 6.2.1). Cada carácter en el alfabeto tiene una raíz correspondiente.

3.25 matriz de raíces: Estructura de datos que contiene el conjunto de todas las raíces del codificador.

3.26 método trenes: Compresión de datos que son transmitidos continuamente por un enlace con entrega garantizada. Este método es el punto central de UIT-T V.44.

3.27 cadena: Una cadena ininterrumpida de dos o más caracteres. En el diccionario del codificador, consiste en un primer carácter (raíz) seguido por uno o más segmentos de cadena.

3.28 extensión de cadena: Secuencia de uno o más caracteres que amplían una cadena para crear una cadena nueva más larga.

3.29 longitud de extensión de cadena: Código binario que indica el número de caracteres que amplían la cadena correspondiente a la palabra de código inmediatamente precedente.

3.30 segmento de cadena: Sección de una cadena que corresponde a una determinada palabra de código en el árbol de nodos del codificador.

3.31 conjunto de cadenas: Estructura de datos en el decodificador que representa las cadenas decodificadas.

3.32 transmisión: En la negociación de parámetros, el sentido correspondiente al codificador de una entidad.

3.33 modo transparente: Modo de funcionamiento en el cual los caracteres recibidos del DTE son transmitidos sin compresión. La función de compresión de datos está activa, pero no afecta la transmisión de datos.

3.34 estructura de árbol: Estructura de datos abstracta en el codificador que se utiliza en la presente Recomendación para representar un conjunto de cadenas, todas con el mismo carácter inicial (véanse la figura 2 y 6.2.1).

3.35 funcionamiento sin compresión: Estado del DCE en el cual la función de compresión de datos está inactiva.

3.36 carácter no concordado: Carácter que origina la terminación de un proceso de concordancia de cadena o de extensión de cadena porque no concuerda con un carácter específico en la historia. El carácter no concordado se convierte en el primer carácter de una posible nueva concordancia de cadena.

3.37 { }: {A} es la representación vectorial de un código binario. Tiene tantos componentes como bits haya en el código. El componente A_1 es el bit menos significativo del código binario.

4 Abreviaturas

En esta Recomendación se utilizan las siguientes siglas.

DCE Equipo de terminación del circuito de datos (*data circuit terminating equipment*)

DTE Equipo terminal de datos (*data terminal equipment*)

ECM Paso al modo con compresión: una instrucción de modo transparente definida en 7.2 (*enter compressed mode*)

EID ESCAPE en datos: una instrucción de modo transparente definida en 7.2 (*ESCAPE in data*)

EPM Paso al modo parámetros: una instrucción de modo transparente definida en 7.2 (*enter parameter mode*)

ETM Paso al modo transparente: código de control definido en 7.2 (*enter transparent mode*)

5 Descripción funcional de un DCE

A continuación se describe un DCE que emplea la función de compresión de datos.

5.1 Generalidades

Un DCE que emplea compresión de datos, según se ilustra en la figura 1, contiene los siguientes componentes:

a) circuitos de enlaces DTE/DCE;

b) un convertidor de señales;

c) una función de control;

d) una función de control de errores;

e) una función de compresión de datos.

5.2 Circuitos de enlace DTE/DCE

Este componente tendrá las capacidades descritas en UIT-T V.42.

5.3 Convertidor de señales

Este componente tendrá las capacidades descritas en UIT-T V.42.

5.4 Función de control

Este componente tendrá las capacidades descritas en 6.2/V.42. Además, ejecutará los siguientes aspectos de funcionamiento:

a) negociación de modos de funcionamiento de la función de compresión de datos con el DCE distante, y negociación de parámetros asociados con el funcionamiento de la función de compresión de datos;

b) instigación de la inicialización o reinicialización de la función de control de datos;

c) coordinación del establecimiento de una conexión con control de errores para ser utilizada por funciones de compresión de datos pares;

d) coordinación de la entrega de datos entre la interfaz DTE/DCE y la función de compresión de datos, de acuerdo con los procedimientos definidos en 6.2/V.42 y 8.4/V.42, incluida la provisión de los procedimientos de control de flujo definidos en la misma;

e) coordinación de la entrega de datos entre la función de compresión de datos y la función de control de errores;

f) acción al detectar una condición de excepción.

5.5 Función de control de errores

Este componente tendrá las capacidades descritas en UIT-T V.42.

5.6 Función de compresión de datos

La función de compresión de datos aplicará los procedimientos definidos en la presente Recomendación con el fin de codificar eficazmente los datos antes de su transmisión por una conexión con control de errores. Ejecutará los siguientes aspectos de la operación:

a) inicialización y reinicialización de los diccionarios del codificador y del decodificador;

b) codificación y decodificación de compresión de datos;

c) conmutación entre los modos de funcionamiento con compresión y transparente;

d) configuración del codificador y del decodificador de acuerdo con los parámetros negociados por la función de control.

6 Procedimientos de la función de compresión de datos

6.1 Visión general de la función de compresión de datos

La función de compresión de datos consiste en un codificador y un decodificador. En general, una conexión de datos soporta transmisión de datos en ambos sentidos, por lo que puede soportar compresión de datos también en ambos sentidos. De este modo, cada par de conexión de datos puede tener un codificador y un decodificador. El codificador transfiere datos comprimidos a su decodificador par en el otro extremo de la conexión y el decodificador descomprime los datos comprimidos recibidos de su codificador par. El par codificador-decodificador para cada sentido debe tener valores de parámetros de compresión de datos coordinados, establecidos mediante negociación; sin embargo, en cualquiera de los extremos los valores de parámetros del codificador y del decodificador pueden diferir, porque se refieren a sentidos de transmisión diferentes.

Los caracteres recibidos del DTE al codificador son concordados contra cualesquiera cadenas de caracteres identificadas previamente. Si una cadena concuerda, la palabra de código que representa la cadena es transferida, y después se intenta ampliar la cadena concordada para codificar caracteres adicionales y crear una nueva cadena más larga. Si no hay concordancia de cadena, se transfiere el valor ordinal correspondiente al primer carácter.

El codificador aumenta continuamente su conjunto de cadenas disponibles para concordancia, colocando caracteres de entrada en la historia y añadiendo nodos al árbol de nodos. Cuando cualquiera de éstos está lleno, el diccionario es reinicializado y la operación de codificación continúa como antes.

El decodificador crea cadenas que reproducen las cadenas creadas por su codificador par, empleando las mismas palabras de código asignadas. Descomprime el tren de datos comprimidos recibido utilizando estas cadenas. El diccionario del codificador es reinicializado al recibir un código de control REINIT.

6.2 Estructura del diccionario

Para describir en detalle el algoritmo de compresión de datos, es útil definir las siguientes estructuras de datos.

6.2.1 Diccionario del codificador

El codificador utiliza un diccionario que consta de tres partes:

1) *matriz de raíces*: Contiene una entrada para cada carácter en el alfabeto. El tamaño del alfabeto es N_4, y está determinado por el tamaño de los caracteres recibidos de la función de control: para un alfabeto de caracteres de 8 bits, hay 256 entradas. Cada entrada tiene un índice dado por su carácter y sirve de raíz para una estructura de árbol; contiene un índice

descendente a un nodo en el árbol de nodos. Este índice es el punto de partida para descender en la estructura de árbol.

2) *árbol de nodos*: Contiene las palabras de código creadas durante el funcionamiento del codificador, y los segmentos de cadena a los cuales corresponden. Los nodos en el árbol de nodos definen un conjunto de estructuras de árbol, y los correspondientes segmentos de cadena. Cada nodo contiene una palabra de código, la posición en la historia del primer carácter del segmento de cadena, el número de caracteres en el segmento de cadena, y dos índices para enlazar con otros nodos. El "índice descendente", si es válido, indica un nodo que representa un segmento de cadena que sigue a este segmento de cadena. El "índice lateral", si es válido, indica un nodo que representa un segmento de cadena en el mismo nivel que este segmento de cadena (es decir, que sigue también a la raíz o al segmento de cadena que precede a este segmento de cadena). El número de palabras de código no puede exceder de N_{2T}.

3) *historia*: Contiene todos los caracteres recibidos del codificador, en orden, desde la reinicialización más reciente del diccionario. Los segmentos de cadena en el árbol de nodos son referenciados por la posición de su primer carácter en la historia y por su longitud. El número de caracteres en la historia no puede exceder de N_{8T}.

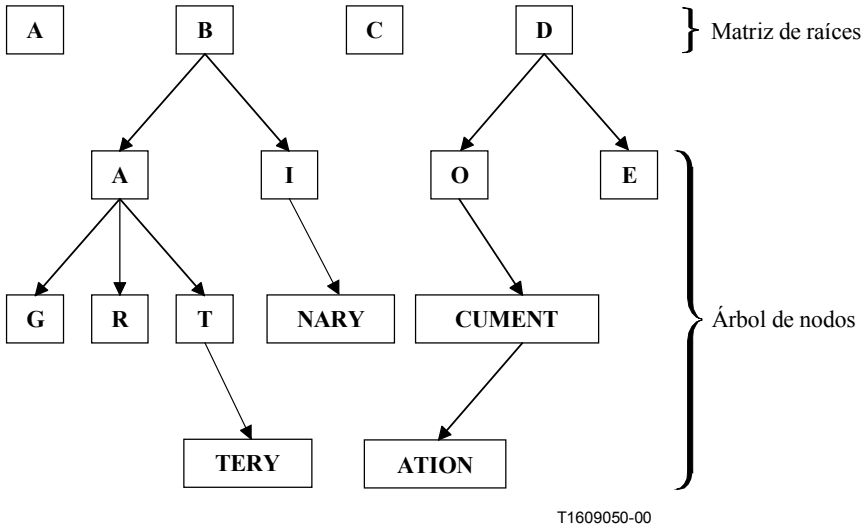

Figura 2/V.44 – Ejemplo de estructuras de árbol

Esta figura sólo muestra las estructuras de árbol que comienzan desde las raíces A, B, C y D.

Las estructuras de árbol de la figura 2 muestran los segmentos de cadena, representados cada uno por un nodo:

– A e I, que siguen al carácter raíz B.
– G, R y T que siguen al segmento de cadena A.
– TERY que sigue al segmento de cadena T.
– NARY que sigue al segmento de cadena I.
– O y E que siguen al carácter raíz D.
– CUMENT que sigue al segmento de cadena O.
– ATION que sigue al segmento de cadena CUMENT.

Las cadenas que están representadas son: BA, BAG, BAR, BAT, BATTERY, BI, BINARY, DO, DOCUMENT, DOCUMENTATION, DE. Una cadena completa es concordada comenzando en la

raíz que utiliza el siguiente carácter que ha de ser codificado y concordando segmentos de cadena sucesivos hacia abajo en el árbol utilizando caracteres introducidos ulteriormente.

6.2.2 Diccionario del decodificador

El decodificador utiliza un diccionario que consta de dos partes:

1) *conjunto de cadenas*: Define las cadenas creadas durante el proceso de decodificación correspondiente a los segmentos de cadena creados por el proceso de codificación. Cada entrada tiene una palabra de código, la posición en la historia del último carácter de la cadena, y la longitud total de la cadena. El número de palabras de código no puede exceder de N_{2R}.
2) *historia*: Contiene todos los caracteres decodificados, en orden, desde la reinicialización más reciente del diccionario. Es idéntica a la historia del codificador, pero se genera utilizando los caracteres resultantes del proceso de decodificación. El número de caracteres en la historia no puede exceder de N_{8R}.

6.3 Codificación

El algoritmo de codificación intenta hallar una cadena de caracteres procesados previamente que concuerda con los siguientes caracteres que se han de procesar; si se encuentra, la palabra de código que representa la cadena es transferida. A continuación, el decodificador, al recibir la palabra de código puede regenerar correctamente esta cadena. La compresión de datos es satisfactoria si la cadena codificada requiere transferir menos bits que los caracteres originales no comprimidos.

Los caracteres recibidos son colocados en los siguientes lugares disponibles en la historia y son procesados por el codificador como sigue:

a) Utilizando el carácter recibido y el siguiente o siguientes caracteres, el codificador intenta hallar la cadena más larga que concuerda en el diccionario.
b) Si no se encuentra una concordancia de cadena, el codificador transfiere el ordinal correspondiente al carácter de entrada y vuelve al paso a) utilizando el siguiente carácter.
c) Si se encuentra una concordancia de cadena, el codificador transfiere la palabra de código asignada al último segmento de cadena completamente concordado de la concordancia de cadena más larga.
d) Después el codificador intenta ampliar la concordancia de cadena más larga, hasta la longitud de cadena máxima. Intenta concordar el siguiente o siguientes caracteres con el carácter o caracteres en la historia que sigue inmediatamente al último carácter del último segmento de cadena completamente concordado de la concordancia de cadena más larga.
e) Si la concordancia de cadena más larga es ampliada satisfactoriamente con uno o más caracteres, el codificador transfiere una longitud de extensión de cadena que indica el número de caracteres con que se ha ampliado la cadena.

Utilizando el carácter no concordado (es decir, el primer carácter que no formó parte de la concordancia de cadena ni de la extensión) como el siguiente carácter, el codificador vuelve al paso a).

6.3.1 Procedimiento de concordancia de cadena

Una cadena en el diccionario del codificador consta de un primer carácter seguido por uno o más segmentos de cadena del árbol de nodos; por ejemplo, en la figura 2, la cadena "BATTERY" tiene el primer carácter "B", seguido por el segmento de cadena "A", seguido por el segmento de cadena "T", seguido por el segmento de cadena "TERY". La longitud total de la cadena no puede exceder de N_{7T}.

El procedimiento de concordancia de cadena utiliza la estructura del diccionario descrita en 6.2 para hallar una concordancia de cadena más larga, como sigue:

– Cada raíz en la matriz de raíces tiene un índice descendente: si se pone a un índice válido, el nodo correspondiente proporciona el primer segmento de cadena para la comparación.

– Cada nodo en el árbol de nodos tiene un índice lateral utilizado por el codificador para buscar otro segmento de cadena que también deriva del mismo nodo o raíz anterior (otro nodo en el mismo nivel).

– Cada nodo en el árbol de nodos tiene un índice descendente utilizado por el codificador con el fin de buscar un segmento de cadena para continuar la comparación (un nodo en el siguiente nivel).

Al buscar una concordancia de cadena, el codificador examina la raíz correspondiente al primer carácter recibido, y compara después los caracteres recibidos con los segmentos de cadena correspondientes a nodos que se derivan directamente de esa raíz. Si los caracteres introducidos consecutivamente concuerdan exactamente con cualquier segmento de cadena completo (las concordancias parciales no son válidas), el codificador compara los caracteres introducidos adicionales con los segmentos de cadena de los nodos derivados de ese nodo. De este modo, el codificador continúa la estructura de árbol descendente hasta que no puede hallar ninguna otra concordancia o hasta que no hay más nodos derivados.

El procedimiento de concordancia de cadena falla si la raíz correspondiente al primer carácter no tiene un índice descendente válido, o si ningún nodo de segundo nivel concuerda con el segundo carácter. En este caso, el ordinal correspondiente al primer carácter es transferido, y el procedimiento de concordancia de cadena comienza de nuevo con el siguiente carácter (no concordado).

En los demás casos, el procedimiento de concordancia de cadena encuentra un nodo que corresponde con la cadena más larga del diccionario que concuerda con los caracteres introducidos. El codificador transfiere la palabra de código para este nodo. (Obsérvese que la palabra de código en realidad corresponde directamente con el último segmento de cadena completamente concordado de la concordancia de cadena más larga, pues el codificador no asocia explícitamente palabras de código con cadenas completas.)

El codificador puede transferir cualquier palabra de código que haya creado. Si el valor de la palabra de código es igual al valor del decodificador de C_1, esto indica una palabra de código que no ha sido creada aún en el decodificador; no obstante, el decodificador tiene que interpretar correctamente todos los valores de palabra de código hasta C_1. Véase 6.4.1.

El procedimiento de concordancia de cadena puede ser terminado también por los siguientes motivos: la historia está llena, o se recibe una petición EVACUACIÓN. El procedimiento es similar al que se produce cuando el siguiente carácter introducido no concuerda con ningún carácter en el segmento de cadena vigente: la palabra de código para el último segmento de cadena completamente concordado (si hubiera alguno) es transferida; después si hay un segmento de cadena parcialmente concordado, los caracteres concordados satisfactoriamente son transferidos como una longitud de extensión de cadena, y se crea la correspondiente entrada de árbol.

NOTA 1 – Para que un segmento de cadena concuerde con los caracteres introducidos, todos los caracteres deben concordar: las concordancias parciales no son válidas. Sin embargo, se puede utilizar una concordancia parcial de un segmento de cadena para extender la cadena, si no se encuentra concordancia de segmento de cadena completa en el mismo nivel. En este caso, el procedimiento de concordancia de cadena ha efectuado ya las comparaciones necesarias para hallar la extensión de cadena más larga posible especificada en 6.3.2, y este número es precisamente el número de caracteres concordados del segmento de cadena concordado parcialmente. Por consiguiente, no es necesario iniciar el procedimiento de concordancia de cadena explícitamente, pues el codificador puede transferir la longitud de extensión de cadena correspondiente al número de caracteres concordados del segmento de cadena parcialmente concordado.

NOTA 2 – Es conveniente ordenar en un nivel los nodos bifurcados de un nodo determinado, alfabéticamente por el primer carácter, para buscar más eficazmente la concordancia de segmentos de cadena. Puede haber múltiples segmentos de cadena, cada uno con el primer carácter igual, bifurcados de un nodo.

6.3.2 Procedimiento de extensión de cadena

Si la concordancia de cadena tiene éxito (es decir, se halla la concordancia de cadena más larga), y si la longitud total de la cadena es menor que N_{7T}, el codificador inicia el procedimiento de extensión de cadena. Intenta extender la concordancia de cadena más larga comparando el siguiente o los siguientes caracteres con el carácter o los caracteres en la historia que siguen inmediatamente al último carácter del último segmento de cadena completamente concordado.

Si se recibe una C-EVACUACIÓN de la función de control justo después de una concordancia exitosa y antes de que el procedimiento de extensión de cadena haya procesado un carácter, se termina el procedimiento de extensión de cadena. El carácter que sigue inmediatamente a la C-EVACUACIÓN, cuando se recibe, comienza una nueva concordancia de cadena y se actualiza el diccionario como si ese carácter no hubiera sido concordado.

Por ejemplo, se supone que la cadena "BATTERY" es la concordancia de cadena más larga. El codificador compara el siguiente carácter con el carácter en la historia que sigue al segmento de cadena "TERY". Hay dos posibilidades:

- El siguiente carácter no concuerda con el carácter en la historia que sigue inmediatamente a la "Y" del segmento de cadena "TERY". El procedimiento de extensión de cadena termina negativamente, y este siguiente carácter es anexado a "TERY" para crear una nueva cadena más larga, y se convierte también en el primer carácter de una nueva posible concordancia de cadena.
- El siguiente carácter concuerda con el carácter en la historia que sigue inmediatamente a la "Y" del segmento de cadena "TERY". El procedimiento de extensión de cadena continúa comparando el siguiente carácter recibido con el segundo carácter en la historia después de la "Y" del segmento de cadena "TERY", el siguiente carácter recibido con el tercer carácter en la historia que sigue a la "Y", y así sucesivamente. El número de caracteres satisfactoriamente concordados es transferido como una longitud de extensión de cadena. El codificador crea un nuevo segmento de cadena en el árbol de nodos, que consiste en la extensión de cadena (la secuencia de caracteres concordados después de la concordancia de cadena más larga). Se asigna a este segmento la siguiente palabra de código disponible, y tiene una longitud de segmento de cadena igual a la longitud de la extensión. Su índice de historia hace referencia a la posición en la historia del primero de los caracteres recibidos más recientemente utilizados para ampliar la cadena.

El procedimiento de extensión de cadena puede ser terminado también por uno de los siguientes motivos: la historia está llena, una petición EVACUACIÓN, la longitud de la cadena total ha alcanzado N_{7T}. El procesamiento es similar al caso cuando el siguiente carácter recibido no concuerda con el siguiente carácter en la historia: cualesquiera caracteres que han sido concordados satisfactoriamente para ampliar la cadena son transferidos como una longitud de extensión de cadena, y se crea una correspondiente entrada del árbol de nodos.

6.3.3 Creación de segmentos de cadena

El codificador crea continuamente nuevas cadenas para posibles futuras concordancias de cadena. Los dos métodos para crear nuevos segmentos de cadena son:

Anexar

Fallo de concordancia de cadena:

– Si la concordancia de cadena falla porque el primer carácter no tiene un índice descendente válido en la matriz de raíces, se crea un segmento de cadena de 1 carácter que enlaza el siguiente carácter con la raíz del primer carácter.

– Si la concordancia de cadena falla porque el siguiente carácter no concuerda con ninguno de los nodos existentes derivados de la primera raíz de carácter, se crea un segmento de cadena

de 1 carácter que utiliza el carácter no concordado, que se deriva de la raíz del primer carácter.

Fallo de extensión de cadena:

– Se crea el segmento de cadena de 1 carácter que enlaza el carácter no concordado con el último segmento de cadena completamente concordado de la concordancia de cadena más larga.

Extender

Extensión de cadena con éxito:

Se crea un segmento de cadena que amplía la concordancia de cadena más larga con uno o más caracteres. El segmento de cadena así creado tiene un longitud igual el número de caracteres de la ampliación, y está enlazado al último segmento de cadena completamente concordado.

6.3.4 Resumen de codificación

Cuadro 1/V.44 – Procedimientos de codificación

Procedimiento	Resultado	Código transferido	Nuevo segmento de cadena	Uso de carácter no concordado
Concordancia de cadena	Fallo	Valor ordinal correspondiente al primer carácter	Anexar carácter no concordado a la raíz del primer carácter	Como primer carácter para la concordancia de cadena
Concordancia de cadena	Éxito	Palabra de código para último segmento de cadena completamente concordado	Ninguno	Para extensión de cadena
Extensión de cadena	Fallo	Nada	Anexar carácter no concordado al último segmento de cadena completamente concordado	Como primer carácter para la concordancia de cadena
Concordancia de cadena	Concordancia de cadena	Longitud de extensión de cadena	Todos los caracteres de extensión, enlazados al último segmento de cadena completamente concordado	Como primer carácter para la concordancia de cadena

El cuadro 1 resume los procedimientos de codificación y el apéndice II ilustra el funcionamiento del algoritmo explícitamente desde el punto de vista de la estructura del diccionario.

6.4 Decodificación

Una cadena en el decodificador viene definida por la posición en la historia del último carácter de la cadena, y por la longitud total de la cadena. Corresponde con una determinada palabra de código que, por construcción, corresponde en el codificador al segmento de cadena final de esa cadena.

En el modo con compresión, los códigos binarios recibidos por el decodificador son analizados en códigos de control, ordinales, palabras de código y longitudes de extensión de cadena, examinando los prefijos de código. El decodificador debe permanecer sincronizado con las cadenas y palabras de código que son creadas por el codificador, actualizando el conjunto de cadenas y la historia sobre la

base de los códigos recibidos. El decodificador es más sencillo que el codificador, porque el codificador debe buscar las concordancias de cadena y crear cadenas nuevas y más largas, mientras que el decodificador sólo tiene que seguir la pista de las cadenas que ha creado.

El decodificador será capaz de funcionar en los modos transparente y con compresión.

6.4.1 Procesamiento de códigos

En el modo con compresión, la función de decodificación funcionará como sigue:

1) Al recibir un ordinal, el decodificador colocará el carácter correspondiente en la salida descomprimida y en la historia.

2) Al recibir una palabra de código menor que C_1, el decodificador copiará la cadena representada por la palabra de código en la salida descomprimida y en la historia.

3) Al recibir una palabra de código igual a C_1, si el código previo recibido fue una palabra de código, el decodificador copiará la cadena representada por esa palabra de código anterior en la salida descomprimida y en la historia, y después colocará el primer carácter de la cadena representado por esa palabra de código previa en la salida descomprimida y en la historia.

4) Al recibir una palabra de código igual a C_1, si el código previo recibido era un ordinal, el decodificador procesará el carácter correspondiente de acuerdo con el paso 3) anterior, como si fuese una cadena de longitud 1.

5) Al recibir una longitud de extensión de cadena, el decodificador utilizará la palabra de código precedente para acceder a los caracteres en la historia inmediatamente después de la cadena representada por esa palabra de código. El número de caracteres indicados por la longitud de extensión de cadena son copiados en la salida descomprimida y en la historia.

6) La recepción de una palabra de código mayor que C_1 constituye un error de procedimiento (véase 7.15).

6.4.2 Creación de nuevas cadenas

Las reglas para crear nuevas cadenas son las siguientes:

Cuadro 2/V.44 – Reglas para la creación de cadenas

Código vigente	Código anterior	Nueva cadena creada
Ordinal	Ordinal	El correspondiente carácter es anexado al carácter previo para crear una cadena de 2 caracteres
Ordinal	Palabra de código	El correspondiente carácter es anexado a la cadena previa para crear una cadena más larga (nota)
Ordinal	Longitud de extensión de cadena	El correspondiente carácter no es anexado a la cadena ampliada
Ordinal	Código de control FLUSH	El correspondiente carácter se trata como si la EVACUACIÓN no hubiera ocurrido. Se actúa como si el código que precede al de FLUSH fuese el "código anterior"
Ordinal	REINIT, ECM o inicialización	Ninguna: El diccionario está vacío
Palabra de código	Ordinal	El primer carácter de la cadena de palabra de código es anexado al carácter previo para crear una cadena de 2 caracteres

Cuadro 2/V.44 – Reglas para la creación de cadenas *(fin)*

Código vigente	Código anterior	Nueva cadena creada
Palabra de código	Palabra de código	El primer carácter de la cadena de palabra de código es anexado a la cadena anterior para crear una cadena más larga (nota)
Palabra de código	Longitud de extensión de cadena	El primer carácter de la cadena de palabra de código no es anexado a la cadena ampliada
Palabra de código	Código de control FLUSH	El primer carácter de la cadena de la palabra de código se trata como si la EVACUACIÓN no hubiera ocurrido. Se actúa como si el código que precede al de FLUSH fuera el "código precedente"
Longitud de extensión de cadena	Palabra de código	Se amplía la cadena anterior para crear una nueva cadena más larga (nota)
NOTA – No se crearán cadenas de longitud mayor que N_{7R}.		

La aparición de un código de control STEPUP no afecta la creación de cadenas.

6.5 Modo transparente

A veces la tara de funcionamiento del modo con compresión puede ser superior a la economía lograda con la compresión. Cuando se produce esta situación, el codificador puede pasar al modo transparente para transferir los caracteres de entrada no codificados, sin requerir una nueva conexión.

Mientras se está en funcionamiento el modo transparente, la actividad de codificación puede continuar, para determinar cuándo sería provechoso retornar al modo con compresión, pero la salida del codificador no es transferida. En cambio, la entrada de caracteres al codificador es transferida sin modificación, alineada en octetos. El codificador reinicializará el diccionario al dejar el modo transparente y volver al modo con compresión.

En el modo transparente, el decodificador procesará los caracteres transparentes y las instrucciones pero no actualizará la historia ni el conjunto de cadenas. El diccionario del decodificador pasará a un estado compatible con el diccionario del codificador par, mediante reinicialización al dejar el modo transparente.

6.5.1 Transición del modo con compresión al modo transparente

En el modo con compresión, si el codificador determina que el tren de datos no es comprimible:

a) asegurará que todos los caracteres recibidos hasta ese punto son transferidos de acuerdo con los procedimientos de codificación normales;

b) transferirá el código de control ETM (paso al modo transparente) para indicar una transición al modo transparente al decodificador par;

c) transmitirá suficientes bits 0 para establecer la alineación en octetos (véase el cuadro 6);

d) pasará al modo transparente.

En el modo con compresión, al recibir el código de control ETM, el decodificador pasará al modo transparente.

6.5.2 Transición del modo transparente al modo con compresión

En el modo transparente, si el codificador determina que el tren de datos es comprimible:

a) transferirá todos los caracteres recibidos hasta este punto, en modo transparente;

b) reinicializará el diccionario del codificador;

c) transferirá el valor vigente de ESCAPE para indicar una secuencia de instrucción;

d) transferirá la instrucción ECM;

e) pasará al modo con compresión.

En el modo transparente, al recibir un ESCAPE seguido inmediatamente por una instrucción ECM, el decodificador reinicializará el diccionario y pasará al modo comprimido.

6.6 Transferencia

En el modo transparente, los caracteres serán transferidos a la función de control para transmisión alineados en octetos, utilizando una primitiva de indicación C-TRANSFERENCIA. Pueden ser transferidos individualmente durante los procedimientos de concordancia de cadena o extensión de cadena, o como una secuencia después de la compleción de dichos procedimientos.

En el modo con compresión, la salida de códigos binarios del codificador (véase 6.3) será transferida a la función de control. El bit menos significativo del prefijo de código binario seguirá inmediatamente al bit menos significativo del código binario precedente. Cada código es precedido por un prefijo de código.

Después de la transición del modo transparente al modo con compresión, el bit menos significativo del prefijo de código para el primer código binario que ha de ser transferido será el bit 1 del primer octeto comprimido.

En la transición del modo con compresión al modo transparente, se transferirán suficientes bits 0 para asegurar que el siguiente carácter transmitido está alineado en octetos.

Después de un código de control FLUSH, se transferirán suficientes bits 0 para asegurar que el siguiente código transmitido está alineado en octetos.

6.6.1 Transferencia de códigos de control, ordinales y palabras de código

Los códigos de control son transferidos utilizando el número de bits definido por el valor vigente de C_2.

Los ordinales son transferidos utilizando el número de bits definido por el valor vigente de C_5.

Las palabras de código son transferidas utilizando el número de bits definido por el valor vigente de C_2.

6.6.2 Transferencia de longitud de extensión de cadena

Una longitud de extensión de cadena es un código que representa la longitud de la extensión de la cadena representada por la palabra de código anterior. Una longitud de extensión de cadena consta de uno o más subcampos, cada uno de los cuales será analizado individualmente por el decodificador. El número de subcampos en una longitud de extensión de cadena depende del número de caracteres (es decir, la longitud) de la extensión de cadena. La longitud de extensión de cadena varía de 1 a 253 ó ($N_{7T} – 2$), el que sea menor. La codificación de los subcampos para extensiones de cadena con longitud de 1 a 12 se muestra en el cuadro 3; en aras de la claridad, se muestra también el orden de transmisión de los bits.

Cuadro 3/V.44 – Longitud de extensión de cadena: longitudes de 1 a 12

Longitud	Subcampos	Orden de transmisión de bits (de izquierda a derecha)
1	"1"	1
2	"0" "01"	0 10
3	"0" "10"	0 01
4	"0" "11"	0 11
5	"0" "00" "0" "000"	0 00 0 000
6	"0" "00" "0" "001"	0 00 0 100
7	"0" "00" "0" "010"	0 00 0 010
8	"0" "00" "0" "011"	0 00 0 110
9	"0" "00" "0" "100"	0 00 0 001
10	"0" "00" "0" "101"	0 00 0 101
11	"0" "00" "0" "110"	0 00 0 011
12	"0" "00" "0" "111"	0 00 0 111

Para longitudes de extensión de cadenas superiores a 12, la codificación depende del valor de la longitud de cadena máxima N_{7T}, como se muestra en el cuadro 4. El valor de N_{7T} determina el número de bits en el último subcampo, mientras que la longitud de extensión de cadena determina el valor del último subcampo:

$$N = \text{longitud de-extensión de cadena} - 13$$

y

$$N_x = \text{x-ésima cifra binaria de } N$$

Cuadro 4/V.44 – Longitud de extensión de cadena: longitudes de 13 a 253

Parámetro de longitud de cadena máxima N_{7T}	Gama de longitudes	Subcampos	Orden de transmisión de bits (de izquierda a derecha)
$32 \le N_{7T} \le 46$	13-44	"0" "00" "1" "$N_4N_3N_2N_1N_0$"	0 00 1 $N_0N_1N_2N_3N_4$
$46 < N_{7T} \le 78$	13-76	"0" "00" "1" "$N_5N_4N_3N_2N_1N_0$"	0 00 1 $N_0N_1N_2N_3N_4N_5$
$78 < N_{7T} \le 142$	13-140	"0" "00" "1" "$N_6N_5N_4N_3N_2N_1N_0$"	0 00 1 $N_0N_1N_2N_3N_4N_5N_6$
$142 < N_{7T} \le 255$	13-253	"0" "00" "1" "$N_7N_6N_5N_4N_3N_2N_1N_0$"	0 00 1 $N_0N_1N_2N_3N_4N_5N_6N_7$
NOTA – La longitud máxima de una extensión de cadena es ($N_{7T} - 2$), porque la longitud mínima de una cadena es 2 caracteres.			

6.6.3 Prefijos de código

Los prefijos de código son transferidos inmediatamente antes de sus códigos y se muestran en el cuadro 5.

Cuadro 5/V.44 – Utilización de prefijos de código

Tipo de código	**Prefijo: inmediatamente después de una palabra de código**		**Prefijo: en los demás casos (Nota)**	
	Subcampos	*Orden de transmisión de bits (de izquierda a derecha*)	*Subcampos*	*Orden de transmisión de bits (de izquierda a derecha*)
Código de control	"1"	1	"1"	1
Ordinal	"0" "0"	0 0	"0"	0
Palabra de código	"1"	1	"1"	1
Longitud de extensión de cadena	"0" "1"	0 1	n/a	n/a
NOTA – Inmediatamente después de un código de control, longitud de extensión de cadena, o después de reinicialización.				

6.6.4 Ejemplo de transferencia

El cuadro 6 ilustra una secuencia de octetos comprimidos transferidos por el codificador. En este ejemplo, varios códigos son transferidos en el modo con compresión, y después el codificador hace una transición al modo transparente. En el ejemplo se supone que el valor del tamaño de palabra de código vigente C_2 es 11 y que el valor del tamaño del ordinal vigente C_5 es 8 (véase el cuadro 12). Los códigos son:

– palabra de código {A} con el prefijo "1";
– ordinal {B} con el prefijo "0" "0" (porque sigue inmediatamente a una palabra de código);
– ordinal {C} con prefijo "0";
– palabra de código {D}con prefijo "1";
– longitud de extensión de cadena 8: prefijo "0" "1", subcampos "0" "00" "0" "011";
– ordinal {E} con prefijo "0";
– código de control ETM con prefijo "1". El valor de ETM es "00000000" que se amplía a 11 bits;
– relleno: cinco 0 para establecer alineación en octetos i + 9;
– carácter {F} en modo transparente, alineado en octetos.

Cuadro 6/V.44 – Correspondencia de octetos para un tren de datos de ejemplo

Bit N.º	8	7	6	5	4	3	2	1	Octeto	Modo
	A_5	A_4	A_3	A_2	A_1	***1***	...	...	i	con compresión
	0	***0***	A_{11}	A_{10}	A_9	A_8	A_7	A_6	i + 1	con compresión
	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	i + 2	con compresión
	C_7	C_6	C_5	C_4	C_3	C_2	C_1	***0***	i + 3	con compresión
	D_6	D_5	D_4	D_3	D_2	D_1	***1***	C_8	i + 4	con compresión
	0	***1***	***0***	D_{11}	D_{10}	D_9	D_8	D_7	i + 5	con compresión
	E_1	***0***	0	1	1	0	0	0	i + 6	con compresión
	1	E_8	E_7	E_6	E_5	E_4	E_3	E_2	i + 7	con compresión
	0	0	0	0	0	0	0	0	i + 8	con compresión
	0	0	0	0	0	0	0	0	i + 9	con compresión
	F_8	F_7	F_6	F_5	F_4	F_3	F_2	F_1	i + 10	transparente
Los bits de prefijo de código se muestran en ***itálicas negritas*** y los bits de relleno están subrayados.										

7 Operaciones de compresión de datos

7.1 Comunicación entre las funciones de control y de compresión de datos

La comunicación entre la función de control y la función de compresión de datos se modela como un conjunto de primitivas abstractas de la forma X-NAME_type que representan el intercambio lógico de información y control para ejecutar una tarea o servicio. En el contexto de la presente Recomendación, la función de control se considera como el "usuario del servicio" y la función de compresión de datos como el "proveedor del servicio". Los tipos de primitivas son: petición, indicación, respuesta y confirmación. Los servicios utilizados en la presente Recomendación se enumeran en el cuadro 7.

Cuadro 7/V.44 – Servicios previstos por la función de control

Servicio	Primitiva	Referencia
Inicializar la función de compresión de datos	C-INICIALIZACIÓN	7.5
Transferir parámetros de negociación a/de la función de compresión de datos	C-PARÁMETRO	7.4.2
Indicar un error a la función de control	C-ERROR	7.15
Transferir datos no comprimidos a/de la función de compresión de datos	C-DATOS	7.7
Transferir datos comprimidos a/de la función de compresión de datos	C-TRANSFERENCIA	7.9
Evacuar datos no transferidos restantes del codificador	C-EVACUACIÓN	7.13

7.2 Comunicación entre funciones de compresión de datos pares

Los códigos de control y las instrucciones utilizados para la comunicación entre funciones de compresión de datos pares se muestran en los cuadros 8 y 9.

Cuadro 8/V.44 – Códigos de control utilizados en el modo con compresión

Código	Nombre	Descripción
0	ETM	Pasar al modo transparente
1	FLUSH	Evacuar datos
2	STEPUP	Aumentar 1 al tamaño de palabra de código o tamaño de ordinal
3	REINIT	Forzar reinicialización de diccionarios

Cuadro 9/V.44 – Instrucciones utilizadas en el modo transparente

Valor	Nombre	Descripción
0	ECM	Pasar al modo con compresión
1	EID	ESCAPE en datos
2	EPM	Pasar al modo parámetro
Variable: véase 7.14	ESCAPE	Iniciar una secuencia de instrucciones

7.3 Negociación de la capacidad V.44

Antes de iniciar la función de compresión de datos, los pares de módems deben determinar si tienen la capacidad V.44. Esta capacidad será determinada en el establecimiento del enlace mediante un protocolo (por ejemplo, el procedimiento XID definido en UIT-T V.42), y permanecerá en efecto durante la conexión con corrección de errores. En particular, cuando se utiliza esta Recomendación con el control de errores V.42, se utilizará el procedimiento de negociación XID (véanse ISO/CEI 13239 y 7.7/V.42, 8.10/V.42 y cláusula 10/V.42). Los parámetros dentro del subcampo de datos de usuario, además de los definidos en UIT-T V.42 se utilizarán para este fin. El subcampo de datos de usuario aparecerá en la trama XID inmediatamente antes que la secuencia de verificación de trama (FCS) y será codificado como se indica en el cuadro A.1.

La presencia del parámetro "capacidad V.44" dentro de un mensaje XID (u otro mensaje de protocolo) indica que la entidad emisora tiene la capacidad descrita en la presente Recomendación. El formato del parámetro capacidad V.44 para un mensaje XID cuando se utilizan procedimientos de corrección de errores V.42 se describe en el anexo A. Cabe señalar que determinados bits dentro de este parámetro deben ser pasados por alto mientras se negocia por conexiones con módems, pues especifican capacidades V.44 para uso en redes de paquetes, según se describe en el anexo B.

NOTA – Durante la fase de establecimiento del protocolo, la presencia de tipo de parámetro 0x40 con un identificador de conjunto de parámetros de "V.44" en el subcampo de datos de usuario de la trama XID indicará la petición de compresión de datos V.44. El respondedor incluirá parámetros para un algoritmo de compresión como máximo (V.42 *bis* o V.44) en la respuesta XID.

7.4 Negociación de parámetros de compresión de datos

Una vez determinada la capacidad V.44, los valores de los parámetros de compresión de datos pueden ser negociados entre las funciones pares de compresión de datos. Estos parámetros se definen en el cuadro 10 de la cláusula 8: los parámetros "P" son los valores propuestos por cada lado de la negociación y los parámetros "N" son los valores finalmente acordados.

Los parámetros de compresión de datos pueden diferir para los dos sentidos de transmisión. Durante la negociación, cada lado de la conexión puede proponer valores para cualquiera de estos parámetros: el subíndice "T" indica que el parámetro hace referencia al sentido de transmisión (codificador) de la entidad que envía el mensaje; el subíndice "R" indica que el parámetro hace referencia al sentido recepción (decodificador) de la entidad que envía el mensaje.

De este modo, al negociar los sentidos en los cuales funcionará la compresión de datos, la respuesta complementaria a un valor de 01 P_0 de una entidad (que propone funcionar solamente en el sentido del codificador de la entidad emisora y el decodificador de la entidad respondedora) es un valor P_0 de 10 [(se acuerda funcionar solamente en el sentido del decodificador de la entidad respondedora y el codificador de la entidad emisora (original)]. Una respuesta de 01 u 11 propondría inadecuadamente compresión de datos en un sentido no solicitado originalmente; una respuesta de 00 resultará en que no habrá compresión de datos en ninguna de los dos sentidos.

Al negociar valores de parámetros, se utiliza el valor menor de dos propuestas. Por ejemplo, P_{2T} propuesto de una entidad se compara con P_{2R} propuesto de la otra entidad; si ambos valores son válidos, se utilizará el valor menor; cualquier intento de especificar un valor menor que el mínimo es un error de procedimiento y resultará en desconexión. El valor final acordado es fijado en N_{7T} por la entidad que propone P_{2T} y en N_{7R} por la entidad que propone P_{2R}.

El apéndice I proporciona orientación sobre las repercusiones de los valores de estos parámetros en el funcionamiento de la compresión de datos.

Estos valores pueden ser negociados de dos maneras, como se indica en el valor del parámetro capacidad V.44: a través de la propia determinación de la capacidad XID o mediante negociación después del establecimiento del enlace. La negociación a través de XID es la negociación por defecto. Un módem que reciba una XID en la que se pide la negociación de parámetros XID debe responder con una XID con parámetros V.44; por consiguiente, si un módem recibe una negociación del modo parámetros en la secuencia XID, pero ha recibido una petición de negociación de XID, debe responder con una XID con parámetros V.44.

7.4.1 Negociación a través de XID

En el subcampo de datos de usuario de XID, el parámetro capacidad V.44 indica cómo se han de negociar los parámetros de compresión de datos; si indica que se han de utilizar los procedimientos XID, los valores propuestos por la entidad emisora son incluidos ulteriormente en el mismo subcampo.

7.4.2 Negociación después del establecimiento del enlace

Si el parámetro capacidad V.44 indica que la negociación de parámetro se ha de efectuar después que se ha establecido el enlace con corrección de errores, la función de control ejecuta este establecimiento del enlace y en cualquier momento ulterior, según lo considere necesario. La función de control asegurará que la transferencia de datos entre las funciones pares de compresión de datos no está en curso, y que el control de flujo está en efecto, antes de iniciar la negociación de parámetros.

La función de control utiliza las primitivas C- PARÁMETRO para pasar parámetros a/desde la función de compresión de datos, como sigue:

- Al recibir una petición C-PARÁMETRO de la función de control, el codificador:
  - pasará del modo con compresión al modo transparente utilizando los procedimientos indicados en 6.5.1; o permanecerá en el modo transparente;
  - pasará al modo parámetro y transferirá la instrucción EPM (pasar al modo parámetro) para indicar esta transición al decodificador par;
  - transferirá los parámetros pasados con la petición C-PARÁMETRO al decodificador par;
  - transferirá el parámetro End (Fin) (véase el cuadro 10) para indicar una transición al modo transparente al decodificador par;
  - emitirá una confirmación C-PARÁMETRO a la función de control;
  - pasará al modo transparente.

- Al recibir una confirmación C-PARÁMETRO de la función de compresión de datos, la función de control:
  - si los parámetros que acaban de ser transferidos al decodificador par completan satisfactoriamente la negociación de parámetros, la función de control reinicializará la función de compresión de datos de acuerdo con los procedimientos definidos en 7.5;
  - en los demás casos, la función de control continuará a mantener el control de flujo hasta recibir parámetros de la función par de compresión de datos.
- Al recibir una instrucción EPM del codificador par mientras está en modo transparente, el decodificador:
  - pasará al modo parámetro;
  - recibirá los parámetros;
  - pasará al modo transparente cuando se recibe el parámetro fin;
  - emitirá una indicación C-PARÁMETRO a la función de control pasándole los parámetros recibidos.
- Al recibir una indicación C-PARÁMETRO de la función de compresión de datos, la función de control:
  - si los parámetros pasados con la indicación C-PARÁMETRO completan satisfactoriamente la negociación de parámetros, la función de control reinicializará la función de compresión de datos de acuerdo con los procedimientos definidos en 7.5;
  - en los demás casos, la función de control emitirá una petición C-PARÁMETRO para transferir los parámetros respondedores a su función par de compresión de datos.

7.5 Inicialización de la función de compresión de datos

Tras la negociación satisfactoria de la capacidad V.44, la función de control emitirá la primitiva petición C-INICIALIZACIÓN a la función de compresión de datos. Si los parámetros de compresión de datos V.44 han sido negociados en el intercambio de mensajes XID, la primitiva C-INICIALIZACIÓN indicará el valor negociado de los parámetros; en los demás casos, esta primitiva indicará el valor por defecto de los parámetros según se define en el cuadro 10. La función de compresión de datos inicializará el diccionario del codificador al estado definido en 7.5.1, y el diccionario del decodificador al estado definido en 7.5.2. La función de compresión de datos se pondrá al modo con compresión, y se asigna el valor 0 a ESCAPE.

Una reinicialización de la función de compresión de datos puede ser invocada por la función de control, que emitirá una petición C-INICIALIZACIÓN a la función de compresión de datos en las condiciones siguientes:

- recepción de una indicación L-ESTABLECIMIENTO o una confirmación L-ESTABLECIMIENTO;
- recepción de una indicación o confirmación L-SEÑAL, donde la primitiva indica una forma destructiva. Véase el cuadro 4/V.42;
- recepción de la indicación C-PARÁMETRO que completa satisfactoriamente la negociación de parámetros iniciada por esta entidad. La primitiva C-INICIALIZACIÓN indicará el valor recién negociado de los parámetros; o
- recepción de confirmación C-PARÁMETRO que completa satisfactoriamente la negociación de parámetros iniciada por la función par de compresión de datos. La primitiva C-INICIALIZACIÓN indicará el valor negociado de los parámetros.

Es responsabilidad de la función de control asegurar que las primitivas de petición C-INICIALIZACIÓN sólo son emitidas cuando no hay datos en tránsito entre las funciones de

compresión de datos (por ejemplo, en las funciones de control de errores), para asegurar la sincronización entre los codificadores y decodificadores.

7.5.1 Estado inicial del diccionario del codificador

Los valores iniciales de las variables del codificador son:

– la "siguiente palabra de código $C_1 = N_5$;

– el "tamaño de palabra de código vigente" $C_2 = 6$;

– el "umbral para cambiar el tamaño de palabra de código" $C_3 = 64$;

– la "posición de historia vigente" $C_4 = 0$;

– el "tamaño de ordinal vigente" $C_5 = 7$;

– todos los índices descendentes en la matriz de raíces se ponen a valores no válidos.

7.5.2 Estado inicial del diccionario del decodificador

Los valores iniciales de las variables del decodificador son:

– la "siguiente palabra de código" $C_1 = N_5$;

– el "tamaño de palabra de código vigente" $C_2 = 6$;

– la "posición de historia vigente" $C_4 = 0$;

– el "tamaño de ordinal vigente" $C_5 = 7$.

7.6 Establecimiento de conexión con control de errores

Al recibir la primitiva de confirmación C-INICIALIZACIÓN de la función de compresión de datos, la función de control indicará al DTE que puede comenzar la transferencia de datos.

7.7 Transferencia de datos entre la interfaz DTE/DCE y la función de compresión de datos

Al completarse el establecimiento de la conexión, la función de control pedirá la codificación de entrada de datos a la interfaz DTE/DCE. Para codificar datos, la función de control emitirá la primitiva petición C-DATOS a la función de compresión de datos. La primitiva indicará los datos que han de ser codificados.

Al recibir la primitiva indicación C-DATOS de la función de compresión, la función de control entregará los datos decodificados a la interfaz DTE/DCE.

Se necesitarán procedimientos de control de flujo para evitar la posible pérdida de datos debido a desbordamiento de la memoria tampón. Cuando los procedimientos definidos en la presente Recomendación se usen junto con los definidos en UIT-T V.42, se aplicarán los procedimientos de control de flujo definidos en 7.3.1/V.42 y 8.4.2/V.42.

7.8 Codificación

Los caracteres recibidos de la función de control serán codificados utilizando los procedimientos descritos en 6.3.

7.9 Transferencia de datos entre la función de compresión de datos y la función de control de errores

Al recibir la primitiva de indicación C-TRANSFERENCIA de la función de compresión de datos, la función de control emitirá la primitiva de petición L-DATOS a la función de control de errores.

Al recibir la primitiva de indicación L-DATOS de la función de control de errores, la función de control emitirá la primitiva de petición C-TRANSFERENCIA a la función de compresión de datos.

La función de compresión de datos utilizará los procedimientos descritos en 6.6.

7.10 Decodificación

Los datos codificados recibidos de la función de control serán decodificados utilizados los procedimientos indicados en 6.4.

7.11 Ajustes autónomos

Durante el funcionamiento, algunas variables son cambiadas autónomamente por la función de compresión de datos. El modo de funcionamiento puede cambiar también autónomamente del modo con compresión al modo transparente.

7.11.1 Tamaño de ordinal y STEPUP

En la reinicialización del diccionario, el codificador y su decodificador par inicializarán la transferencia de un ordinal de 7 bits ($C_5 = 7$). Cuando el codificador está a punto de transferir el primer ordinal con valor numérico superior a 127_{10}, pondrá C_5 a 8. Si está en el modo con compresión, transferirá también el código de control STEPUP inmediatamente antes de transferir el prefijo y el ordinal.

7.11.2 Tamaño de palabra de código y STEPUP

Las palabras de código y los códigos de control son transferidos utilizando el número de bits definido por C_2. En la reinicialización del diccionario, C_2 se pone al valor de 6 y C_3 al de 64. Cuando el decodificador está a punto de transferir una palabra de código con valor numérico superior o igual a C_3:

a) el codificador transferirá el código de control STEPUP utilizando el tamaño de palabra de código vigente, C_2;

b) el tamaño de palabra de código C_2 se aumentará en uno;

c) C_3 será doblado;

d) si la palabra de código que ha de ser transferida es aún numéricamente mayor o igual a C_3, se repetirán los pasos a) a c).

Después, el prefijo y la palabra de código serán transferidos.

7.11.3 Árbol de nodos lleno

Cuando el codificador no puede crear una nueva palabra de código porque el árbol de nodos está lleno ($C_1 = N_{2T}$), reinicializará el diccionario, como se describe en 7.12.

7.11.4 Historia llena

Cuando la historia está llena ($C_4 = N_{8T}$), el codificador terminará cualquier actividad de concordancia de cadena o de extensión de cadenas en curso, transferirá el código o códigos resultantes y reinicializará el diccionario, como se describe en 7.12. El siguiente carácter recibido se colocará en la primera posición de la historia.

7.11.5 Supervisión de compresibilidad de datos

El codificador aplicará periódicamente una prueba para determinar la compresibilidad de los datos. Aunque en la presente Recomendación no se especifica la naturaleza de la prueba, consistirá en la comparación del número de bits requeridos para representar un segmento del tren de datos antes y después de la compresión.

La supervisión del tren de datos continúa en los modos de funcionamiento con compresión y transparente.

En el modo con compresión, si el codificador determina que el tren de datos no es comprimible, pasará al modo transparente, como se describe en 6.5.1.

En el modo transparente, si el codificador determina que el tren de datos es comprimible, pasará al modo con compresión, como se describe en 6.5.2.

7.12 Reinicialización del diccionario

El diccionario del codificador es reinicializado fijándolo al estado definido en 7.5.1. Si la función de compresión de datos está en el modo con compresión, el codificador transferirá también el código de control REINIT, y el decodificador, al recibir REINIT, fija el diccionario del decodificador al estado definido en 7.5.2.

7.13 Transferencia de datos acelerados y EVACUACIÓN

En determinadas circunstancias, puede ser necesario terminar la concordancia de cadena o la extensión de cadena y transferir inmediatamente cualesquiera datos parcialmente codificados. La especificación de estas condiciones está fuera del ámbito de la presente Recomendación, pero un ejemplo sería si la función de control de errores estuviese en la condición de reposo. La función de control emitirá una primitiva de petición C-EVACUACIÓN a la función de compresión de datos, y transferirá los datos restantes de acuerdo con 7.9.

Si la función de compresión de datos está en el modo comprimido, al recibir una petición C-EVACUACIÓN de la función de control, el codificador:

a) terminará los procedimientos de concordancia de cadena o de extensión de cadena, según se describe en 6.3.1 y 6.3.2;

b) transferirá a los códigos restantes de acuerdo con 6.6;

c) actualizará el diccionario del codificador según proceda;

d) transferirá el código de control FLUSH;

e) si es necesario, transferirá suficientes bits 0 para establecer la alineación en octetos.

El diccionario del codificador no es reinicializado. Es actualizado, según proceda, anexando el carácter que sigue a la evacuación.

Al recibir un código de control FLUSH, el decodificador establecerá la alineación en octetos. El diccionario del decodificador no es reinicializado, y la recepción del código de control FLUSH no afecta la creación de nuevas cadenas: el decodificador procesará el código recibido después de FLUSH en continuidad con el código recibido antes de FLUSH, de acuerdo con el proceso de creación de cadenas del decodificador definido en 6.4.2.

Si la función de compresión de datos está en modo transparente, al recibir una petición C-EVACUACIÓN de la función de control, el codificador transferirá todo los datos recibidos hasta ese punto.

7.14 Secuencia de instrucción ESCAPE

Una secuencia de instrucción de modo transparente consistirá en el ESCAPE, seguido por una de las instrucciones enumeradas en el cuadro 9 anterior, salvo ESCAPE.

El valor de ESCAPE es variable. En la inicialización de la función de compresión de datos, se le asigna el valor 0. Durante el modo transparente, si se detecta el valor vigente de ESCAPE dentro del tren de datos de DTE, el ESCAPE detectado será transferido y seguido inmediatamente por la instrucción EID. El valor vigente de ESCAPE se cambia añadiéndole 51_{10}, la adición ejecutada en

módulo 256. Obsérvese que mientras está en el modo con compresión o en el modo parámetro, el valor vigente de ESCAPE NO es modificado si se detecta dentro del tren de datos del DTE.

7.15 Acción al detectar C-ERROR

La indicación C-ERROR se utiliza para informar a la función de control que una excepción (por ejemplo, un error de procedimiento o pérdida de sincronización) ha sido detectada por la función de compresión de datos. La función de control ejecutará la acción de restablecimiento apropiada, incluido el restablecimiento de la conexión con corrección de errores.

Las siguientes condiciones reconocidas por el decodificador resultarán en la generación de la primitiva de indicación C-ERROR:

– la recepción de un código de control STEPUP que haría que el valor de C_2 excediese de N_1;

– la recepción de un código de control STEPUP que haría que el valor de C_5 excediese de 8; o

– la recepción de una palabra de código mayor que C_1.

8 Parámetros

El cuadro 10 define los parámetros negociados entre funciones de compresión de datos pares. Los parámetros "P" son los valores propuestos por cada lado de la negociación, y los parámetros "N" son los valores acordados finalmente. El cuadro 11 define todos los parámetros del funcionamiento de compresión de datos. (MSB indica el byte más significativo, LSB indica el byte menos significativo.)

Cuadro 10/V.44 – Negociación de parámetros para la función de compresión de datos

Parámetro	Identificador 8……1	Longitud (octetos)	Valor 8……1	Significado	Gama	Por defecto	Valor acordado
	00000000 a 01000000			No se utiliza			
C_0	01000001	00000001	PM00000N	Capacidad V.44: Valor de N: 0 negociación de parámetros mediante protocolo (es decir, XID) 1 negociación de parámetros después de establecimiento del enlace Los bits P y M no son utilizados por conexiones de módems: véase el anexo B.			
P_0	01000010	00000001	000000RT	Petición de compresión de datos: Valor de RT: 00 ninguno de los dos sentidos 01 sólo en el sentido transmisión 10 sólo en el sentido recepción 11 ambos sentidos	00 – 11	11	
P_{1T}	01000011	00000010	(MSB) (LSB)	Número propuesto de palabras de código (incluidos códigos de control) en el sentido transmisión	256 – 65535	1024	N_{2T}

Cuadro 10/V.44 – Negociación de parámetros para la función de compresión de datos *(fin)*

Parámetro	Identificador 8……1	Longitud (octetos)	Valor 8……1	Significado	Gama	Por defecto	Valor acordado
P_{1R}	01000100	00000010	(MSB) (LSB)	Número propuesto de palabras de código (incluidos códigos de control) en el sentido recepción	256 – 65535	1024	N_{2R}
P_{2T}	01000101	00000001		Longitud de cadena máxima propuesta para el sentido transmisión	32 – 255	255	N_{7T}
P_{2R}	01000110	00000001		Longitud de cadena máxima propuesta para el sentido recepción	32 – 255	255	N_{7R}
P_{3T}	01000111	00000010	(MSB) (LSB)	Longitud de historia propuesta para el sentido transmisión	≥ 512	3 * P_{1T}	N_{8T}
P_{3R}	01001000	00000010	(MSB) (LSB)	Longitud de historia propuesta para el sentido recepción	≥ 512	3 * P_{1R}	N_{8R}
	01001001 a 11111110			Reservado para uso futuro			
End	11111111			Fin de parámetros: Salir del modo parámetros			

Cuadro 11/V.44 – Parámetros operativos para la función de compresión de datos

Parámetro	Significado	Valor
N_{1T}, N_{1R}	Tamaño máximo de palabra de código (en bits)	Derivado de N_{2T}, N_{2R}
N_{2T}, N_{2R}	Número máximo de palabras de código (incluidos códigos de control)	Véase el cuadro 10
N_3	Tamaño de carácter recibido, en bits	8
N_4	Número de caracteres en alfabeto	256
N_5	Número de códigos de control & primera palabra de código disponible	4
N_6	Reservado	
N_{7T}, N_{7R}	Longitud de cadena máxima	Véase el cuadro 10
N_{8T}, N_{8R}	Tamaño de historia	Véase el cuadro 10

El cuadro 12 define las variables operativas utilizadas en la función de compresión de datos. El codificador y el decodificador deben mantener un conjunto separado de estas variables.

Cuadro 12/V.44 – Variables operativas para la función de compresión de datos

Parámetro	Significado
C_1	Valor de palabra de código de siguiente entrada disponible de árbol de nodos (en el codificador) o conjunto de cadenas (en el decodificador)
C_2	Tamaño de palabra de código vigente, en bits
C_3	Umbral para cambiar el tamaño de palabra de código; igual a 2 a la potencia C_2. Sólo utilizado por el codificador
C_4	Posición vigente en la historia
C_5	Tamaño de ordinal, en bits

ANEXO A

Campo de información XID para negociar la capacidad V.44 cuando se utiliza con el algoritmo V.42

Para la negociación de XID, véanse 7.3 y 7.4.1.

Cuadro A.1/V.44 – Subcampo de datos de usuario XID para negociar parámetros de compresión de datos V.44

MSB: byte más significativo LSB: byte menos significativo	Bit 8.....1	
Identificador de grupo	1 1 1 1 1 1 1 1	Subcampo de datos de usuario
Identificador de parámetro	0 1 0 0 0 0 0 0	UIT-T V.44 – Identificador de conjunto de parámetros
Longitud de parámetro	0 0 0 0 0 0 1 1	Longitud de cadena, en octetos: 3
Valor de parámetro	0 1 0 1 0 1 1 0 0 0 1 1 0 1 0 0 0 0 1 1 0 1 0 0	V 4 4
Identificador de parámetro	0 1 0 0 0 0 0 1	Capacidad UIT-T V.44 (C_0)
Longitud de parámetro	0 0 0 0 0 0 0 1	Longitud de campo, en octetos: 1

Cuadro A.1/V.44 – Subcampo de datos de usuario XID para negociar parámetros de compresión de datos V.44 *(continuación)*

Valor de parámetro	PM 0 0 0 0 0 N	Valor para PM: 00 no se soporta el método paquetes ni el método multipaquetes: para conexiones de módem solamente. 01 no válido 10 se soporta el método paquetes. 11 se soportan los métodos paquetes y multipaquetes. El bit N no se utiliza en redes de paquetes. Valor para N: 0 Negociación de parámetros mediante el intercambio de XID y parámetros por debajo. 3 Negociación de parámetros después del establecimiento del enlace. Los bits P y M no se tienen en cuenta para conexiones de módems.
Identificador de parámetro	0 1 0 0 0 0 1 0	UIT-T V.44 – Petición de compresión de datos (P_0)
Longitud de parámetro	0 0 0 0 0 0 0 1	Longitud de campo, en octetos: 1
Valor de parámetro	0 0 0 0 0 0 RT	Petición de sentido de compresión señalizada por valor de RT: 00 ninguno de los dos sentidos 01 sólo en el sentido transmisión 10 sólo en el sentido recepción 11 ambos sentidos
Identificador de parámetro	0 1 0 0 0 0 1 1	UIT-T V.44 – Número de palabras de código para el sentido transmisión (P_{1T})
Longitud de parámetro	0 0 0 0 0 0 1 0	Longitud de campo, en octetos: 2
Valor de parámetro	NNNNNNNN (MSB) NNNNNNNN (LSB)	Valor de parámetro P_{1T}
Identificador de parámetro	0 1 0 0 0 1 0 0	UIT-T V.44 – Número de palabras de código para el sentido recepción (P_{1R})
Longitud de parámetro	0 0 0 0 0 0 1 0	Longitud de campo, en octetos: 2
Valor de parámetro	NNNNNNNN (MSB) NNNNNNNN (LSB)	Valor de parámetro P_{1R}

Cuadro A.1/V.44 – Subcampo de datos de usuario XID para negociar parámetros de compresión de datos V.44 *(fin)*

Identificador de parámetro	0 1 0 0 0 1 0 1	UIT-T V.44 – Longitud máxima de cadena para el sentido transmisión (P_{2T})
Longitud de parámetro	0 0 0 0 0 0 0 1	Longitud de campo, en octetos: 1
Valor de parámetro	NNNNNNNN	Valor de parámetro P_{2T}
Identificador de parámetro	0 1 0 0 0 1 1 0	UIT-T V.44 – Longitud máxima de cadena para el sentido recepción (P_{2R})
Longitud de parámetro	0 0 0 0 0 0 0 1	Longitud de campo, en octetos: 1
Valor de parámetro	NNNNNNNN	Valor de parámetro P_{2R}
Identificador de parámetro	0 1 0 0 0 1 1 1	UIT-T V.44 – Longitud de historia para el sentido transmisión (P_{3T})
Longitud de parámetro	0 0 0 0 0 0 1 0	Longitud de campo, en octetos: 2
Valor de parámetro	NNNNNNNN (MSB) NNNNNNNN (LSB)	Valor de parámetro P_{3T}
Identificador de parámetro	0 1 0 0 1 0 0 0	UIT-T V.44 – Longitud de historia para el sentido recepción(P_{3R})
Longitud de parámetro	0 0 0 0 0 0 1 0	Longitud de campo, en octetos: 2
Valor de parámetro	NNNNNNNN (MSB) NNNNNNNN (LSB)	Valor de parámetro P_{3R}

ANEXO B

Funcionamiento de la capacidad V.44 en redes de paquetes

El funcionamiento de la capacidad V.44 en redes de paquetes difiere de su funcionamiento en módems, principalmente porque el terminal de red tiene conocimiento de los límites de los paquetes, incluida la delineación entre encabezamiento y datos. El terminal de red puede identificar un paquete completo de la capa de transporte y procesarlo como una unidad, mientras que un módem procesa un tren continuo de datos no tramados.

Este anexo describe dos métodos para comprimir datos en paquetes: el método paquetes, en el cual cada paquete es procesado separadamente, y el método multipaquetes, en el cual varios paquetes, o porciones de paquetes, son procesados como una continuación. El método multipaquetes requiere la entrega garantizada de todos los paquetes. El método primario de funcionamiento de la capacidad V.44, descrita en el texto principal de la presente Recomendación, se denomina el "método trenes".

En este anexo, el término "paquete" se utiliza para indicar un paquete de la capa de transporte, paquete IP, trama, bloque, N-PDU, etc. El término "red de paquetes" se utiliza para describir cualquier red que trata paquetes; como ejemplos cabe citar IP, retransmisión de tramas, etc. El término "segmento" se utiliza para describir una porción de un paquete que ha sido segmentado para poder transmitirlo por un enlace.

Entre las características importantes de la compresión de datos en redes de paquetes cabe citar:

– Los paquetes son comprimidos y transmitidos como una unidad. De este modo, se efectúa una evacuación, utilizando el código de control FLUSH, después que el último byte de datos del paquete ha pasado al codificador para completar la compresión y transmitir el paquete entero.

– La negociación de parámetros de compresión de datos no es necesaria, pues el conjunto de parámetros V.44 por defecto para redes de paquetes se define en la subcláusula B.1. Si es necesario, se puede efectuar la negociación de compresión de datos y parámetros conexos fuera de la capacidad V.44, antes de la transferencia de datos. Esta negociación puede utilizar XID o cualquier otro intercambio de mensaje de protocolo.

– Si la capacidad V.44 es negociada entre funciones pares de compresión de datos (es decir, utilizando XID), los bits P y M del parámetro capacidad V.44 se utilizan para indicar que se soporta el método paquetes V.44 y el método multipaquetes V.44. Una función par que soporte el método multipaquetes V.44 deberá soportar también el método paquetes. Si soporta ambos métodos pero la otra función par sólo soporta el método paquetes, se elige el método paquetes. Véase el cuadro A.1.

– El modo transparente no es soportado en los métodos paquetes y multipaquetes. Se utiliza un indicador para señalizar al otro extremo si el paquete está comprimido o no. Por consiguiente, se transmite el paquete original más pequeño y el resultado de la operación de compresión de datos. El indicador es un bit (u otra indicación) en el paquete o encabezamiento de protocolo, o el código de control ETM (precedido por su prefijo de código "1") en el primer byte del paquete, seguido por los datos originales, alineados en octetos. (En este contexto, el código ETM es simplemente un indicador, y no supone el comportamiento indicado en 6.5.)

NOTA – En redes de paquetes, se puede mejorar el tiempo de ejecución si, en el momento de aumentar el tamaño de la palabra de código (por ejemplo, de 8 a 9 bits), se verifica si la compresión de los datos ha sido satisfactoria hasta ese momento. Si no ha sido así, la codificación puede ser abortada y el paquete original transmitido.

B.1 Funcionamiento del método paquetes V.44

El método paquetes es el método V.44 por defecto en redes de paquetes. Cuando se utiliza este método, no es necesario negociar las funciones pares de compresión de datos.

B.1.1 Descripción general

El método paquetes se debe utilizar en redes en las cuales se transmiten paquetes o segmentos utilizando el modo con acuse de recibo, sin entrega garantizada. En este caso, como es posible que un paquete anterior no haya sido recibido, el método paquetes no se basa en la información de paquetes previos para la compresión y descompresión de cualquier paquete. El método paquetes puede ser preferido también incluso si se garantiza la entrega, debido a su simplicidad en comparación con el método multipaquetes. El método paquetes permite que un computador central de red o conmutador de paquetes que sustenta numerosos enlaces utilice un solo caso de diccionarios de codificador y decodificador V.44 y el contexto para sustentar todos los enlaces, puesto que cada paquete es procesado separadamente.

Si la capacidad V.44 es negociada entre funciones pares de compresión de datos, el apoyo del método paquetes se indica en el parámetro capacidad V.44 fijando el bit P a "1" y el bit M a "0".

Las diferencias específicas entre el método paquetes V.44 y el método trenes V.44 son:

– Los paquetes son comprimidos y transmitidos como una unidad, el codificador transfiere un código de control FLUSH al final del paquete.

– Entre paquetes, los diccionarios del codificador y del decodificador son reinicializados. La reinicialización es ejecutada por la función de control o está incorporada en el propio algoritmo. El codificador no transfiere un código de control REINIT.
– La historia del codificador y del decodificador no es asignada. Los datos no comprimidos en el paquete que se está codificando sirven como la historia del codificador, y los datos descomprimidos colocados en una memoria tampón de salida por el decodificador sirven como la historia del decodificador.
– Cuando se ha creado la última palabra de código (el árbol de nodos está lleno), las actividades de concordancia de cadena y extensión de cadena continúan normalmente, pero no se crean palabras de código adicionales. De este modo, las longitudes de extensión de cadena son transferidas, pero no son utilizadas para definir segmentos de cadena adicionales en el árbol de nodos o nuevas cadenas en el conjunto de cadenas.
– Los valores por defecto de los parámetros de compresión de datos se enumeran en B.1.2. Es posible fijar otros valores mediante negociación.

B.1.2 Valores por defecto de parámetros de compresión de datos para el método paquetes

Si no ha habido negociación de parámetros entre los dispositivos dentro de una red de paquetes, la capacidad V.44 funciona en el método paquetes con los valores de parámetros por defecto indicados a continuación:

– P_0 – 11, ambos sentidos;
– P_{1T} – 1525 palabras de código;
– P_{1R} – 1525 palabras de código;
– P_{2T} – longitud máxima de cadena 255;
– P_{2R} – longitud máxima de cadena 255;
– P_{3T} – no es aplicable;
– P_{3R} – no es aplicable.

B.2 Funcionamiento del método multipaquetes V.44

A expensas de complejidad adicional y memoria, el método multipaquetes puede proporcionar mejor compresión que el método paquetes. Sólo se puede utilizar en una red de paquetes en la cual los paquetes o segmentos son transmitidos utilizando el modo con acuse de recibo con entrega garantizada. El funcionamiento del método multipaquetes impone una carga de complejidad y memoria al computador central de la red o al conmutador de paquetes que soporta numerosos enlaces punto a punto con terminales u otros dispositivos de usuario, pues cada enlace punto a punto que utiliza el método multipaquetes requiere diccionarios de codificador y decodificador y el contexto.

El método multipaquetes requiere también un superconjunto de los procedimientos y estructuras utilizados en el método paquetes. Los dispositivos que soporten el método multipaquetes pueden soportar también el funcionamiento con el método paquetes.

B.2.1 Descripción general

Si la capacidad V.44 es negociada entre funciones pares de compresión de datos, el soporte del método multipaquetes se indica en el parámetro capacidad V.44. Si uno de los pares indica método multipaquetes y el otro indica método paquetes, se utiliza el método paquetes.

Las diferencias específicas entre el método multipaquetes y el método trenes V.44 son pequeñas, a saber:

– Los paquetes son comprimidos y transmitidos como una unidad; el codificador transfiere un código de control FLUSH al final del paquete.

– El tamaño de historia debe ser mayor que la longitud de paquete máxima. La longitud de historia es 3072 por defecto, a menos que se haya negociado o prefijado otra cosa. Si los tamaños de diccionario son modificados con respecto a los valores por defecto mediante la negociación, el tamaño de historia debe ser modificado en consecuencia.

– El codificador y el decodificador reinicializarán el diccionario después de procesar un paquete en el cual la compresión fue insatisfactoria (es decir, un paquete en el cual la entrada de datos originales fue más pequeña que el resultado de la operación de compresión), por lo que el paquete original ha sido transmitido.

– Antes de o durante la codificación, los datos que han de ser comprimidos son copiados del paquete en las siguientes posiciones disponibles de la historia del codificador, hasta el límite del tamaño máximo de la historia. Si la historia se llena en la mitad de un paquete, después de haber sido codificado el carácter situado en la última posición de la historia, el diccionario es reinicializado, se transfiere un código de control REINIT y el resto de los datos del paquete son copiados en la historia y comprimidos.

– Cuando se ha creado la última palabra de código (el árbol de nodos está lleno), el diccionario del codificador es reinicializado, se transfiere un código de control REINIT (incluso en la mitad de un paquete) y los caracteres no procesados de la historia se trasladan a las primeras posiciones de la misma.

B.2.2 Valores por defecto de parámetros de compresión de datos para el método multipaquetes

En el método multipaquetes, los valores por defecto son iguales que los utilizados en el método trenes. Véase el cuadro 10.

Se señala que, para que el método multipaquetes se desarrolle correctamente, el número de palabras de código utilizables debe ser superior al paquete de longitud máxima esperada. Así pues, en el caso de una red cuyo paquete de longitud máxima es de 1518 bytes, el número de palabras de código debe ser de al menos 1522, para tener en cuenta las cuatro palabras de código reservadas, y debería ser bastante mayor para que la compresión fuera lo mejor posible.

APÉNDICE I

Notas relativas a la implementación

Las siguientes notas proporcionan información sobre la implementación del algoritmo de compresión de datos y la selección de parámetros. Cuando se varía el número de palabras de código y la longitud de historia, el algoritmo V.44 puede ser escalonado para aprovechar la memoria disponible.

I.1 Selección de N_2: el número total de palabras de código

El tamaño del diccionario es igual a N_2 (suponiendo que se proporcionan entradas para los códigos de control reservados). En general, la selección de un valor grande para N_2 proporciona mejor compresión a expensas de más memoria para los diccionarios. La selección de un gran valor para N_2 significa que el número de cadenas disponibles aumenta, pero también que el valor de N_1 aumenta correspondientemente. En algunos casos, la ganancia de calidad de funcionamiento obtenida con la selección de un diccionario mayor puede ser contrapesada por el mayor tamaño de palabra de código requerido. De hecho, para determinados tipos de datos, se puede mejorar el funcionamiento

utilizando un diccionario más pequeño. En general el valor 2048 para N_2 proporciona una adecuada compresión a través de una amplia gama de tipos de datos.

Para acomodar plataformas de soporte físico con memoria limitada, N_2 puede ser diferente para los dos sentidos. Esto podría permitir, por ejemplo, un mayor número de palabras de código en el sentido servidor a cliente que en el sentido cliente a servidor, para mejorar la compresión en el sentido telecarga de Web a la vez que se minimiza la memoria total requerida para el codificador y el decodificador.

El valor de N_2 no tiene que ser una potencia de 2. Como el diccionario es reinicializado tan pronto como se crea la entrada de árbol de nodos correspondiente a palabra de código ($N_2 - 1$), en la capacidad V.44 no hay penalidad permanente para un valor mayor de N_1 con el fin de obtener palabras de código adicionales. Ésta es una ventaja con respecto al algoritmo V.42 *bis*, que casi nunca reinicializa el diccionario, y mantiene así N_1 en su valor máximo, una vez que ha sido alcanzado.

I.2 Selección de N_7: longitud máxima de cadena

Para el algoritmo de compresión de datos descrito en esta Recomendación, la longitud de cadena mayor proporciona una compresión mucho mejor, a diferencia del algoritmo V.42 *bis*. Por consiguiente, se sugiere utilizar el valor máximo 255.

I.3 Selección de N_8: estructuras de datos y longitud de historia

La historia contiene todos los caracteres recibidos en el codificador, o decodificados por el decodificador, desde la inicialización más reciente del diccionario. Para optimizar el funcionamiento, el diccionario debe ser reinicializado con la menor frecuencia posible. El diccionario es reinicializado siempre que la historia o el árbol de nodos están llenos; sin embargo, para datos que son más comprimibles, el funcionamiento es generalmente mejor si el árbol de nodos se llena antes que la historia. Por ejemplo, para una relación de compresión de ~ 10:1, un árbol de nodos de 2044 palabras de códigos (correspondiente a un valor de 2048 para N_{2T}) funciona mejor con una historia de 15 000 caracteres aproximadamente (aunque se puede obtener una buena calidad de funcionamiento con 4096).

Con miras a un funcionamiento satisfactorio de la capacidad V.44, se propone que la longitud de la historia (N_8) sea siempre superior o igual a $2 \times N_2$. Sin embargo, si se pretende acomodar plataformas de soporte físico con memoria limitada, N_8 puede diferir para los dos sentidos. Esto permitiría, por ejemplo, una historia mayor en el sentido servidor a cliente que en el sentido cliente a servidor, con el fin de mejorar la compresión en el sentido hacia el destino (telecarga de página Web) dentro de un requisito de memoria total fija para el codificador y el decodificador. Por ejemplo, en vez de utilizar el valor por defecto de $3 \times N_2$ para ambos sentidos, se podría utilizar $4 \times N_2$ para el sentido servidor a cliente y sólo $2 \times N_2$ en el sentido cliente a servidor.

Otro método para maximizar la utilización de los recursos de memoria del codificador es agregar la memoria asignada para la historia y el árbol de nodos, y permitir que la historia aumente sobre su límite nominal en una memoria no utilizada asignada aparte para las palabras de código del árbol de nodos. Si los datos son más comprimibles que de costumbre, y el árbol de nodos aumenta más lentamente que de costumbre, el codificador tendrá una ejecución más larga antes de la reinicialización, porque la historia puede utilizar la memoria restante planificada para el árbol de nodos.

El codificador no debe utilizar más historia que la que el decodificador ha negociado, con el fin de evitar desbordamiento. Sin embargo, en algunas aplicaciones útiles (por ejemplo, decodificador en el cliente, codificador en el servidor) los recursos de memoria del decodificador pueden ser mucho mayores que los del codificador. Si el codificador negocia un tamaño para la historia del

decodificador mayor que el asignado para su propia historia, tiene un margen para aumentar su propia asignación, siempre que se reinicialice antes que la historia del decodificador esté llena.

Por ejemplo, se supone que el servidor tiene 6000 bytes presupuestados para la historia del codificador (6000 caracteres) y 14 308 bytes presupuestados para el árbol de nodos del codificador (2044 palabras de código con 7 bytes por entrada de palabra de código: cada entrada de palabra de código utiliza un byte para longitud de segmento de cadena y dos bytes para índice de historia, índice descendente e índica lateral); la memoria agregada para el codificador es 20 308 bytes. Sin embargo, en negociación con el cliente, el codificador del servidor propone para la historia un valor mayor que 6000, tal como 15 000. El cliente puede aceptarlo o convenir un número más pequeño; no obstante, el valor acordado, indicado como N_{8T} por el codificador, probablemente será mayor que los 6000 presupuestados por el codificador para la historia. Se supone que N_8 es 15 000.

En el codificador del servidor, la historia y el árbol de nodos compartirán un conjunto de memoria común de modo que:

– la primera posición de historia está en el *primer* byte del conjunto de memoria;

– el índice de historia vigente se incrementa para cada nuevo carácter colocado en la historia;

– la estructura de cada entrada del árbol de nodos (todos los campos de datos para una palabra de código dada) están en 7 bytes consecutivos de memoria;

– la estructura para la primera palabra de código (4) está colocada en los *últimos* 7 bytes del conjunto de memoria;

– la estructura para la última palabra de código (2047) está colocada en los 7 bytes inmediatamente *después* del último byte de la historia; y

– el siguiente puntero de estructura de palabra de código disponible es disminuido para cada palabra de código creada.

En el funcionamiento, la historia se llena desde el primer byte del conjunto e incrementa. Las entradas de árbol de nodos se crean a partir de los últimos 7 bytes del conjunto y disminuyen. El codificador reinicializa el diccionario cuando se produce uno de los siguientes eventos:

a) se crea la última palabra de código ($C_1 = N_{2T} – 1 = 2047$);

b) la posición de historia vigente alcanza N_{8T} ($C_4 = N_{8T} = 15\ 000$); o

c) la posición de historia vigente está dentro de N_{7T} de la siguiente estructura de palabra de código disponible.

La condición a) es normal, e indica que el árbol de nodos del codificador está lleno. La condición b) impide que la historia del *decodificador* desborde. La condición c) asegura que la historia del codificador no sobreescribe entradas de palabras de código, mientras se extiende en la memoria asignada aparte para las estructuras de palabra de código.

El resultado es que si los datos son comprimidos en la gama de 10:1, el árbol de nodos crece lentamente en comparación con la historia, y la historia puede extenderse en la memoria originalmente asignada para el árbol de nodos, posiblemente aumentando la historia utilizable hasta 9000 (= 15 000 – 6000) bytes.

I.4 Compresión eficaz de datos Unicode

Unicode es un sistema internacional de codificación de caracteres normalizado de 16 bits diseñado para soportar el intercambio, procesamiento y visualización de muchos modernos idiomas escritos. Consta de representaciones numéricas para casi todos los caracteres de los principales idiomas escritos de Europa, América, Oriente Medio, India, África, Asia y Pacífico. Es una aplicación de ISO/CEI 10646, restringida a 2 octetos, y se conoce también como UCS-2.

El algoritmo de compresión de datos definido en UIT-T V.44 no trata específicamente caracteres de 16 bits, pero comprimirá eficazmente los datos Unicode como caracteres de 8 bits. Para las aplicaciones en las cuales las capas de protocolo por encima de V.42 saben que UCS-2 es una porción importante del tráfico, se recomienda utilizar el esquema de compresión especificado en el Informe Técnico Unicode #6 en los datos antes de pasarlos al algoritmo V.44, pues esto puede mejorar la calidad de funcionamiento global.

I.5 Aplicabilidad del modo transparente

El modo transparente se requiere para funcionamiento por una conexión de módem. En redes de paquetes, no hace falta el modo transparente porque el encabezamiento de la trama está marcado. En los demás casos, habría algún motivo para utilizarlo, como ha sido cedido.

I.6 Cálculo de la característica de compresión

El cálculo de la característica de compresión se puede expresar como la relación entre el número de caracteres recibido por el codificador y el número de octetos transferidos a la función de control. El cómputo de caracteres y octetos se debe fijar a 0 en la inicialización de la función de compresión de datos, y también entre pruebas.

I.7 Diferencias entre los algoritmos V.44 y V.42 *bis*

A continuación figura una lista de las principales diferencias entre los algoritmos de compresión descritos en la presente Recomendación y en UIT-T V.42 *bis*:

a) Para la mayoría de los tipos de ficheros encontrados al hojear Internet, logra relaciones de compresión superiores.

b) El codificador y el decodificador utilizan una memoria intermedia de historia.

c) La estructura del diccionario del decodificador es diferente de la estructura del diccionario del codificador.

d) Los tamaños de diccionario, etc., para los sentidos de transmisión y recepción son negociados independientemente.

e) Los nodos en el diccionario del codificador pueden definir segmentos de cadena, en vez de caracteres individuales.

f) Los nodos del diccionario utilizan índices de la posición en la historia de un carácter, en vez de los propios caracteres.

g) Reinicializa el diccionario cuando está lleno, en vez de recuperar palabras de código.

h) No reserva las primeras 256 palabras de código para todos los caracteres de 8 bits posibles.

i) Utiliza un procedimiento de extensión de cadena para codificar rápidamente una cadena encontrada para el segundo tiempo.

j) Transfiere códigos para caracteres y longitudes de extensión de cadena, así como para cadenas y control.

k) Puede utilizar inmediatamente todas las palabras de código creadas por el codificador, incluso la más reciente, una palabra de código que el decodificador puede no haber creado aún.

l) No modifica el valor de ESCAPE cuando es detectado en el modo con compresión.

m) No actualiza el diccionario del decodificador cuando funciona en el modo transparente.

n) Reinicializa los diccionarios en la transición del modo transparente al modo con compresión.

o) La mejor respuesta a una propuesta de parámetro P_0 es la inversión: la respuesta complementaria a una propuesta, de la entidad A, para la compresión en el *sentido*

transmisión (de la entidad A) *solamente*, es un acuerdo, de la entidad B, para la compresión en el *sentido recepción* (de la entidad B) *solamente*.

APÉNDICE II

Ilustración del funcionamiento del algoritmo V.44

En los ejemplos de este apéndice se supone una implementación del codificador que anexa caracteres y actualiza el árbol de nodos antes de que se reciba realmente el siguiente carácter. De este modo, en determinados pasos de estos ejemplos, el contenido del árbol de nodos puede diferir de los de una realización que anexa caracteres y actualiza el árbol de nodos solamente después que se ha recibido el siguiente carácter.

II.1 Compresión y descompresión de "ABCDEXABCDEYABCDE FF_H AC"

A continuación se pretende ilustrar el funcionamiento del algoritmo V.44 en el modo con compresión, mostrando el procesamiento de un ejemplo específico. El procesamiento muestra, en diferentes etapas:

i) los nuevos caracteres, como entrada al codificador;

ii) el análisis del tren de entrada por el algoritmo, para codificación;

iii) la estructura de diccionario revisada del codificador;

iv) la información transferida por el codificador, y el método por el cual es codificada;

v) el análisis de los datos comprimidos por el algoritmo, para decodificación; y

vi) la estructura de diccionario revisada del decodificador.

En este ejemplo, se supone que la estructura de diccionario acaba de ser reinicializada, por lo que no hay historia previa y C_5 se pone 7 y C_2 se pone a 6. Los caracteres son introducidos uno por uno, pero esta presentación mostrará también etapas, consolidando algunas etapas cuando el procesamiento es obvio. En la matriz de raíces, no se muestran las raíces antes de "A".

Los caracteres no comprimidos se colocan en la historia del codificador según llegan; el procesamiento por el codificador produce códigos binarios, que son transferidos; después que el decodificador ha recibido los datos comprimidos, los caracteres decodificados son colocados en la historia del decodificador.

Etapa 0: Ningún dato previo.

El estado del codificador es:

Historia del codificador:

Posición																			
Carácter																			

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	..	FF_H
Índice descendente	-	-	-	-	-	-	-	-	-	-

Árbol de nodos:

Palabra de código												
Posición del primer carácter												
Longitud de segmento												
Índice descendente												
Índice lateral												

El estado del decodificador es:

Conjunto de cadenas:

Palabra de código												
Posición del último carácter												
Longitud de cadena												

Historia del decodificador:

Posición																			
Carácter																			

Etapa 1:

Nuevos caracteres: A

Análisis del codificador: No hay índice descendente para "A" en la matriz de raíces, por lo que es transferido como un ordinal. La primera entrada de árbol de nodos disponible (palabra de código 4) se utiliza para crear un segmento de cadena: tiene la posición de primer carácter 1 (la posición siguiente a "A"), longitud de segmento 1 (es un segmento de 1 carácter) y no tiene índices descendente ni lateral válidos: esencialmente anexado al siguiente carácter, no visto, para crear una cadena de 2 caracteres. El conjunto de raíces es actualizado también con un índice descendente para "A", que indica esta palabra de código 4.

Historia del codificador:

Posición	0																		
Carácter	A																		

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	..	FF_H
Índice descendente	4	-	-	-	-	-	-	-	-	-

Árbol de nodos:

Palabra de código	4											
Posición del primer carácter	1											
Longitud de segmento	1											
Índice descendente	-											
Índice lateral	-											

Transferido: ordinal para "A": 41_H = "1000001"

Análisis del decodificador: El ordinal para "A" es ampliado al carácter de 8 bits y colocado en la historia. Al reinicializar el diccionario, el decodificador es inicializado para no crear una palabra de código cuando se recibe el primer ordinal.

Conjunto de cadenas:

Palabra de código												
Posición del último carácter												
Longitud de cadena												

Historia del decodificador:

Posición	0																		
Carácter	A																		

Etapa 2:

Nuevos caracteres: B

Análisis del codificador No hay índice descendente para "B" en el conjunto de raíces, por lo que se transfiere como un ordinal. La siguiente entrada del árbol de nodos disponible (palabra de código 5) se utiliza para crear un segmento de cadena: tiene la posición de primer carácter 2 (la posición que sigue a "B"), longitud de segmento 1 (es un segmento de 1 carácter) y no tiene índices descendente ni lateral válidos. El diccionario de raíces es actualizado también con un índice descendente para "B", que indica la palabra de código 5.

Historia del codificador:

Posición	0	1																	
Carácter	A	B																	

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	..	FF_H
Índice descendente	4	5	-	-	-	-	-	-	-	-

Árbol de nodos:

Palabra de código	4	5										
Posición del primer carácter	1	2										
Longitud de segmento	1	1										
Índice descendente	-	-										
Índice lateral	-	-										

Transferido: ordinal para "B": 42_H = "1000010"

Análisis del decodificador: El carácter "B" es colocado en la historia. El conjunto de cadenas es actualizado creando una cadena que termina con "B" (en posición 1), de longitud 2 (que representa la cadena "AB") y que utiliza la primera palabra de código disponible, 4.

Conjunto de cadenas:

Palabra de código	4											
Posición del último carácter	1											
Longitud de cadena	2											

Historia del decodificador:

Posición	0	1																	
Carácter	A	B																	

Etapas 3, 4, 5, 6:

Nuevos caracteres: C D E X

Análisis del codificador Para estos caracteres, el procesamiento es igual que para "B": cada uno de ellos es transferido como un ordinal y el árbol de nodos y la matriz de raíces son actualizados en consecuencia.

Nuevos datos: C D E X

Historia del codificador:

Posición	0	1	2	3	4	5													
Carácter	A	B	C	D	E	X													

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	..	FF_H
Índice descendente	4	5	6	7	8	-	9	-	-	-

Árbol de nodos:

Palabra de código	4	5	6	7	8	9						
Posición del primer carácter	1	2	3	4	5	6						
Longitud de segmento	1	1	1	1	1	1						
Índice descendente	-	-	-	-	-							
Índice lateral	-	-	-	-	-							

Transferido: ordinales transferidos para "C", "D", "E", "X": 43_H 44_H 45_H 58_H = "1000011" "1000100" "1000101" "1011000"

Análisis del decodificador: Los caracteres "C", "D", "E", "X" son colocados individualmente en la historia. Se crean cadenas de 2 caracteres que terminan con C, D, E y X en el conjunto de cadenas.

Conjunto de cadenas:

Palabra de código	4	5	6	7	8							
Posición del último carácter	1	2	3	4	5							
Longitud de cadena	2	2	2	2	2							

Historia del decodificador:

Posición	0	1	2	3	4	5													
Carácter	A	B	C	D	E	X													

Etapa 7:

Nuevos caracteres: A B

Análisis del codificador: Cuando se recibe "A", el codificador sigue el índice descendente desde la raíz "A" a la palabra de código 4 en el árbol de nodos, correspondiente al segmento de cadena "B", que concuerda con el siguiente carácter "B"; como la palabra de código 4 no tiene índice descendente válido, ésta es la concordancia de cadena más larga. La palabra de código 4 es transferida y el codificador inicia el procedimiento de extensión de cadena. En este momento no se crea un nuevo segmento de cadena.

Historia del codificador:

Posición	0	1	2	3	4	5	6	7											
Carácter	A	B	C	D	E	X	A	B											

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	..	FF_H
Índice descendente	4	5	6	7	8	-	9	-	-	-

Árbol de nodos:

Palabra de código	4	5	6	7	8	9						
Posición del primer carácter	1	2	3	4	5	6						
Longitud de segmento	1	1	1	1	1	1						
Índice descendente	-	-	-	-	-	-						
Índice lateral	-	-	-	-	-	-						

Transferido: palabra de código 4: "000100"

Análisis del decodificador: El decodificador accede a la palabra de código 4 en el conjunto de cadenas. Copia la cadena "AB" de la posición 0 en las siguientes posiciones disponibles en la historia. Crea también la cadena de 2 caracteres que termina con el primer carácter de la cadena representada por la palabra de código 4 (es decir, "A"), que utiliza la palabra de código 9.

Conjunto de cadenas:

Palabra de código	4	5	6	7	8	9						
Posición del último carácter	1	2	3	4	5	6						
Longitud de cadena	2	2	2	2	2	2						

Historia del decodificador:

Posición	0	1	2	3	4	5	6	7												
Carácter	A	B	C	D	E	X	A	B												

Etapa 8a:

Nuevos caracteres: C

Análisis del codificador: El procedimiento de extensión de cadena halla que el nuevo carácter "C" concuerda con el carácter "C" en la historia que sigue inmediatamente al segmento de cadena "B" de la palabra de código 4. Como la longitud total de la cadena para la palabra de código 4 y la extensión de 1 carácter es menor que la longitud máxima de cadena, el procedimiento de extensión de cadena continúa con el siguiente carácter.

Historia del codificador:

Posición	0	1	2	3	4	5	6	7	8											
Carácter	A	B	C	D	E	X	A	B	C											

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	..	FF_H
Índice descendente	4	5	6	7	8	-	9	-	-	-

Árbol de nodos:

Palabra de código	4	5	6	7	8	9						
Posición del primer carácter	1	2	3	4	5	6						
Longitud de segmento	1	1	1	1	1	1						
Índice descendente	-	-	-	-	-	-						
Índice lateral	-	-	-	-	-	-						

Transferido: No se transfiere nada en esta subetapa.

Análisis del decodificador: No se recibe nada en esta subetapa.

Conjunto de cadenas:

Palabra de código	4	5	6	7	8	9						
Posición del último carácter	1	2	3	4	5	6						
Longitud de cadena	2	2	2	2	2	2						

Historia del decodificador:

Posición	0	1	2	3	4	5	6	7											
Carácter	A	B	C	D	E	X	A	B											

Etapa 8b:

Nuevos caracteres: D E Y

Análisis del codificador: En este procedimiento de extensión de cadena, el codificador encuentra que los siguientes caracteres siguen concordando con la historia hasta la "Y". Se transfiere una longitud de extensión de cadena de 3, que representa "CDE" que sigue a "B" original de la palabra de código 4. El codificador crea un nuevo segmento de cadena, para ampliar la concordancia de cadena más larga, con longitud 3 y una posición de historia de 8, la posición del primer carácter de extensión; utiliza la siguiente palabra de código disponible, 10. Esto termina el procedimiento de extensión de cadena: el codificador NO anexa además un segmento de cadena de 1 carácter.

El carácter "Y" no concordado se convierte en la raíz de una nueva cadena posible. Sin embargo, como no tiene índice descendente válido, es transferido como un ordinal; se crea la palabra de código 11 para anexar el segmento de cadena de 1 carácter para el carácter que sigue a "Y", y el índice descendente para la entrada de raíz de "Y" es actualizado con la palabra de código 11.

Historia del codificador:

Posición	0	1	2	3	4	5	6	7	8	9	10	11								
Carácter	A	B	C	D	E	X	A	B	C	D	E	Y								

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	..	FF_H
Índice descendente	4	5	6	7	8	-	9	11	-	-

Árbol de nodos:

Palabra de código	4	5	6	7	8	9	10	11				
Posición del primer carácter	1	2	3	4	5	6	8	12				
Longitud de segmento	1	1	1	1	1	1	3	1				
Índice descendente	10	-	-	-	-	-	-	-				
Índice lateral	-	-	-	-	-	-	-	-				

Transferido: longitud de extensión de cadena 3: "0" "10"; ordinal para "Y": 59_H = "1011001"

Análisis del decodificador: El decodificador amplía la cadena de palabra de código previa (palabra de código 4) con 3: copia los 3 caracteres en la historia que sigue inmediatamente a la cadena representada por la palabra de código 4 (utilizando la posición del último carácter más 1 como punto de partida) en las siguientes posiciones disponibles en la historia. Crea entonces la cadena que termina en "E" y le asigna la palabra de código 10: su longitud es 5 (la longitud de cadena para la palabra de código 4) + (la longitud de extensión de cadena)) = (2 + 3) = 5), y la posición del último carácter se fija a 10, la posición del último carácter copiado en la historia.

El carácter Y se coloca también en la historia. NO se crea una cadena de 2 caracteres.

Conjunto de cadenas:

Palabra de código	4	5	6	7	8	9	10					
Posición del último carácter	1	2	3	4	5	6	10					
Longitud de cadena	2	2	2	2	2	2	5					

Historia del decodificador:

Posición	0	1	2	3	4	5	6	7	8	9	10	11								
Carácter	A	B	C	D	E	X	A	B	C	D	E	Y								

Etapa 9:

Nuevos caracteres: A B C D E

Análisis del codificador Cuando se recibe "A", el codificador sigue el índice descendente desde la raíz "A" hasta la palabra de código 4 en el árbol de nodos, correspondiente al segmento de cadena "B", que concuerda con el siguiente carácter, por lo que el codificador continúa el procedimiento de concordancia de cadena: sigue el puntero descendente hasta la palabra de código 10, correspondiente al segmento de cadena "CDE". Los siguientes caracteres concuerdan completamente con este segmento de cadena. Como la palabra de código 10 no tiene índice

descendente válido, ésta es la concordancia de cadena más larga. La palabra de código 10 es transferida y el codificador inicia el procedimiento de extensión de cadena.

Historia del codificador:

Posición	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16			
Carácter	A	B	C	D	E	X	A	B	C	D	E	Y	A	B	C	D	E			

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	..	FF_H
Índice descendente	4	5	6	7	8	-	9	11	-	-

Árbol de nodos:

Palabra de código	4	5	6	7	8	9	10	11				
Posición del primer carácter	1	2	3	4	5	6	8	12				
Longitud de segmento	1	1	1	1	1	1	3	1				
Índice descendente	10	-	-	-	-	-	-	-				
Índice lateral	-	-	-	-	-	-	-	-				

Transferido: palabra de código 10: "001010"

Análisis del decodificador: El decodificador accede a la palabra de código 10 en el conjunto de cadenas. Copia esta cadena de 5 caracteres en las siguientes posiciones disponibles en la historia. Anexa también "A" a "Y" y crea la cadena de 2 caracteres, que utiliza la palabra de código 11.

Conjunto de cadenas:

Palabra de código	4	5	6	7	8	9	10	11				
Posición del último carácter	1	2	3	4	5	6	10	12				
Longitud de cadena	2	2	2	2	2	2	5	2				

Historia del decodificador:

Posición	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16			
Carácter	A	B	C	D	E	X	A	B	C	D	E	Y	A	B	C	D	E			

Etapa 10:

Nuevos caracteres: FF_H

Análisis del codificador: En este procedimiento de extensión de cadena, el codificador halla que el siguiente carácter "FF_H" no concuerda con la historia, y termina el procedimiento de extensión de cadena. Se anexa un nuevo segmento de cadena de longitud 1 (el "FF_H") al final de "ABCDE" y se le asigna la palabra de código 12. El índice descendente para la palabra de código 10 se fija a la palabra de código 12.

Después "FF_H" se utiliza como la raíz para una posible nueva concordancia de cadenas; como no tiene índice descendente válido, es transferido como un ordinal. Se crea un segmento de cadena de 1 carácter, palabra de código 13, anexando el siguiente carácter a "FF_H", y el índice descendente de la entrada raíz de "FF_H" se fija a la palabra de código 13.

(Como $FF_H = 255_{10} > 127_{10}$, el ordinal requiere transferir 8 bits; C_5 se pone a 8, y el código de control STEPUP se transfiere antes del ordinal.)

Historia del codificador:

Posición	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17		
Carácter	A	B	C	D	E	X	A	B	C	D	E	Y	A	B	C	D	E	FF_H		

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	..	FF_H
Índice descendente	4	5	6	7	8	-	9	11	-	13

Árbol de nodos:

Palabra de código	4	5	6	7	8	9	10	11	12	13		
Posición del primer carácter	1	2	3	4	5	6	8	12	17	18		
Longitud de segmento	1	1	1	1	1	1	3	1	1	1		
Índice descendente	10	-	-	-	-	-	12	-	-	-		
Índice lateral	-	-	-	-	-	-	-	-	-	-		

Transferido: STEPUP ("000010"), ordinal para "FF_H": FF_H = "11111111"

Análisis del decodificador: Al recibir el código de control STEPUP, el decodificador fija C_5 a 8. El carácter "FF_H" es colocado en la historia. El decodificador asigna la palabra de código 12 a la cadena de longitud 6 que termina con "FF_H".

Conjunto de cadenas:

Palabra de código	4	5	6	7	8	9	10	11	12			
Posición del último carácter	1	2	3	4	5	6	10	12	17			
Longitud de cadena	2	2	2	2	2	2	5	2	6			

Historia del decodificador:

Posición	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17		
Carácter	A	B	C	D	E	X	A	B	C	D	E	Y	A	B	C	D	E	FF_H		

Etapa 11:

Nuevos caracteres: A C y *C-FLUSH_request*

Análisis del codificador: Cuando se recibe "A", el codificador sigue el índice descendente desde la raíz "A" a palabra de código 4 en el árbol de nodos, correspondiente al segmento de cadena "B", que NO concuerda con el siguiente carácter "C"; como no hay índice lateral para otra opción para una concordancia en "C", "A" es transferido como un ordinal. El codificador crea un segmento de cadena de 1 carácter (palabra de código 14) anexando "C" a "A"; la palabra de código 14 se fija como el índice lateral para la palabra de código 4 (el segmento de cadena "AB" anterior).

"C" se convierte en la raíz para una nueva posible concordancia de cadena: sin embargo, como el codificador recibe una instrucción FLUSH de la función de control, debe completar cualquier concordancia de cadena en curso. En este caso, transfiere "C" como un ordinal, seguido por el código de control FLUSH. Normalmente, el codificador crea un segmento de cadena de 1 carácter anexando el siguiente carácter a "C", asignado a la palabra de código 15.

Historia del codificador:

Posición	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Carácter	A	B	C	D	E	X	A	B	C	D	E	Y	A	B	C	D	E	FF_H	A	C

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	..	FF_H
Índice descendente	4	5	6	7	8	-	9	11	-	13

Árbol de nodos:

Palabra de código	4	5	6	7	8	9	10	11	12	13	14	15
Posición del primer carácter	1	2	3	4	5	6	8	12	17	18	19	20
Longitud de segmento	1	1	1	1	1	1	3	1	1	1	1	1
Índice descendente	10	-	-	-	-	-	12	-	-	-	-	
Índice lateral	14	-	-	-	-	-	-	-	-	-	-	

Transferido: ordinales para "A" "C", código de control FLUSH, relleno de 0 bits: "01000001" "01000011" "000001" *0000*

Análisis del decodificador: El carácter "A" se coloca en la historia y se crea la cadena de 2 caracteres que termina con "A" y se le asigna la palabra de código 13. El carácter "C" se coloca en la historia y se crea la cadena de 2 caracteres que termina con "C" y se le asigna palabra de código 14. La recepción del código de control FLUSH indica un fin a los datos codificados hasta ese momento. El siguiente bit significativo estará en la primera posición de bit del siguiente octeto. El decodificador no crea la palabra de código 15 hasta que se recibe el código después de FLUSH.

Conjunto de cadenas:

Palabra de código	4	5	6	7	8	9	10	11	12	13	14	
Posición del último carácter	1	2	3	4	5	6	10	12	17	18	19	
Longitud de cadena	2	2	2	2	2	2	5	2	6	2	2	

Historia del decodificador:

Posición	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Carácter	A	B	C	D	E	X	A	B	C	D	E	Y	A	B	C	D	E	FF_H	A	C

En este punto, los diccionarios del codificador y del decodificador están sincronizados, salvo para la palabra de código 15.

La figura II.1 muestra el árbol de nodos después del paso 11.

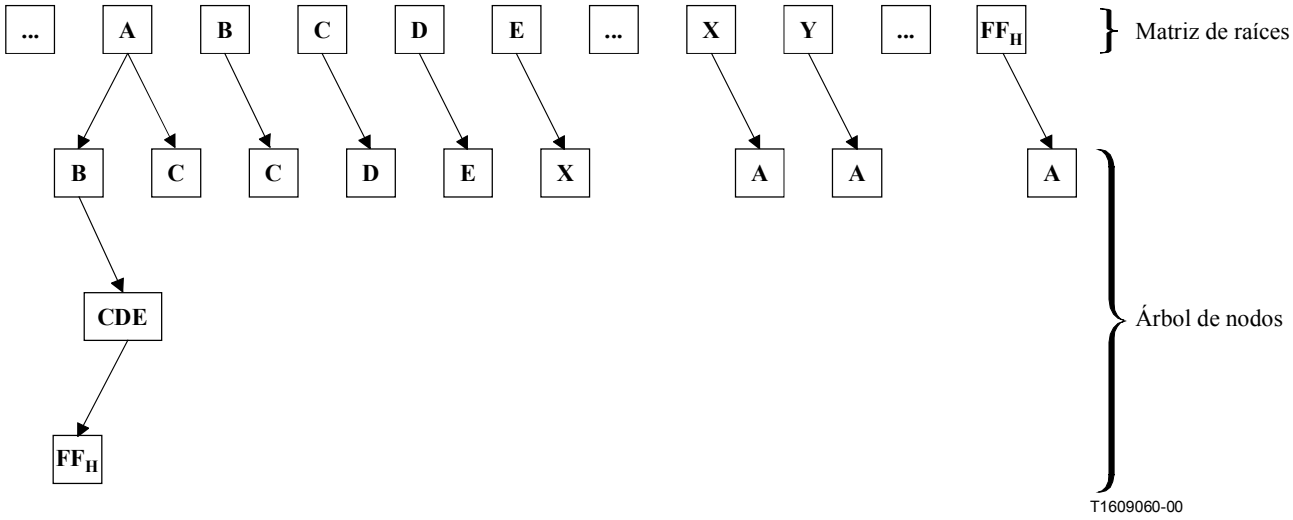

Figura II.1/V.44 – Estructuras de árbol después del paso 11

El cuadro II.1 muestra la correspondencia de octetos para el ejemplo de II.1. Para el prefijo de código y la codificación de longitud de extensión de cadena, véanse los cuadro 3 y 5.

Cuadro II.1 – Correspondencia de octetos para el ejemplo de II.1

N.º	8	7	6	5	4	3	2	1	Octeto
	1	0	0	0	0	0	1	***0***	1
	1	0	0	0	0	1	0	***0***	2
	1	0	0	0	0	1	1	***0***	3
	1	0	0	0	1	0	0	***0***	4
	1	0	0	0	1	0	1	***0***	5
	1	0	1	1	0	0	0	***0***	6
	0	0	0	0	1	0	0	***1***	7
	0	0	1	***0***	1	0	0	***1***	8
	0	1	0	***1***	1	0	1	1	9
	0	0	1	0	***1***	0	0	1	10
	1	1	1	1	1	***0***	0	0	11
	0	0	0	1	***0***	1	1	1	12
	0	1	1	***0***	0	1	0	0	13
	0	1	***1***	0	1	0	0	0	14
	0	0	0	0	0	0	0	0	15

Los bits de prefijo de código se indican en ***itálicas negritas,*** y los bits de relleno están subrayados.

II.2 Compresión y descompresión de "CCCCCCCCCCX"

La finalidad del siguiente ejemplo es ilustrar un aspecto particular del funcionamiento del algoritmo V.44, que permite comprimir con mucha rapidez secuencias de caracteres repetidos. Esto requiere que el decodificador interprete una palabra de código que no ha sido creada aún.

En el funcionamiento general del algoritmo, se están creando continuamente palabras de códigos en ambos lados, que en el codificador corresponden específicamente con segmentos de cadena y en el decodificador corresponden con la cadena completa que termina con el segmento de cadena correspondiente del codificador (en el contexto de la estructura de árbol). Como se muestra en el ejemplo de II.1, el codificador puede estar un paso más adelantado que el decodificador en la creación de palabras de código. En algunas circunstancias especiales, el codificador transferirá una palabra de código que, para el decodificador, es exactamente la siguiente palabra de código disponible (palabra de código igual a C_1).

El decodificador trata esta situación de acuerdo con las reglas indicadas en 6.4.1, en particular 3) y 4).

El procesamiento del ejemplo muestra, en diferentes etapas:

i) los nuevos caracteres, como entrada al codificador;

ii) el análisis del tren de entrada por el algoritmo, para codificación;

iii) la estructura de diccionario revisada del codificador;

iv) la información transferida por el codificador, y el método por el cual es codificada;

v) el análisis de los datos comprimidos por el algoritmo, para decodificación; y

vi) la estructura de diccionario revisada del decodificador.

En este ejemplo, se supone que la estructura de diccionario acaba de ser reinicializada, por lo que no hay historia previa. Los caracteres son introducidos uno por uno, pero esta presentación mostrará también etapas, saltando algunos pasos cuando el procedimiento es obvio.

Los caracteres no comprimidos se colocan en la historia del codificador según llegan; el procesamiento por el decodificador produce códigos binarios, que son transferidos; después que el decodificador ha recibido los datos comprimidos, los caracteres decodificados son colocados en la historia del decodificador.

Etapa 0: Ningún dato previo.

El estado del decodificador es:

Historia del codificador:

Posición																			
Carácter																			

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	Z	..
Índice descendente	-	-	-	-	-	-	-	-	-	-

Árbol de nodos:

Palabra de código												
Posición del primer carácter												
Longitud de segmento												
Índice descendente												
Índice lateral												

El estado del decodificador es:

Conjunto de cadenas:

Palabra de código												
Posición del último carácter												
Longitud de cadena												

Historia del decodificador:

Posición																			
Carácter																			

Etapa 1: Nuevos caracteres: C

Análisis del codificador: No hay índice descendente para "C" en el conjunto de raíces, por lo que es transferido como un ordinal. La primera entrada de árbol de nodos disponible (palabra de código 4)

se utiliza para crear un segmento de cadena: tiene la posición de primer carácter 1 (la posición siguiente a "C"), longitud de segmento 1 (es un segmento de cadena de 1 carácter) y no tiene índice descendente ni lateral válidos: esencialmente anexado al siguiente carácter, no visto, para crear una cadena de 2 caracteres. El conjunto de raíces es actualizado también con un índice descendente para "C", que indica esta palabra de código 4.

Historia del codificador:

Posición	0																		
Carácter	C																		

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	Z	..
Índice descendente	-	-	4	-	-	-	-	-	-	-

Árbol de nodos:

Palabra de código	4											
Posición del primer carácter	1											
Longitud de segmento	1											
Índice descendente	-											
Índice lateral	-											

Transferido: ordinal "C"

Análisis del decodificador: El carácter "C" se coloca en la historia. Al reinicializar el diccionario, el decodificador es inicializado para no crear una palabra de código cuando se recibe el primer carácter.

Conjunto de cadenas:

Palabra de código												
Posición del último carácter												
Longitud de cadena												

Historia del decodificador:

Posición	0																		
Carácter	C																		

Etapa 2: Nuevos caracteres: C C

Análisis del codificador: Cuando se recibe el primer carácter C, el codificador sigue el índice descendente desde la raíz "C" a la palabra de código 4 en el árbol de nodos, correspondiente al segmento de cadena "C" (es decir, C en posición 1), que concuerda con el segundo nuevo C; y como la palabra de código 4 no tiene índice descendente válido, ésta es la concordancia de cadena más

larga. La palabra de código 4 es transferida y el codificador inicia el procedimiento de extensión de cadena. En este momento no se crea un nuevo segmento de cadena.

Historia del codificador:

Posición	0	1	2																
Carácter	C	C	C																

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	Z	..
Índice descendente	-	-	4	-	-	-	-	-	-	-

Árbol de nodos:

Palabra de código	4												
Posición del primer carácter	1												
Longitud de segmento	1												
Índice descendente	-												
Índice lateral	-												

Transferido: palabra de código 4

Análisis del decodificador: De acuerdo con la regla 4) de 6.4.1: como la palabra de código 4 es la siguiente palabra de código que se ha de crear ($C_1 = 4$), el decodificador copia la cadena previa de 1 carácter ("C" de la posición 0) en la historia (en posición 1); y después coloca el primer carácter de la cadena copiada ("C" de la posición 1) en la historia (en posición 2).

De acuerdo con el cuadro 2: el primer carácter de la cadena de la palabra de código 4 ("CC") es anexado al carácter anterior (C en posición 0) para crear la palabra de código 4: ésta es una cadena de 2 caracteres con el último carácter en posición 1.

Conjunto de cadenas:

Palabra de código	4												
Posición del último carácter	1												
Longitud de cadena	2												

Historia del decodificador:

Posición	0	1	2																
Carácter	C	C	C																

Etapa 3: Nuevos caracteres: C

Análisis del codificador: El procedimiento de extensión de cadena halla que el nuevo carácter C concuerda con el carácter que sigue al segmento de cadena (para la palabra de código 4) en la

historia (C en posición 2). Por tanto, el procedimiento de extensión de cadena continúa (porque la longitud total de la cadena correspondiente a la palabra de código 4 más esta extensión de 1 carácter es menor que la longitud máxima de cadena).

Historia del codificador:

Posición	0	1	2	3															
Carácter	C	C	C	C															

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	Z	..
Índice descendente	-	-	4	-	-	-	-	-	-	-

Árbol de nodos:

Palabra de código	4											
Posición del primer carácter	1											
Longitud de segmento	1											
Índice descendente	-											
Índice lateral	-											

Transferido: No se transfiere nada en esta subetapa.

Análisis del decodificador: No se recibe nada en esta subetapa.

Conjunto de cadenas:

Palabra de código	4											
Posición del último carácter	1											
Longitud de cadena	2											

Historia del decodificador:

Posición	0	1	2																
Carácter	C	C	C																

Etapa 4: Nuevos caracteres: C C C C C C

Análisis del codificador: En el procedimiento de extensión de cadena, cada nuevo carácter es comparado sucesivamente con los caracteres en la historia: en este caso, como la extensión comienza con el carácter colocado en posición 3 (que fue comparado con el carácter en posición 2), el primer nuevo carácter es el que está en posición 4 (que se ha de comparar con el que está en posición 3), y así sucesivamente. Como los nuevos caracteres consisten en C consecutivos, cada uno de los cuales concuerda con el anterior, la extensión de cadena continúa hasta el sexto nuevo C. La longitud de cadena total (longitud de cadena de palabra de código 4) + (número de caracteres de extensión hasta

ese momento) = 2 + 7 = 9) es aún menor que el máximo, por lo que el procedimiento de extensión de cadena continúa.

Historia del codificador:

Posición	0	1	2	3	4	5	6	7	8	9										
Carácter	C	C	C	C	C	C	C	C	C	C										

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	Z	..
Índice descendente	-	-	4	-	-	-	-	-	-	-

Árbol de nodos:

Palabra de código	4												
Posición del primer carácter	1												
Longitud de segmento	1												
Índice descendente	-												
Índice lateral	-												

Transferido: No se transfiere nada en esta subetapa.

Análisis del decodificador: No se recibe nada en esta subetapa.

Conjunto de cadenas:

Palabra de código	4												
Posición del último carácter	1												
Longitud de cadena	2												

Historia del decodificador:

Posición	0	1	2																	
Carácter	C	C	C																	

Etapa 5: Nuevos caracteres: X

Análisis del codificador: X termina la concordancia de extensión de cadena: el codificador encuentra que no concuerda el carácter en posición 9 de la historia (el décimo C). El codificador crea un nuevo segmento de cadena (palabra de código 5) para ampliar la concordancia de cadena más larga con una longitud de extensión de 7 (que representa "CCCCCCC" que siguen a la concordancia de cadena más larga, palabra de código 4); La posición del primer carácter es 3. Esto termina el procedimiento de extensión de cadena, y el codificador no anexa un segmento de cadena de 1 carácter.

El carácter no concordado "X" se convierte en la raíz de una nueva cadena posible. Sin embargo, como no tiene índice descendente válido, es transferido como un ordinal; se crea la palabra de

código 6 para anexar el segmento de cadena de 1 carácter para este carácter que sigue a "X"; y el índice descendente para la entrada raíz de "X" es actualizado con la palabra de código 6.

Historia del codificador:

Posición	0	1	2	3	4	5	6	7	8	9	10									
Carácter	C	C	C	C	C	C	C	C	C	C	X									

Matriz de raíces:

Raíz	A	B	C	D	E	..	X	Y	Z	..
Índice descendente	-	-	4	-	-	-	6	-	-	-

Árbol de nodos:

Palabra de código	4	5	6									
Posición del primer carácter	1	3	11									
Longitud de segmento	1	7	1									
Índice descendente	5											
Índice lateral	-											

Transferido: Longitud de extensión de cadena 7, ordinal para "X"

Análisis del decodificador: El decodificador amplía la cadena de la palabra de código anterior con 7: copia los 7 caracteres en la historia que sigue inmediatamente a la cadena representada por la palabra de código 4, que utiliza la posición del último carácter (posición 1) más 1 como su punto de partida. Esto se hace carácter por carácter: el punto de partida para copiar es la posición 2 (que se ha de copiar en la primera posición disponible, que es la posición 3), el siguiente carácter que se ha de copiar es el de la posición 3 (que se ha de copiar en la posición 4), y así sucesivamente. El decodificador crea después la cadena que termina con el décimo C, que utiliza la palabra de código 5: su longitud es 9 ((longitud de cadena para palabra de código 4) + (longitud de extensión de cadena) = (2 + 7) = 9); y la posición del último carácter de la cadena es la posición del último carácter copiado en la historia, 9.

El carácter "X" se coloca después en la historia en la posición 10. No se crea una cadena de 2 caracteres.

Conjunto de cadenas:

Palabra de código	4	5										
Posición del último carácter	1	9										
Longitud de cadena	2	9										

Historia del decodificador:

Posición	0	1	2	3	4	5	6	7	8	9	10									
Carácter	C	C	C	C	C	C	C	C	C	C	X									

Bibliografía

– UIT-T V.14 (1993), *Transmisión de caracteres arrítmicos por canales portadores síncronos.*

– UIT-T V.76 (1996), *Multiplexor genérico que utiliza procedimientos basados en LAPM de la Recomendación V.42.*

– ISO/CEI 10646-1:2000, *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane.*

– Unicode Technical Report No. 6, *A Standard Compression Scheme for Unicode,* Revision 3.1.

SERIES DE RECOMENDACIONES DEL UIT-T

Serie A	Organización del trabajo del UIT-T
Serie B	Medios de expresión: definiciones, símbolos, clasificación
Serie C	Estadísticas generales de telecomunicaciones
Serie D	Principios generales de tarificación
Serie E	Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos
Serie F	Servicios de telecomunicación no telefónicos
Serie G	Sistemas y medios de transmisión, sistemas y redes digitales
Serie H	Sistemas audiovisuales y multimedios
Serie I	Red digital de servicios integrados
Serie J	Transmisiones de señales radiofónicas, de televisión y de otras señales multimedios
Serie K	Protección contra las interferencias
Serie L	Construcción, instalación y protección de los cables y otros elementos de planta exterior
Serie M	RGT y mantenimiento de redes: sistemas de transmisión, circuitos telefónicos, telegrafía, facsímil y circuitos arrendados internacionales
Serie N	Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión
Serie O	Especificaciones de los aparatos de medida
Serie P	Calidad de transmisión telefónica, instalaciones telefónicas y redes locales
Serie Q	Conmutación y señalización
Serie R	Transmisión telegráfica
Serie S	Equipos terminales para servicios de telegrafía
Serie T	Terminales para servicios de telemática
Serie U	Conmutación telegráfica
Serie V	**Comunicación de datos por la red telefónica**
Serie X	Redes de datos y comunicación entre sistemas abiertos
Serie Y	Infraestructura mundial de la información y aspectos del protocolo Internet
Serie Z	Lenguajes y aspectos generales de soporte lógico para sistemas de telecomunicación

Ginebra, 2001