UNIVERSIDAD Privada DR. RAFAEL BELLOSO CHACÍN

TÉLÉMATIQUE

ANÁLISIS TEÓRICO DEL ALGORITMO DE COMPRESIÓN LLRUN PARA LA COMPRESIÓN DE CADENAS DE BITS SPARSE

(Theoretical analysis LLRUN compression algorithm for compressing bit strings SPARSE)

Recibido: 14/04/2015 **Aceptado:** 05/06/2014

Rincón, Carlos
Universidad del Zulia, Venezuela
crincon@fec.luz.edu.ve

Bracho, David
Universidad del Zulia, Venezuela
drbracho@fec.luz.edu.ve

Acurero, Alfredo
Universidad del Zulia, Venezuela
aacurero@fec.luz.edu.ve

RESUMEN

La presente investigación tuvo como finalidad realizar un análisis teórico sobre el funcionamiento del algoritmo de compresión LLRUN, para determinar su posible uso en la compresión de cadenas de bits sparse. Fraenkel y Klein (1985) plantearon un método para comprimir cadenas de bits sparse mezclando las técnicas run length encoding (RLE), Gamma coding y Huffman coding, mostrando un rendimiento prometedor. El resultado del presente estudio permitió determinar que LLRUN ofrece una relación de compresión teórica entre 87,5% y 50%, siendo una herramienta efectiva para la compresión de cadenas de bits sparse.

Palabras clave: compresión, cadenas de bits sparse, LLRUN, rendimiento.

ABSTRACT

The purpose of this research was to make a theoretical analysis about the performance of LLRUN compression algorithm on sparse bit strings. Fraenkel and Klein (1985) proposed a method to compress sparse bit strings, using run length encoding (RLE), Elias - gamma coding and Huffman coding, with promising performance. The results of this research showed that LLRUN offers a theoretical compression ratio between 87,5% and 50%, being an effective tool to compress sparse bit strings.

Keywords: compression, sparse bit strings, LLRUN, performance.

Depósito Legal: PPX200002ZU2142 / ISSN: 1856-4194. Volumen 14 Edición No 2 Julio – Diciembre 2015

INTRODUCCIÓN

En la actualidad la compresión de datos es una de las áreas de las ciencias computacionales de mayor auge, como consecuencia de la necesidad creciente de almacenamiento / transmisión de información por porte de los usuarios.

Un caso particular de la compresión de datos es la compresión de cadenas de bits sparse.

Fraenkel y Klein (1985) plantearon el método LLRUN para comprimir cadenas de bits sparse, mezclando las técnicas Run Length Encoding (RLE), Elias - Gamma Coding y Huffman Coding.

A pesar de mostrar resultados prometedores, es escasa la información de índole académica sobre el funcionamiento de este algoritmo, siendo esta una de las razones que fundamentó la presente investigación.

Rincón y col. (2009) presentaron un nuevo algoritmo de compresión basado en la teoría de la información. Estudios adicionales sobre el rendimiento de este algoritmo, realizado por Rincón y col. (2012), permitieron determinar un efecto adverso del tamaño del alfabeto en el rendimiento objeto de estudio, generando una cadena de bits sparse como resultado de la ejecución del algoritmo.

La necesidad de contar con una herramienta para la compresión de cadenas de bits sparse fue la otra razón que llevó a los investigadores a estudiar el algoritmo LLRUN.

TEORÍA DE LA INFORMACIÓN

El artículo “Una teoría matemática de la comunicación” de Shannon (1948) revolucionó el concepto que hasta la fecha se tenía sobre la información.

Sin embargo, en la actualidad, la definición que se maneja sobre la información (inclusive por miembros de la comunidad de la computación) es la unión de un conjunto de datos organizados que representan algo. El problema básico de esta definición es que no estipula el mecanismo que permita “medir” la información.

A pesar de la complejidad matemática implícita en los estudios de Shannon, la fundamentación de estos es relativamente sencilla. Para Shannon, la información puede ser medida mediante una función representada por la inversa de la probabilidad de ocurrencia del evento que se está estudiando. Específicamente, Shannon define la información mediante la fórmula $I(x) = \log X(1/P(x))$, donde x es el evento estudiado.

COMPRESIÓN DE DATOS

La compresión de datos es una técnica que permite representar la información generada por una fuente de datos con una menor cantidad de bits, utilizando para este fin, códigos óptimos.

Depósito Legal: PPX200102ZU2313 / ISSN: 1856-6189. Volumen 14 Edición No 2 Julio – Diciembre 2015

Las soluciones de compresión se dividen en 2 pasos:

(a) Algoritmo de compresión: es aquel que determina un código óptimo que permita representar los datos con la menor cantidad de bits y que genera un archivo comprimido basándose en los códigos óptimos encontrados.

(b) Algoritmo de descompresión: es aquel que se encarga de transformar el archivo comprimido en la información original.

RELACIÓN DE COMPRESIÓN

Es una métrica que permite determinar el rendimiento de un algoritmo de compresión. Matemáticamente puede definirse de dos formas:

(a) Relación n a 1: RC = TO/TC y

(b) Como porcentaje: RC = (TO-TC)/TO, donde TO = tamaño del archivo original y TC = tamaño del archivo comprimido.

A pesar que la relación de compresión es planteada como una función cuyas variables independientes son el tamaño del archivo original y el tamaño del archivo comprimido, esta métrica puede ser simplificada considerando que el tamaño del archivo original es igual al número de símbolos del archivo multiplicado por el tamaño promedio de los símbolos sin comprimir (N_{SC}), y que el tamaño del archivo comprimido es igual al número de símbolos del archivo multiplicado por el tamaño promedio de los símbolos comprimidos (N_C). Considerando lo antes expuesto, se puede concluir que RC = N_{SC}/ N_C (relación n a 1) y RC = (N_{SC} - N_C)/ N_{SC} *100 (porcentaje).

RUN LENGTH ENCODING (RLE)

Golomb (1966) planteó un método para codificar una fuente donde se presenten una cantidad sucesiva de eventos desfavorables. Al aplicar la teoría de la información, el algoritmo RLE codifica la cantidad de eventos desfavorables (Run Lengths), minimizando la información generada.

El comportamiento probabilístico de la codificación de un Run Length de tamaño n es igual p^nq, para n = 0,1,2,3, …., siendo similar a la distribución geométrica.

Para la presente investigación, se utilizó la versión RLE de 8bit, dividiendo la cadena sparse en octetos y codificando la posición dentro de cada octeto de los bits encendidos.

Es importante resaltar que al dividir la cadena en octetos, se minimiza a 3 la cantidad de bits necesaria para representar una posición, pero se tiene ambigüedad (código no es de traducción única) en el caso que la posición del primer bit encendido del octeto actual es mayor que la última posición del último bit encendido del octeto anterior. Ejemplo:

Depósito Legal: PPX200102ZU2313 / ISSN: 1856-6189. Volumen 14 Edición No 2 Julio – Diciembre 2015

Cadena de Bits:

01000000 00000001 00010000,

Cadena RLE 8bits= 2 8 4

Decodificación cadena RLE =

01000001 00010000

Para evitar este error, se decidió sumar ocho (8) a todas las posiciones encendidas de un octeto, sólo en el caso que la posición del primer bit encendido del octeto actual sea mayor que la última posición del último bit encendido del octeto anterior. Al conseguir el primer valor mayor que 8, se conoce que es el nuevo octeto aquel cuyos bits encendidos serán aquellos valores de las posiciones (sucesivas a la primera encontrada) que sean mayores que 8. La posición real es aquella que se obtiene de restar 8 a las posiciones mayores que 8.

Al aplicar esta modificación al ejemplo anterior tenemos:

Cadena de Bits:

01000000 00000001 00010000,

Cadena RLE 8bits= 2 16 4

Decodificación cadena RLE =

01000000 00000001 00010000

CODIFICACIÓN ELIAS – GAMMA

Elias (1975), planteó un método para representar números enteros utilizando $2Log_2(X) + 1$ bits, siendo X el número entero a codificar.

En resumen, el método de codificación consiste en representar el número entero en 2 partes.

La primera parte (el prefijo) representa la potencia de 2 más cercana al número X, tal que $2^N \leq X$. La segunda parte (el sufijo), es la representación binaria del número que se debe sumar a 2^N para ser igual a X. El tamaño del sufijo es igual a la cantidad de bits del prefijo - 1.

Para la implementación del algoritmo LLRUN en la presente investigación, se utilizó la representación en codificación Elias – Gamma, de los primeros 16 números (ver Tabla 1).

Depósito Legal: PPX200102ZU2313 / ISSN: 1856-6189. Volumen 14 Edición No 2 Julio – Diciembre 2015

Tabla 1. Codificación Elias - Gamma

Prefijo	Sufijo	Cálculo	Valor
1	N/A	2^0 + N/A	1
01	0	2^1 + 0	2
01	1	2^1 + 1	3
001	00	2^2 + 0	4
001	01	2^2 + 1	5
001	10	2^2 + 2	6
001	11	2^2 + 3	7
0001	000	2^3 + 0	8
0001	001	2^3 + 1	9
0001	010	2^3 + 2	10
0001	011	2^3 + 3	11
0001	100	2^3 + 4	12
0001	101	2^3 + 5	13
0001	110	2^3 + 6	14
0001	111	2^3 + 7	15
00001	0000	2^4 + 0	16

Fuente: elaboración propia.

Para los prefijos, se utiliza una representación inversa de la cadena de bits que representa la potencia de 2, con la finalidad de utilizar el bit encendido como delimitador entre el prefijo y el sufijo.

En la implementación de la codificación Elias – Gamma de la presente investigación, se descartó el sufijo del valor 16, por ser el único valor cuyo prefijo tiene 5 bits y ser igual a cero.

ALGORITMO DE COMPRESIÓN DE HUFFMAN

Huffman (1952) planteó una técnica que permite generar códigos óptimos para realizar el proceso de compresión. Está técnica de compresión sin pérdida se conoce en la actualidad como el algoritmo de compresión de Huffman, el cual mantiene su vigencia a pesar de la introducción de nuevos algoritmos para generar códigos óptimos.

Los pasos que estipuló Huffman para obtener códigos óptimos son:

1. Calcular las frecuencias de los símbolos que aparecen en el mensaje.

2. Ordenar los símbolos según su frecuencia de menor a mayor. Deben considerarse criterios para determinar el orden en los casos donde los símbolos tienen la misma frecuencia.

Depósito Legal: PPX200102ZU2313 / ISSN: 1856-6189. Volumen 14 Edición No 2 Julio – Diciembre 2015

3. Crear un símbolo compuesto que estará conformado por los símbolos con menor frecuencia de la lista producto del paso 2. La frecuencia de este símbolo compuesto será igual a la suma de las frecuencias de los símbolos que lo conforman.

4. Reemplazar los símbolos seleccionados por el símbolo compuesto y reordenar los símbolos del mensaje según su frecuencia.

5. Ir al paso 3 hasta que se tenga un árbol binario cuyo nodo padre sea un símbolo compuesto con una frecuencia igual al número de símbolos del mensaje.

6. Etiquetar las ramas del árbol con el valor de 0 y 1 (se debe establecer y mantener el criterio sobre qué valor que va a tomar la rama izquierda y el valor que va a tomar la rama derecha).

7. Construir los códigos de Huffman (1952) de los símbolos del mensaje uniendo las etiquetas que conforman el camino desde la raíz del árbol hasta el la hoja que representa dicho símbolo. El árbol resultante tendrá tantos nodos hoja como símbolos tenga el mensaje.

ALGORITMO LLRUN

Fraenkel y Klein (1985) presentaron un nuevo método para la compresión de cadenas de bits sparse.

Los pasos de este algoritmo son los siguientes:

1. Aplicar RLE a la cadena de bits sparse, utilizando compresión índice para descartar los octetos sin bits encendidos.

2. Aplicar la codificación Elias – Gamma a la cadena RLE.

3. Aplicar el algoritmo de Huffman (1952) a los prefijos producto de la codificación Elias - Gamma.

En la siguiente sección, se presenta el funcionamiento del algoritmo.

RENDIMIENTO DEL ALGORITMO LLRUN

El rendimiento de un algoritmo de compresión se mide determinando su mejor y peor caso. El mejor caso se considera cuando en el mensaje sólo aparece un símbolo del alfabeto mientras que el peor caso es cuando aparecen n símbolos de manera equiprobable en el mensaje.

Mejor Caso:

Para el algoritmo LLRUN, considerando que se trata de una técnica de compresión para cadenas de bits sparse, se utilizó una cadena de 32 bits donde cada uno de los 4 octetos tiene encendido sólo el bit más significativo:

Depósito Legal: PPX200102ZU2313 / ISSN: 1856-6189. Volumen 14 Edición No 2 Julio – Diciembre 2015

Cadena de bits:

10000000 10000000 10000000 10000000

Al ejecutar el RLE de 8bits se obtiene el siguiente arreglo:

Arreglo RLE: 1 1 1 1

Al aplicar la codificación Elias - Gamma al arreglo RLE se obtiene la siguiente cadena:

Cadena GAMMA: 1 1 1 1 (4 bits)

Al aplicar el algoritmo de Huffman (1952) a los prefijos de la cadena Gamma se obtiene el siguiente código:

C (1) = 0

Al sustituir el código producto de la ejecución de Huffman (1952) en la cadena GAMMA se obtiene la cadena:

Cadena Comprimida = 0 0 0 0 (4bits)

Es importante resaltar que para este caso ninguno de los códigos GAMMA tiene sufijo dado a que el número 1 se representa como $2^0 + 0$.

La relación de compresión para el mejor caso es:

RC (mejor caso) = (32 bit – 4 bits)/32 bit *100 = 87,5 %

Peor Caso:

Para el peor caso, se consideró una cadena de 64 bits donde cada uno de los octetos tiene encendido un bit diferente. La razón por la que se consideraron los 64 consistió en cumplir la característica de equiprobabilidad que deben cumplir los símbolos en el mensaje.

Cadena de bits: 10000000 01000000 00100000 00010000 00001000 00000100 00000010 00000001

Al ejecutar el RLE de 8bits se obtiene el siguiente arreglo:

Arreglo RLE: 1 2 3 4 5 6 7 8, sin embargo dada la implementación realizada del algoritmo RLE de 8bit que permite diferenciar el octeto al que pertenece una posición encendida, el arreglo final es:

Arreglo RLE: 1 10 3 12 5 14 7 16

Al aplicar la codificación Elias - Gamma al arreglo RLE se obtiene la siguiente cadena:

Depósito Legal: PPX200102ZU2313 / ISSN: 1856-6189. Volumen 14 Edición No 2 Julio – Diciembre 2015

Cadena GAMMA: 1 0001 010 01 1 0001 100 001 01 0001 110 001 11 00001

(40 bits)

Al aplicar el algoritmo de Huffman (1952) a los prefijos de la cadena Gamma se obtienen los siguientes códigos:

C(0001) = 11, C(001) = 01, C(01) = 101, C(00001)=00, C(1)=100

Al sustituir los códigos producto de la ejecución de Huffman (1952) en la cadena GAMMA se obtiene la cadena:

Cadena Comprimida = 100 11 010 101 1 11 100 01 01 11 110 01 11 00 (32 bits)

La relación de compresión para el peor caso es:

RC (peor caso) = (64 bit – 32 bits)/32 bit *100 = 50 %

Es importante resaltar que para el cálculo de la relación de compresión no se considera la información adicional que necesita Huffman (1952) y la codificación por índice para aplicar la descompresión. Las razones que fundamentan este criterio se derivan del aporte mínimo que significa esta información adicional, cuando se comprimen archivos de Megabits o Gigabits de información.

CONCLUSIONES Y TRABAJOS FUTUROS

Luego del análisis teórico realizado, se determinó que el rendimiento que ofrece el algoritmo LLRUN se encuentra entre el 87,5% y el 50% (sin considerar la información de cabecera de las técnicas de compresión utilizadas). Este rendimiento permite proponer a LLRUN como una opción efectiva para la compresión de cadenas de bits sparse.

Según Rincón y col. (2012), el algoritmo de Compresión Probabilístico, basado en la posición de los símbolos, genera cadenas de bits sparse para alfabetos con tamaño mayor a 15. La próxima investigación consistirá en aplicar el algoritmo LLRUN para tratar de optimizar el rendimiento del algoritmo, propuesto por Rincón y col. (2009).

REFERENCIAS BIBLIOGRÁFICAS

Elias, P. (1975). Universal codeword sets and representations of the integers. IEEE Transactions on Information Theory. Volumen 21, edición 2. (194-203).

Fraenkel, A. y Klein, S. (1985). Novel compression of Sparse Bit-Strings-Preliminary report. En Apostolico, A. y Galil, Z. (editors). Combinatorial Algorithms on Words. Volumen 12, NATO ASI Series F. Estados Unidos. Springer-Verlag.

Golomb, S. (1966). Run-length encodings (Corresp.). Information Theory, IEEE Transactions. Volumen 12, número 3. (Pp. 399-401).

Depósito Legal: PPX200102ZU2313 / ISSN: 1856-6189. Volumen 14 Edición No 2 Julio – Diciembre 2015

Huffman, D. (1952). A method for the construction of minimum redundancy codes. Proceedings of the IRE. Volumen 40, edición 9. (Pp. 1098-1101).

Rincón, C.; Rodríguez, D.; Acurero, A.; Bracho, D. y Jakymec, J. (2009). Algoritmo de compresión probabilístico basado en la teoría de la información. Octava Conferencia Iberoamericana en Sistemas, Cibernética e Informática, CISCI 2009. Julio, Estados Unidos.

Rincón, C.; Bracho, D.; Acurero, A. (2012). Efecto del tamaño del alfabeto en el rendimiento de un algoritmo de compresión probabilístico. Revista Enl@ce. Volumen 13, número 3. (Pp. 83-93).

Shannon, C. (1948). A mathematical theory of communication. The Bell Systems Technical Journal. Volumen 27. (Pp. 379-423, 623-656).

Depósito Legal: PPX200102ZU2313 / ISSN: 1856-6189. Volumen 14 Edición No 2 Julio – Diciembre 2015