CIENCIA 21(2), 90 - 98, 2013
Maracaibo, Venezuela

Evaluación de técnicas de compresión para cadenas de bits sparse

Carlos Rincón, David Bracho, Alfredo Acurero y Francisco Salas*
Unidad de Redes e Ingeniería Telemática, Departamento de Computación, Facultad Experimental de Ciencias, Universidad del Zulia. Maracaibo, Venezuela

Recibido: 10-10-12 Aceptado: 28-05-13

Resumen

El propósito de la presente investigación consistió en medir el rendimiento de las técnicas de compresión *Run Length Encoding (RLE)*, Huffman y compresión por índice, sobre cadenas de bits sparse generadas a través del algoritmo de Compresión Probabilístico basado en la Teoría de Información. El trabajo utilizó una metodología experimental propuesta por Rincón y colaboradores, manipulando las variables dependientes (relación y tiempo de compresión) e independientes (tamaño del archivo, tamaño del alfabeto y las técnicas de compresión). Los resultados obtenidos mostraron para la variable tiempo de compresión, que los algoritmos de Huffman y RLE son 60% más rápidos que la compresión por índice, mientras que para la variable relación de compresión el mejor resultado se obtuvo con el algoritmo de Huffman. De las tres técnicas investigadas el algoritmo de Huffman modificado fue el que ofreció los mejores resultados. Se hizo un análisis estadístico dividido en dos partes, la primera sin bloquear la variable independiente técnica de compresión y la segunda bloqueando la misma. De estos análisis se concluye que el tamaño del alfabeto influye en las variables dependientes, y que mientras mas sparse es el archivo, mayor será la relación de compresión y menor el tiempo empleado en comprimirlo.

Palabras clave: compresión, rendimiento, sparse.

Evaluation of compression techniques for sparse strings

Abstract

The purpose of this research was to measure the performance of the compression techniques Run Length Encoding (RLE), Huffman and hierarchical, on sparse bit strings generated by the Probabilistic Compression Algorithm based on Information Theory. This work used an experimental methodology employed by Rincon et al, manipulating the dependent variables (compression time and compression ratio) and the independents variables (file size, alphabet size and compression techniques). The results showed that for compression time, Huffman and RLE algorithms are 60% faster than hierarchical, while for compression ratio, the best performance was obtained using Huffman algorithm. Overall, of the three studied techniques, the Huffman modified algorithm was the one that offered the best results. Statistical analysis was divided into two parts, the first without blocking the independent variable compression technique and the second blocking it. From this analysis we conclude that the alphabet size influences the depen-

Autor para la correspondencia: crincon@fec.luz.edu.ve

dent variables, and while the file is more sparse, we obtain a higher compression ratio and a lower compression time.

Keywords: compression, performance, sparse.

Introducción

Cada día recibimos más y más información, mucha de esta información no sería capaz de llegar hacia nosotros de no ser por la compresión de datos. Todas las imágenes, videos y audio que existen en la web están comprimidos, muchos sistemas de archivos comprimen la información al ser almacenada y el resto lo hacemos nosotros.

En 1948, Claude Shannon (1) se encargó de definir un método matemático que permitiera cuantificar la información generada por una fuente de datos. Shannon logró determinar lo que hoy llamamos teoría de la información, utilizando un concepto relativamente sencillo: considerando que todo evento tiene una probabilidad de ocurrencia, la cantidad de información producto de este evento puede cuantificarse mediante una función representada como la inversa de la probabilidad de ocurrencia de dicho evento. De este concepto se concluye que un evento con menor probabilidad de ocurrencia generará mayor información que un evento con mayor probabilidad de ocurrencia.

La proliferación de información a grandes escalas representa un desafío en términos de almacenamiento, manipulación, recuperación de la misma. Debido a esto múltiples técnicas para la compresión de datos han sido desarrolladas para ganar desempeño (disminuir tiempo) en transmisiones así como para economizar espacio en dispositivos. La compresión de datos consiste en la reducción del volumen de información tratable (procesar, transmitir o grabar). En principio, con la compresión se pretende transportar la misma información, pero empleando la menor cantidad de espacio.

Para resolver el problema planteado, la ciencia de la computación utilizó las mismas técnicas aplicadas en el pasado para resolver problemas similares. Mediante la codificación eficiente de los símbolos, se obtiene la representación de la información original utilizando una menor cantidad de unidades de información.

La compresión de datos es el proceso que permite encontrar la codificación óptima para representar los datos generados por una fuente, con la finalidad de minimizar la cantidad de información producida por la misma. Según Sayood (2), los algoritmos de compresión se clasifican en algoritmos con pérdida (cuando la información que resulta del proceso de compresión no es exactamente igual a la información original) y algoritmos sin pérdida (cuando la información que resulta del proceso de compresión es exactamente igual a la original). El uso de técnicas de compresión con o sin pérdida dependerá de las características de la información a comprimir.

Al principio de los años 50 David Huffman (3) planteó una técnica la cual permite generar códigos óptimos para realizar el proceso de compresión, Huffman se basó en la teoría de la información propuesta en 1948 por su profesor en el Instituto Tecnológico de Massachussets Claude Shannon, esta técnica se conoce en la actualidad como el algoritmo de Huffman. Este algoritmo toma un alfabeto de n símbolos, junto con sus frecuencias de aparición asociadas, y produce un código de Huffman para ese alfabeto y esas frecuencias. Algunas implementaciones del algoritmo de Huffman son adaptativas, actualizando las frecuencias de cada símbolo conforme recorre el texto.

A través de los años, la compresión de datos ha ganado gran popularidad a nivel de usuarios, producto de que estos tienden a acumular grandes archivos de datos y detestan la idea de desecharlos, no importa que tan grande sea la unidad de almacenamiento

que estas personas posean tarde o temprano se va a saturar. La compresión de datos retarda lo inevitable.

La investigación "Un algoritmo de compresión probabilístico basado en la teoría de la Información" (4) plantea un algoritmo de compresión sin pérdida el cual fue diseñado usando la posición de los símbolos para crear cadenas de bits comprimidas.

Esta técnica presenta un problema, a medida que el alfabeto crece la cadena de bits generada será mas sparse, lo cual afecta negativamente el rendimiento del algoritmo. Salomon (5) define sparse como un archivo de entrada representado como una cadena de bits, donde la mayoría de los bits son cero.

El propósito del presente trabajo, consistió en el estudio de distintas técnicas de compresión a cadenas de bits sparse las cuales serán previamente generadas con el algoritmo de compresión probabilístico basado en la teoría de la información, con la finalidad de mejorar el rendimiento del algoritmo.

Motivación

El algoritmo de compresión de Huffman, fue diseñado en 1952 por David Huffman fundamentado en la teoría de la información propuesta en 1948 por su profesor en el Instituto Tecnológico de Massachussets, Claude Shannon. A pesar del tiempo que ha pasado desde su diseño y el desarrollo de nuevos algoritmos de compresión de datos, Abraham Bookstein (6) en su investigación titulada "Is Huffman Coding Dead?, concluyó que el algoritmo de Huffman puede utilizarse actualmente para resolver diferentes problemas informáticos que requieran de la compresión de datos.

Una de las aplicaciones en las que actualmente se utiliza el algoritmo de compresión de Huffman es la académica, como lo demuestran los trabajos de Mohamed Hamada (7) y James Keeler (8), donde se implementa el algoritmo de Huffman para demostrar los conceptos implícitos en la teoría de la información.

El algoritmo de Huffman posee ciertas limitaciones ya que es necesario conocer de antemano las frecuencias de aparición de cada símbolo, y su eficiencia depende de lo próximas a las frecuencias reales que sean las estimadas.

Sayood (9) explica cómo se puede medir el rendimiento de una técnica de compresión, se puede medir la complejidad relativa de un algoritmo, la memoria requerida para implementarlo, que tan rápido se desempeña en una maquina predeterminada, la cantidad de información comprimida, y que tanto la reconstrucción del archivo se asemeja al original.

Algunos análisis se enfocan solo en el tamaño para así evitar dependencias de hardware. Mahoney (10) propone otra forma de medir la proporción de compresión es a través de los bits por carácter (bpc) bits comprimidos por byte sin comprimir. Ya sea por el tamaño o por el bpc en ambos casos mientras menor sea el número mejor. 8 bpc significa que no hay compresión. 6 pbc significan un compresión de 25% del 75% del archivo original.

El propósito de la investigación planteada consistió en medir el efecto de la variación del tamaño del archivo, las técnicas de compresión y el tamaño del alfabeto en el rendimiento del algoritmo de Rincón y col (2009), con la finalidad de poder identificar, con un análisis estadístico, cuál de las técnicas escogidas ofrece el mejor desempeño al comprimir cadenas de bits sparse.

Metodología

La metodología utilizada es una modificación de la metodología usada en la investigación "*Efecto del tamaño del archivo, la entropía y el tamaño del alfabeto en el rendimiento del algoritmo de Huffman*" (11). En ella se mide el efecto de un conjunto de

variables independientes sobre unas variables dependientes, definidas a continuación:

Definición de las variables independientes

Para el presente estudio se definieron las siguientes variables independientes con sus respectivos valores:

1) Tamaño del alfabeto: representa la cantidad de símbolos del alfabeto ASCII que se utilizaron para construir el archivo. Los valores seleccionados fueron: cuatro (4), ocho (8), doce (12) y dieciséis (16) caracteres.

2) Tamaño del archivo: representa la dimensión que tendrán los archivos de pruebas, los tamaños seleccionados son cien (100), doscientos cincuenta (250) y quinientos (500) megabytes.

3) Las técnicas de compresión: Las técnicas de compresión elegidas fueron: algoritmo de Huffman, algoritmo RLE y el algoritmo de compresión por índice (compresión jerárquica)

La selección de los niveles de las variables independientes antes definidas, se fundamentó un muestreo no aleatorizado, producto del consenso entre los autores y los expertos en el área estadística.

Definición de las variables dependientes

Para el presente estudio se definieron como variables dependientes los siguientes parámetros producto de la ejecución del algoritmo de Huffman:

1) Relación de compresión: métrica que permite determinar el grado de minimización en la representación de un archivo.

2) Tiempo de compresión: es la cantidad de unidades de tiempo que se tardan las técnicas implementadas en realizar el proceso de codificación de un archivo.

Determinación del número de repeticiones a realizar por experimento

Para garantizar la validez del análisis estadístico de los datos y considerando los grados de libertad del modelo estadístico, se determinó que eran necesarias como mínimo 4 repeticiones.

Generación de los archivos de prueba

Conociendo el número de repeticiones, se procedió a generar los 48 archivos producto con el algoritmo generador de archivos de prueba. El número proviene de la iteración de las variables independientes (4 niveles del tamaño del alfabeto, 3 niveles del tamaño del archivo) por las 4 repeticiones, con la finalidad de aplicar a cada uno de estos las tres técnicas seleccionadas para obtener los valores de las variables dependientes.

Ejecución de las técnicas de compresión

Generados los archivos de prueba, se procedió a implementar los algoritmos de en un lenguaje de alto nivel (C++), para luego aplicar el algoritmo a estos archivos. Como resultado de esta aplicación, se obtuvieron los valores de las variables dependientes necesarios para poderlos analizar estadísticamente.

Análisis estadístico de los resultados

El Diseño seleccionado para el estudio de las variables dependientes Relación de Compresión (RCOMP) y Tiempo de Compresión (TCOMP), fue un Bloques al Azar utilizando el Tipo de algoritmo como factor de bloqueo y los tratamientos en un arreglo factorial 3×2; es decir, un arreglo factorial de dos factores, uno con tres niveles y el otro con dos y tres repeticiones. Para el Análisis de los Datos se utilizó el Procedimiento ANOVA del Paquete Estadístico SAS (Statistical Analysis System). El Modelo Aditivo Lineal que representa el comportamiento de cada una de las variables en estudio es el siguiente:

$$Y_{ijk} = \mu + A_i + B_j + C_k + (AB)_{ij} + (AC)_{ik} + (BC)_{jk} + (ABC)_{ijk} + E_{ijk}$$

donde:

i: 1,2,3,4;

j: 1,2,3;

k: 1,2,3;

Las variables del modelo son las siguientes:

μ: es la media de la variable en estudio.

A_i: efecto del i-esimo tamaño del alfabeto.

B_j: efecto del j-esimo tamaño de archivo.

C_k: efecto del k-esimo tipo de algoritmo.

$(AB)_{ij}$: efecto de la interacción entre el i-esimo tamaño del alfabeto y el j-esimo tamaño de archivo.

$(AC)_{ik}$: efecto de la interacción entre el i-esimo tamaño del alfabeto y el k-esimo tipo de algoritmo.

$(BC)_{jk}$: efecto del j-esimo tamaño del archivo y el k-esimo tipo de algoritmo.

$(ABC)_{ijk}$: efecto de la interacción entre el i-esimo tamaño del alfabeto, el j-esimo tamaño del archivo y el k-esimo tipo de algoritmo.

E_{ijk} : Error experimental.

Adicionalmente al análisis de varianza, que nos permite determinar la significancia de las variables independientes y sus iteracciones sobre las variables dependientes, se realizó la prueba de medias de Tukey, la cual es un procedimiento de comparación múltiple que permite realizar un análisis a fondo sobre el comportamiento de cada variable independiente, estableciendo grupos dentro de las mismas y determinando la significacia entre grupos. De esta manera se determina, para cada variable dependiente, los niveles de las variables independientes con los que se obtiene mejor y peor resultado, según el comportamiento de la variable a objeto de estudio, analizando para cada grupo si existen diferencias significativas.

Resultados

Después de ejecutar las técnicas de compresión en los archivos de prueba generados, se obtuvieron los siguientes valores de las variables dependientes:

Algoritmo de compresión Run Length Encoding (RLE)

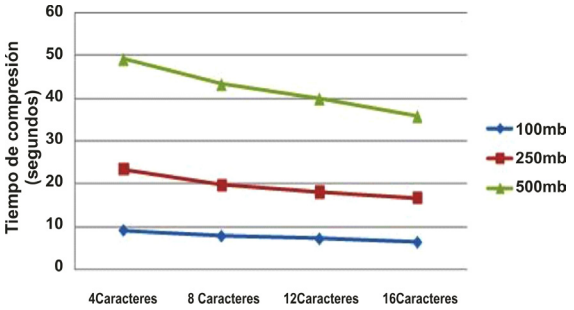
De la figura 1 se desprende que a medida que aumenta el tamaño del archivo, aumenta el tiempo de compresión, obteniéndose un mejor resultado para el tamaño del archivo 100MB. De igual forma se observa que a medida que aumenta el tamaño del alfabeto, disminuye el tiempo de compresión, obteniendo el mejor resultado para el tamaño del alfabeto 16. Para la variable dependiente tiempo de compresión, a menor valor, mejor rendimiento.

En la figura 2, sólo se muestran los resultados de la relación de compresión para el caso de archivos de 500 MB, esto debido a un comportamiento similar para los tamaños de archivo de 100 MB y 250 MB. Se observó que a medida que aumenta el tamaño del alfabeto, aumenta la relación de compresión, obteniéndose el mejor resultado para el tamaño del alfabeto 16. Para la variable dependiente relación de compresión, a mayor valor, mejor rendimiento.

Algoritmo de compresión de HUFFMAN

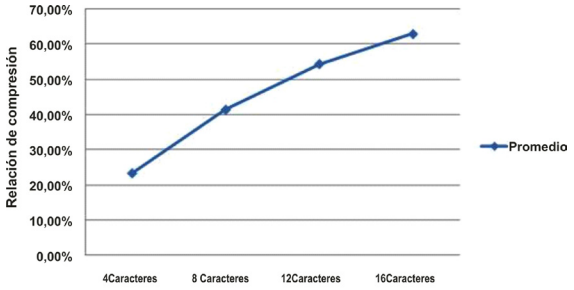
De la figura 3 se desprende que a medida que aumenta el tamaño del archivo, aumenta el tiempo de compresión, obteniéndose un mejor resultado para el tamaño del archivo 100MB. De igual forma se observa que a medida que aumenta el tamaño del alfabeto, disminuye marginalmente el tiempo de compresión, obteniendo el mejor resultado para el tamaño del alfabeto 16. Para la variable dependiente tiempo de compresión, a menor valor, mejor rendimiento.

En la figura 4, sólo se muestran los resultados de la relación de compresión para



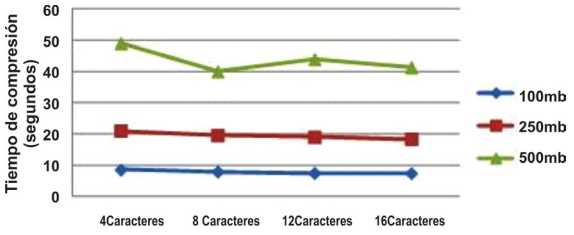
Fuente Propia

Figura 1. Tiempo de compresión para el algoritmo RLE.



Fuente Propia

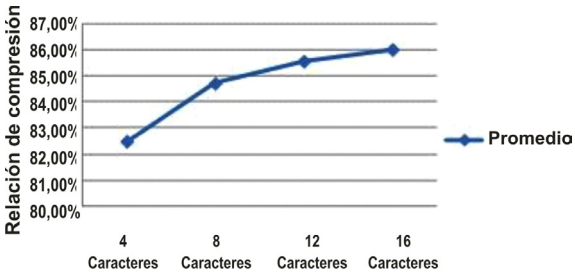
Figura 2. Relación de compresión para el algoritmo RLE.



Fuente Propia

Figura 3. Tiempo de compresión para el algoritmo de Huffman.

el caso de archivos de 500 MB, esto debido a un comportamiento similar para los tamaños de archivo de 100 MB y 250 MB. Se observó que a medida que aumenta el tamaño del alfabeto, aumenta la relación de compresión, obteniéndose el mejor resultado para el tamaño del alfabeto 16. Para la variable dependiente relación de compresión, a mayor valor, mejor rendimiento.



Fuente Propia

Figura 4. Relación de compresión para el algoritmo de Huffman.

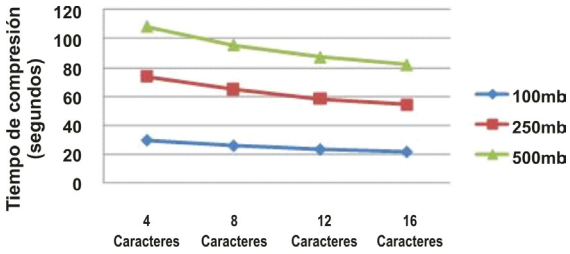
Algoritmo de compresión por índice

De la figura 5 se desprende que a medida que aumenta el tamaño del archivo, aumenta el tiempo de compresión, obteniéndose un mejor resultado para el tamaño del archivo 100MB. De igual forma se observa que a medida que aumenta el tamaño del alfabeto, disminuye el tiempo de compresión, obteniendo el mejor resultado para el tamaño del alfabeto 16. Para la variable dependiente tiempo de compresión, a menor valor, mejor rendimiento.

En la figura 6, sólo se muestran los resultados de la relación de compresión para el caso de archivos de 500 MB, esto debido a un comportamiento similar para los tamaños de archivo de 100 MB y 250 MB. Se observó que a medida que aumenta el tamaño del alfabeto, aumenta la relación de compresión, obteniéndose el mejor resultado para el tamaño del alfabeto 16. Para la variable dependiente relación de compresión, a mayor valor, mejor rendimiento.

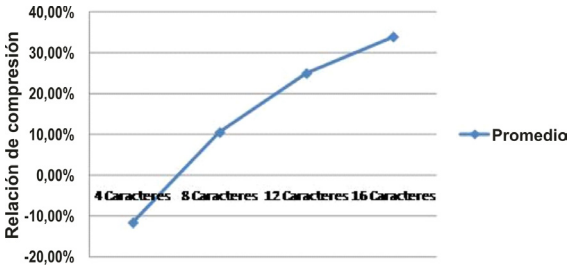
Discusión

Se realizaron dos tipos de análisis, el primero sin bloquear la variable independiente algoritmo de compresión y el segundo bloqueando la misma. La ejecución de estos dos modelos permitió determinar, de las técnicas a objeto de estudio, la mejor técnica de compresión de cadenas de bits sparse.



Fuente Propia

Figura 5. Tiempo de compresión para el algoritmo de compresión por Índice.



Fuente Propia

Figura 6. Relación de compresión para el algoritmo de compresión por Índice.

Análisis estadístico sin la variable bloqueada

En primera instancia se realizó un análisis de la varianza, para determinar el efecto que sobre las dos variables dependientes estudiadas, RCOMP (Relación de compresión) y TCOMP (Tiempo de compresión) tienen el Tipo de Algoritmo, el tamaño del Alfabeto y el tamaño del archivo.

Del primer análisis puede deducirse que ambas variables dependientes son afectadas tanto por el tipo de algoritmo, como por el tamaño del alfabeto, del archivo y sus interacciones.

La prueba de Tukey para la variable RCOMP indica diferencias (P≤0.01) entre los tres tipos de algoritmo, observándose el mayor promedio de RCOMP (84.60) para el Tipo 2 y el promedio más bajo (12.33) para el Tipo 3. En cuanto al TCOMP, se observan diferencias entre el Tipo 3 comparado con los Tipos 1 y 2, sin embargo, no se observan diferencias entre los Tipos 1 y 2.

La prueba de Tukey detectó diferencias significativas para el RCOMP entre los diferentes tamaños del alfabeto, observándose el promedio más alto (60.91) para el Tamaño del alfabeto 16 y el más bajo (31.19), para el tamaño 4.

En cuanto a la variable TCOMP, también se encuentran diferencia significativas (P≤0.01) entre los cuatro tamaños del alfabeto pero esta vez, el promedio más alto de TCOMP se observa para el tamaño 4 (40.993) y el más bajo (31.557), para el tamaño 16.

Análisis estadístico con la variable bloqueada

Tomando en cuenta los resultados previamente expuestos, se decidió entonces realizar el análisis de la varianza para las variables dependientes antes citadas, pero esta vez bloqueando por el tipo de algoritmo. Así, el análisis de la varianza para la variable RCOMP, revela que hay diferencias significativas (P≤0.01) entre los diferentes tamaños del alfabeto, no ocurriendo así para el tamaño del archivo ni tampoco para la interacción entre ellos.

Cuando se realiza el análisis de la varianza para la variable TCOMP se observan diferencias significativas (P≤0.01) entre los diferentes tamaños de archivo, mas no entre los tamaños del alfabeto ni entre la interacción. toda vez que fueron detectadas las diferencias significativas antes mencionadas, se decidió aplicar una prueba de medias utilizando el método de Tukey, la cual en el caso de la variable RCOMP arroja los siguientes resultados: no se observan diferencias significativas (P≤0.01) entre los tamaños del alfabeto 8, 12 y 16, pero si entre estos tres tamaños del alfabeto (8,12 y 16) y el Tamaño 4, observándose el más alto promedio para el Tamaño 16 (60.910) y el más bajo, para el Tamaño 4 (31.190).

La misma prueba de Tukey realizada para la variable dependiente TCOMP revela diferencias significativas entre los tres Tamaños de Archivo (100, 250 y 500), observándose el promedio de TCOMP mas alto (59.326) para el Tamaño 500 y el más bajo (13.534), para el Tamaño 100. En ninguno de los análisis de varianza para las variables estudiadas se refleja interacción significativa.

Conclusiones

Como resultado la observación directa de los resultados y del análisis estadístico de los datos obtenidos mediante la aplicación de los algoritmos de compresión a los diferentes archivos de texto generados variando los parámetros tamaño del alfabeto, tamaño del archivo, se puede concluir lo siguiente:

1) De los cuatro tamaños de alfabeto seleccionados podemos observar que mientras mayor sea este el número de caracteres usados más sparse será el archivo generado, influyendo esto notablemente en la compresión del archivo. Mientras mas sparse es el archivo mayor será la relación de compresión.

2) El tiempo de compresión se ve afectado notablemente por la variable tamaño del alfabeto, mientras menor sea el numero de caracteres usados mayor será el tiempo empleado por las técnicas en comprimir los archivos. Entretanto sí el numero de caracteres usados es mayor menor será el tiempo empleado por los algoritmos en comprimir los archivos de prueba. No ocurre lo mismo con el tamaño del archivo, mientras más grande sea el archivo de prueba mayor será el tiempo empleado por las técnicas para comprimir los mismos.

3) De las tres técnicas estudiadas en esta investigación podemos concluir que el algoritmo de Huffman modificado es el que ofrece los mejores resultados. La relación de compresión siempre fue mayor al 80% en todas las pruebas así como el tiempo de compresión quedo bastante igualado al del algoritmo RLE el cual es el menor.

4) De haber usado la técnica de Huffman sin modificar los resultados obtenidos habrían sido serian nulos, el algoritmo no comprimiría y el archivo resultante sería una copia del original, la compresión la logra por la forma en la cual guarda los caracteres en el archivo y no por el algoritmo de Huffman.

5) El algoritmo de compresión por índice por su parte es de las técnicas estudiadas la que peor rendimiento obtuvo en comparación de las otras, obteniendo una relación de compresión negativa en algunos casos y tiempos de compresión bastante superiores a las otras técnicas.

6) El algoritmo RLE modificado usado en esta investigación ofreció resultados bastante positivos, un tiempo de compresión bastante similar e incluso menor en algunos casos al algoritmo de Huffman, con una relación de compresión en el mejor de los casos superior al 60%.

Referencias bibliográficas

1. SHANNON C. ***Bell Systems Technical Journal*** 27:379-423, 623-656. 1948.
2. SAYOOD, K. ***Introduction to Data Compression.*** Morgan Kaufmann Publishers, Inc. San Francisco, California, USA. 1996.
3. HUFFMAN D. ***In Proc. IRE.*** 40(9): 1098-1101. 1952.
4. RINCÓN C., RODRÍGUEZ D., ACURERO A., BRACHO D., JAKYMEC J. Algoritmo de Compresión Probabilístico basado en la Teoría de la Información. Octava Conferencia Iberoamericana en Sistemas, Cibernética e Informática: CISCI 2009.
5. SALOMON D., MOTTA G. ***Handbook of data compression.*** Editorial Springer. Nueva York (EE.UU.) 2010.

6. BOOKSTEIN A., KLEIN S., RAITA T. ***Proceedings of the 16th Annual international ACM SIGIR Conference on Research and Development in information Retrieval*** (Pittsburgh, Pennsylvania, United States, June 27-July 01, 1993). R. Korfhage, E. Rasmussen, and P. Willett, Eds. SIGIR '93. ACM, New York, NY, 80-87. 1993.

7. HAMADA M. ***Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*** (Covington, Kentucky, USA, March 07-11, 2007). SIGCSE '07. ACM, New York, NY, 60-64. 2007.

8. KEELER J. ***J Comput Small Coll*** 19(5): (May. 2004), 289-290. 2004.

9. SILVA DE MOURA E., NAVARRO G., ZIVIANI N., BAEZA-YATES R. ***ACM Trans. Inf. Syst.*** 18(2): (Apr. 2000), 113-139. 2000.

10. GOPAL L. Modified JPEG Huffman Coding. ***IEEE Transactions on Image Processing*** Vol. 12(2): February 2003, 159-169. 2003.

11. RINCON C., ACURERO A., BRACHO D., JAKYMEC J. ***Ciencia*** Vol. 16, 2008.