ANÁLISIS COMPARATIVO DE CODIFICADORES DE AUDIO SIN PÉRDIDAS

FERNANDO A. MARENGO RODRIGUEZ [1], ERIBERTO A. ROVERI, JUAN MANUEL RODRÍGUEZ GUERRERO, MAURO TREFFILÓ, FEDERICO MIYARA [1]

[1] Universidad Nacional de Rosario, Facultad de Ciencias Exactas, Ingeniería y Agrimensura, Escuela de Ingeniería Electrónica, Laboratorio de Acústica y Electroacústica, Rosario, Argentina.

fmarengorodriguez@yahoo.com.ar
ea_roveri@hotmail.com.ar
jmanu1986@gmail.com
maurotreffilo@gmail.com
fmiyara@fceia.unr.edu.ar

Resumen – *A raíz de la mayor capacidad de transmisión y recursos de almacenamiento disponibles, la codificación de audio sin pérdidas ha ganado popularidad en los audiófilos con respecto a la codificación perceptual. Hasta el momento, se hicieron muchas comparaciones entre los codec sin pérdidas, pero ninguna de ellas nos da noción de cuál codec es el mejor para comprimir un archivo de audio dado en formato PCM estéreo. Este trabajo pretende cubrir esa necesidad, haciendo una comparación exhaustiva entre varios codec populares en términos del factor de compresión y de la velocidad de codificación para piezas musicales de distintos géneros provenientes de CD comerciales.*

1. INTRODUCCIÓN

La compresión de información ha jugado un papel fundamental en la tecnológica digital, ya que su empleo permite minimizar el espacio de almacenamiento de información, así como el ancho de banda necesario para su transmisión entre equipos remotos. Dicha información puede ser de varios tipos, como por ejemplo texto, audio, video y multimedia, y su importancia se mantiene vigente.

Actualmente existen numerosos métodos de compresión de información digital, pero sólo algunos de ellos aprovechan las particularidades de las señales típicas de audio, como ser la relativa predecibilidad a corto plazo y un alto grado de redundancia. Los programas que los implementan son también conocidos como *codec de audio* (codec es una contracción de codificador-decodificador), ya que pueden codificar (comprimir) la información original en un formato digital determinado y decodificar (descomprimir) estos datos al formato original.

Hay dos clases de codificadores de audio masivamente empleados. Por un lado, están los codec perceptuales, que minimizan la redundancia presente en la señal de entrada, además de suprimir información psicoacústicamente no relevante para el oído humano. Estos codec se caracterizan por alcanzar gran compresión —reducen la información de entrada entre 8 y 13 veces—, y algunos ejemplos de éstos son MPEG-1 audio layer 3 —norma internacional popularmente conocido como MP3 [1]— y Ogg Vorbis —de formato libre y abierto [2]—.

Por otro lado, se tienen los codec sin pérdidas (CSP), que minimizan la redundancia de la señal de entrada sin introducir distorsión alguna. Este beneficio se consigue a costa de alcanzar menor compresión —típicamente entre 1,5 y 6 veces [3]— pero su uso se difunde cada vez más entre los audiófilos por varias razones:

- Tecnológicas: El avance de la tecnología permite acceder más fácilmente a recursos de almacenamiento con mayor capacidad de memoria. Asimismo, los canales de transmisión de datos vía Internet aumentan su capacidad. Estos hechos compensan las diferencias de compresión entre los codec perceptuales y los CSP.
- Económicas: Los equipos de reproducción de sonido de alta fidelidad disminuyen de costo con el paso del tiempo, haciéndose más masivo su acceso. En estos equipos se vuelve evidente la distorsión que introduce la codificación perceptual.
- Masterización: Las grabaciones de estudio se pueden transmitir entre equipos remotos en formato comprimido, pero en estos casos no se admite pérdida de información: la única opción viable es usar CSP.
- Medición: Las grabaciones de audio que realizan los especialistas en acústica suelen ocupar espacio

considerable si se contempla solamente el formato WAV. En estos casos no se admite pérdida de información, por lo que el uso de CSP puede constituir una excelente alternativa en aras de economizar el espacio de memoria ocupado.

En la actualidad, el usuario común puede acceder fácilmente a una cantidad importante de CSP diferentes. Ante la pregunta sobre cuáles son los más eficientes, se puede recurrir a estudios comparativos de rendimiento [4]. Sin embargo, no se ha encontrado una publicación revisada donde se compare el desempeño de los CSP más populares en la actualidad cuando se apunta a distintos géneros musicales.

La presente publicación intenta brindar una guía al audiófilo sobre los CSP de mejor rendimiento para algunos géneros musicales ampliamente difundidos en la República Argentina. Se analiza un conjunto de CSP con diversas piezas musicales completas provenientes de discos compactos comerciales, donde las muestras digitales fueron extraídas con tasa de muestreo F_s = 44 100 Hz y resolución n_b = 16 bits en formato PCM estéreo. El rendimiento de los CSP se analizó en términos de la compresión alcanzada y de la demora de procesamiento.

Este trabajo se organiza de la siguiente manera. En la sección 2 se repasa brevemente el principio de funcionamiento de los CSP evaluados y sus variantes funcionales. El análisis comparativo de los CSP propuesto se expone en la sección 3, y sus resultados se mencionan en la sección 4. La sección 5 muestra un algoritmo de selección de CSP óptimos según los requerimientos del usuario en términos de la velocidad de procesamiento y la compresión deseadas.

2. FUNCIONAMIENTO DE LOS CODEC SIN PÉRDIDAS

Básicamente, los codificadores sin pérdidas procesan la señal en cuadros de longitud determinada, según lo ilustra la Figura 1. En cada cuadro se realiza la decorrelación de las muestras de entrada, obteniéndose un conjunto de datos pequeños llamado residuo, que posteriormente se comprime con un codificador de entropía que típicamente es de tipo Rice [4], [5], [6]. Éstos se multiplexan con los parámetros del decorrelacionador para conformar la trama de salida del codificador.

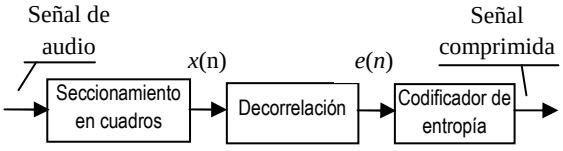

Figura 1: Esquema general del codificador sin pérdidas.

El decorrelacionador minimiza la redundancia existente entre muestras consecutivas de la señal de entrada. Esta operación es importante, pues las señales de audio provenientes de grabaciones de estudio poseen una longitud de correlación extensa [7] que es indispensable suprimir. Los CSP analizados en este trabajo realizan esta tarea en el dominio temporal, mediante la aproximación de la entrada actual $x(n)$ como una combinación lineal de valores pasados de entrada y de salida. Este proceso se conoce como predicción lineal, y es de tipo FIR o IIR, según se contemplen N entradas o M salidas pasadas (ver Figura 2). A fin de reducir la representación de los coeficientes intervinientes, se los cuantifica con una determinada cantidad de bits. La diferencia entre la muestra original y su valor aproximado es el residuo $e(n)$.

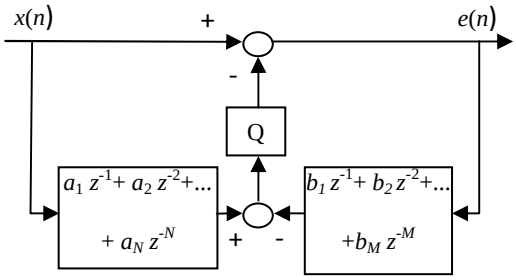

Figura 2: Diagrama de bloques del decorrelacionador empleado en los codificadores sin pérdidas analizados.

En general los CSP funcionan con M=0, constituyendo un modelo autoregresivo –en inglés, autoregressive ó AR-. En este caso, los N coeficientes del predictor pueden ser de valores fijos [5], [8] o bien dependientes de los datos a codificar (codificador lineal predictivo ó LPC). En el primer caso los coeficientes provienen de la aplicación de polinomios de interpolación de Lagrange [6]. En el otro caso, el predictor se ajusta a la señal de entrada, logrando en general mejor compactación a cambio de mayor carga computacional. Esos métodos de ajuste minimizan la energía del error de predicción (norma de mínimos cuadrados) [9] con algoritmos recursivos como Levinson-Durbin [8], o vía redes neuronales [10].

Sólo algunos CSP emplean predictores IIR [11], [12], [13], [14], [20] ya que el ajuste de los coeficientes es más costoso computacionalmente que en el caso FIR, y las mejoras de compresión son a lo sumo modestas [9]. Sin embargo, se dice en [11], [12] que el modelo IIR tiene el potencial de adaptarse mejor a datos con diversas formas espectrales, hecho que puede ser útil si se piensa en la existencia de componentes de banda ancha en la señal de entrada.

En el extremo receptor, el decodificador reconstruye la señal original con los datos provenientes de cada cuadro de la trama codificada. Para dicha reconstrucción, se hace la combinación lineal con los mismos coeficientes empleados en el codificador. A este resultado le suma el residuo previamente decodificado (ver Figura 3).

Existe otra clase de CSP que en lugar de predictores utilizan bloques de transformadas y extraen los coeficientes más relevantes [15]. Otros CSP de im-

portante difusión son Philips [16] y DVD-Audio [11], [12], [13]. Estos codec no se evalúan en este trabajo porque no disponemos sus archivos ejecutables. Philips se basa en la predicción FIR de orden 10 y codificación Rice del residuo. DVD por su parte utiliza un predictor IIR y codifica el residuo con el método Huffman. Se pueden encontrar más referencias sobre el desempeño de estos CSP en [9].

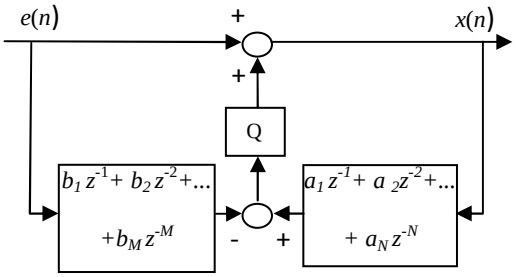

Figura 3: Estructura empleada en los CSP para la reconstrucción de la señal original.

En la Tabla 1 se enumeran los codec analizados y los comandos necesarios para decodificar y codificar archivos. Estos codec fueron seleccionados de acuerdo a su popularidad en la comunidad audiófila, y muchos de ellos poseen varias opciones de codificación. Sin embargo, solamente seleccionamos aquéllas que maximizan su velocidad (rápido) o su compresión (óptimo). Para el análisis propuesto, usamos los archivos ejecutables que fueron provistos por los diseñadores de los correspondientes CSP.

3. ANÁLISIS PROPUESTO

Para analizar el rendimiento de los CSP, se los excitó con información proveniente de CD comerciales de audio. Se eligió más de una docena de piezas musicales de cada uno de los siguientes géneros: folklore, tango, jazz, música clásica, pop, rock y recitales de rock en vivo. Esta última opción se tuvo en cuenta no sólo por su popularidad entre muchos audiófilos, sino también para analizar la influencia del ruido ambiente. La selección de las señales contempló discos y artistas muy conocidos, así como un abanico de ritmos variados en cada rubro.

Se codificó la batería de piezas seleccionadas en diferentes computadoras personales de escritorio, haciendo funcionar cada codec —cuando era posible— en los modos normal (modo por defecto), rápido y óptimo. En todos los casos se midió el rendimiento a través de:

- el *factor de compresión* (*FC*) definido como el cociente entre los tamaños del archivo de entrada y del codificado.
- el *rate* (*R*) definido por el cociente entre el tiempo de reproducción del archivo original y el tiempo de codificación o decodificación según corresponda.

Con fines de simplificación, se trabajó con el valor promedio de estos parámetros para cada género musical.

Codec	Modo		Comando
FLAC [8]	Decodificación		-d -f infile(1) -o outfile(1)
	Cod.	Rápida	-0 -f --no-padding infile -o outfile
		Normal	-5 -f --no-padding infile -o outfile
		Óptima	-8 -f --no-padding infile -o outfile
LPAC [17]	Decod.		-x -v infile outfile
	Cod.	Rápida	-v -1 infile outfile
		Normal	-v -3 infile outfile
		Óptima	-v –5 infile outfile
Monkey's Audio [10]	Decod.		infile outfile -d
	Cod.	Rápida	infile outfile -c1000
		Normal	infile outfile -c2000
		Óptima	infile outfile -c5000
MPEG-4 ALS [14]	Decod.		-x -v infile outfile
	Cod.	Normal	-v infile outfile
		Óptima	-7 -v infile outfile
OptimFrog [18]	Decod.		--decode --verbose --time --overwrite infile --output outfile
	Cod.	Rápida	--verbose –time --mode fast --overwrite infile --output outfile
		Normal	--verbose –time --mode normal --overwrite infile --output outfile
		Óptima	--verbose --time --maximumcompression --overwrite infile --output outfile
Shorten [5]	Decod.		-x infile outfile
	Cod.	Normal	infile outfile
		Óptima	-p12 -b1024 infile outfile
TAK [19]	Decodificación		-d -overwrite infile outfile
	Cod.	Rápida	-e -p0 -overwrite infile outfile
		Normal	-e –p2 -overwrite infile outfile
		Óptima	-e –p4 -overwrite infile outfile
TTA [20]	Decod.		-d infile -o outfile
	Cod.		-e infile -o outfile
WavPack [21] (2)	Decod.		-y infile outfile
	Cod.	Rápida	-f -y infile outfile
		Normal	-y infile outfile
		Óptima	-hx6 -y infile outfile

Nota 1: infile y outfile representan el nombre de los archivos de entrada y salida respectivamente, incluyendo la ruta.

Nota 2: WavPack posee dos ejecutables "wavpack.exe" y "wvunpack.exe" que realizan la compresión y descompresión respectivamente.

Tabla 1: Modos disponibles y comandos utilizados para invocar los CSP estudiados en este trabajo.

La ejecución de los CSP se puede llevar a cabo mediante el empleo de herramientas gráficas de acceso libre [22], pero preferimos hacer uso de una interfaz gráfica de diseño propio que se ajuste mejor a nuestros propósitos de ensayo [23]. Entre otras ventajas, esto nos permitió minimizar y estimar el retardo que ésta introduce, pausar la ejecución del proceso para detectar la presencia de errores y presentar los resultados de interés en forma tabulada. Su aspecto se ilustra en la Figura 4, y la demora que introduce su uso es aproximadamente 0,38 s.

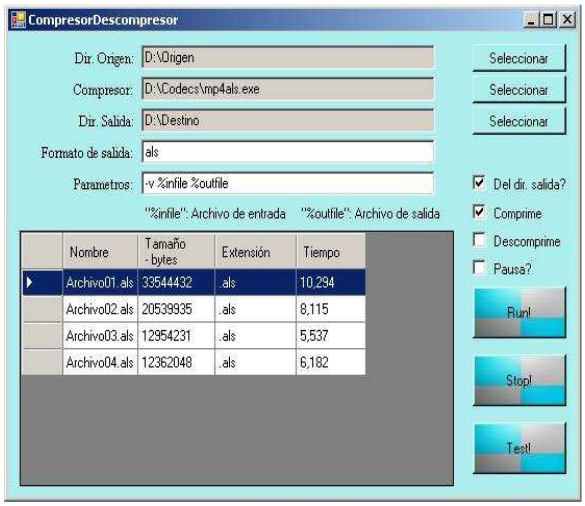

Figura 4: Aspecto de la herramienta gráfica utilizada en nuestro análisis.

Además de la codificación, se contempló la decodificación, no sólo para determinar si hubo pérdida de información alguna luego de recuperar los datos, sino también para conocer los tiempos que implica esta operación. Para estos fines, se hizo uso de la herramienta gráfica antes mencionada.

4. RESULTADOS DEL ANÁLISIS

Se estudió el desempeño de los CSP en las siguientes máquinas:

- PC portátil equipada con placa madre Hewlett-Packard HP Pavilion dv7, procesador AMD Turion II Ultra Dual-Core Mobile M600 de 2,4GHz y memoria SDRAM DDR2-800 de 2 x 2GB. Funcionó con sistema operativo Windows Vista Home Premium 6.1.7600.
- PC de escritorio equipada con placa madre Intel Rock Lake D865PERL, procesador Pentium IV HT de 3 GHz y memoria SDRAM DDR de 2 x 512 MB. Se empleó el sistema operativo Windows XP Professional ServicePack 3.
- PC de escritorio equipada con placa madre MSI K9N6PGM-F/FI, procesador AMD Athlon 64 x 2 Dual Core de 2,2 GHz y memoria SDRAM DDR2 de 2 GB.

  Salvo aclaración contraria, se hará referencia al rendimiento obtenido con la PC portátil y se abrevia el término género musical por sus iniciales GM.

4.1 Codificación en modo normal

El porcentaje de CPU consumido por cada CSP fue similar, ya que estuvo comprendido entre el 24 % y el 28 %.

Al agrupar la información por GM**,** se obtuvo la compresión expuesta en la Figura 5. Se observa que cada GM tiene una zona de operación alrededor de la cual comprimen todos los CSP. Este fenómeno se acentúa si se excluye Shorten. Además, se ordenó la información de cada GM en orden decreciente de compresión, siendo más reducible el jazz y más difícil el pop. En cada GM, el *FC* promedio que se alcanza con cualquiera de los CSP evaluados tiene los valores detallados en la Tabla 2.

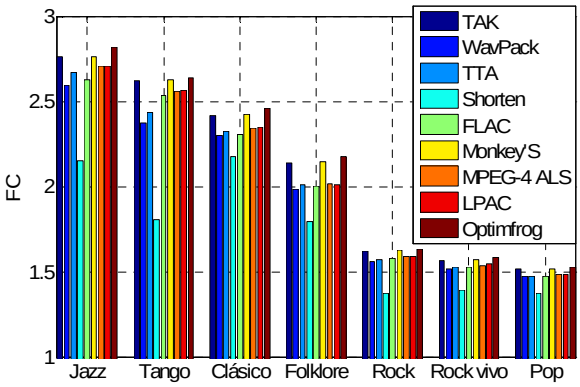

Figura 5: Compresión obtenida con los codificadores analizados en cada género musical.

Jazz	Tango	Música clásica	Folklore	Rock	Rock en vivo	Pop
2,64	2,46	2,34	2,03	1,57	1,53	1,48

Tabla 2: Compresión promedio que se puede obtener en cada género musical.

La Figura 6 ilustra las zonas operativas de cada CSP, en términos de compresión y velocidad. Se identifican los codec por color y los géneros por marcadores. Por un lado, se puede caracterizar cada CSP por su velocidad, siendo TAK, WavPack, TTA y Shorten los más veloces y Optimfrog el más lento. Por otro lado, la velocidad de compresión en cada codec se ve afectada por el género que procesa. Por ejemplo, lleva menos tiempo comprimir tango y más tiempo comprimir música clásica, rock en vivo y pop. Es importante remarcar que la demora en codificar una dada pieza musical se puede calcular dividiendo su tiempo de reproducción por el valor de *R* correspondiente. Por ejemplo, si la pieza musical es folklórica y se reproduce en 3 min, se codificaría con Optimfrog en

3·60/22 = 8,2 s y con TAK en 3·60/120 = 1,5 s. Naturalmente, al multiplicar la diferencia entre estos tiempos por la cantidad de piezas a codificar hará sobresalir al codec más veloz.

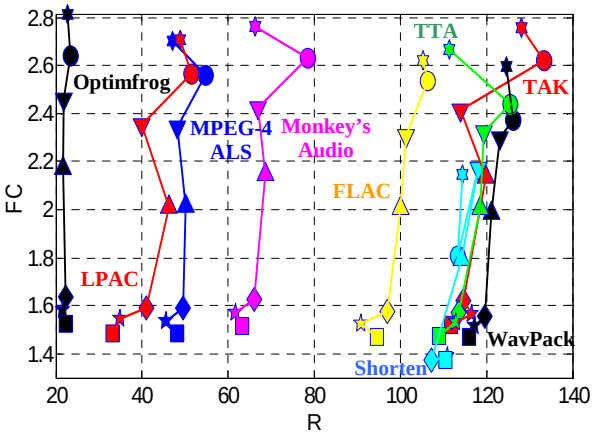

Figura 6: Valores promedio de compresión y rate obtenidos por género musical en cada codificador. Los GM representados son: jazz (✸), tango (●), música clásica (▼), folklore (▲), rock (♦), rock en vivo (★) y pop (■). La información cada codificador se representa como un conjunto de segmentos de un solo color.

4.2 Codificación en otros modos

En los casos donde fue posible, obtuvimos los valores correspondientes a los otros modos. Como era de esperarse, la compresión mejoró en el modo óptimo y empeoró en el modo rápido. Por ejemplo, la compresión de la pieza "Música para los Reales Fuegos Artificiales – Allegro" de Handel posee las variaciones ilustradas en la Tabla 3. En cualquier codec en el modo normal, esta pieza se comprime a la mitad (excepto en Shorten donde *FC* = 1,85). Se observa que MPEG-4 ALS es el que mejor incrementa su rendimiento, hecho sobre el cual nos explayaremos en una sección posterior cuando se haga el balance de resultados de todos los GM.

	Óptimo	Rápido
TAK	4,16	-1,96
WavPack	2	-1,92
TTA	-	-
Shorten	3,57	-
FLAC	0,85	-5,5
Monkey's Audio	4,16	-1,96
MPEG-4 ALS	8,51	-
LPAC	1	-8
Optimfrog	6,52	-2

Tabla 3: Cambios del *FC* como porcentaje del valor correspondiente al modo normal, donde *FC* = 2.

La velocidad de procesamiento puede describirse de forma análoga a la compresión, unificando los resultados para cada codec. Esto es válido ya que el rate es mucho más dependiente del codec empleado que del GM tratado. Los detalles pertinentes a la velocidad se exponen en la Tabla 4 para los 3 modos analizados. Allí sobresale la performance de TAK, ya que es capaz de procesar cualquier archivo en menos del 0,5 % de su tiempo de reproducción. En contraste, la codificación con Optimfrog puede demandar más del doble del tiempo de reproducción del archivo original.

	Óptimo	Normal	Rápido
TAK	36,76	119,77	205,58
WavPack	2,59	121,12	142,52
TTA	-	115,72	-
Shorten	82,92	112,63	-
FLAC	31,59	99,37	140,84
Monkey's Audio	11,11	67,33	84,16
MPEG-4 ALS	1,37	49,09	-
LPAC	23,90	42,21	-
Optimfrog	0,44	22,25	29,43

Tabla 4: Valores promedio de rate en cada codec en las distintas configuraciones adoptadas.

4.3 Decodificación

En todos los casos estudiados, los CSP estudiados preservaron la fidelidad de los datos en un 100 %. Además, se presentaron algunas analogías entre decodificación y codificación. Por un lado, la velocidad de procesamiento depende mucho más del codec utilizado que del género musical tratado. Esa velocidad se ve afectada por el modo con el cual se generaron esos datos, siendo favorable en el caso rápido y desfavorable en el caso óptimo. Entonces, si se contemplan los tiempos de compresión de los archivos y su posterior recuperación al formato original, el modo rápido es doblemente beneficioso y el modo óptimo todo lo contrario.

Como era de esperarse, en la mayoría de los casos la decodificación fue más veloz que la codificación, típicamente entre 4 y 50 veces. Esta asimetría se cumplió en el modo normal y se agudizó en el óptimo, excepto el caso de Monkey's Audio donde se tuvo simetría temporal. Al usar el modo rápido, no se manifestó ese comportamiento asimétrico en todos los codec, aunque siempre se conservó buena velocidad de decodificación como se esperaba.

En la Tabla 5 se ilustran los tiempos de codificación / decodificación correspondientes a la pieza musical mencionada en la sección anterior. Salvo aclaración contraria en la tabla, se trataron los modos óptimo y rápido. Aquí se muestra que la reproducción del archivo es más veloz usando FLAC, hecho que lo hace más apto para reproducir archivos codificados en tiempo real en equipos de bajo costo.

	Óptimo	Rápido
TAK	36 / 88	120 / 113
WavPack	3 / 91	150 / 114
TTA	116 / 111 (único modo)	
Shorten	89 / 116	122 / 146 (modo normal)
FLAC	33 / 196	90 / 211
Monkey's Audio	11 / 11	85 / 66
MPEG-4 ALS	1 / 11	28 / 58 (modo normal)
LPAC	20 / 48	85 / 50
Optimfrog	0,44 / 3	31 / 44

Tabla 5: Valores de rate asociados a la codificación / decodificación en la pieza musical "Música para los Reales Fuegos Artificiales – Allegro" de Handel.

4.4 Resultados en diferentes equipos

La evaluación de los diferentes CSP en las distintas PC no modificó los resultados antes mencionados, excepto por una merma en la velocidad de la siguiente manera:

- En la PC con procesador Pentium IV, la codificación se lentificó entre 4 y 50 veces en el modo óptimo y entre 2 y 4 veces en el modo rápido.
- En el caso de la PC equipada con AMD Athlon, la codificación se lentificó 2 veces en el modo óptimo y de 3 a 7 veces en el modo rápido.

En ambas PC, Optimfrog fue el codec más lento y WavPack, TAK y FLAC los más veloces. Los resultados de decodificación manifestaron que estos tres codec superaron valores de rate de 50.

5. MÉTODO DE FRONTERA

Los análisis y gráficos obtenidos hasta ahora se basan en la observación simultánea de todos los datos {*R*, *FC*} recolectados. Dada su gran cantidad, se dificulta hacer un análisis que permita juzgar la conveniencia de los pares codec-modo (C-M de aquí en adelante) en términos del compromiso compresión-velocidad. Desde el punto de vista del usuario, se plantea un método simple para reducir la representación a una única gráfica para todos los géneros.

La idea es que, si se pueden graficar los resultados para los distintos C-M y los distintos archivos en una misma escala *R*-*FC*, se puede elegir un subconjunto de C-M que mantiene el mismo ordenamiento promedio en todos los archivos. Esto ofrece el máximo beneficio posible para el usuario y además la serie resultante es reducida en puntos y monótonamente decreciente (a mayor rate, menor compresión).

La Figura 7 plantea el caso hipotético de dos codificadores "A" y "B" distintos. En el panel (a), el codec A es más lento pero comprime más que el codec B. En este caso, ambos codec representan opciones perfectamente válidas. El panel (b) ilustra otra situación donde se descarta el codec A por ser más lento y de menos compresión que B. Por esta razón, se suprimen los casos asociados a C-M que no permitan construir una curva monótona decreciente de C-M óptimos en cada valor de rate. Los puntos que "sobreviven" este proceso son considerados óptimos y constituyen la "frontera" del máximo rendimiento.

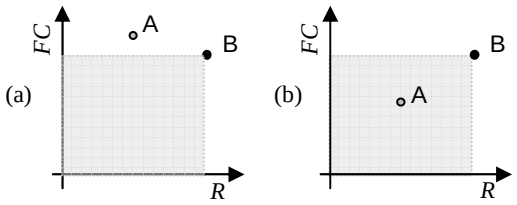

Figura 7: Eliminación de codec: (a) "A" y "B" válidos, (b) "A" se descarta.

A continuación se describe el método recursivo para la selección de los puntos óptimos:

1. Ordenar el conjunto de datos {*R*, *FC*} obtenidos en función de valores crecientes de *R*.
2. Elegir el punto de mayor *FC* como el primero en la frontera.
3. Acotar la tabla a los C-M de mayor rate con respecto al elegido en el paso previo. Si no hay más puntos disponibles, terminar.
4. Determinar el próximo punto en la frontera como el de máximo *FC* en el nuevo conjunto de datos.
5. Volver al paso 3.

Posteriormente, se representa el conjunto de datos obtenidos en una escala gráfica.

Cabe agregar que en este proceso conviene trabajar con valores normalizados de rate y compresión, dado que sólo interesa la comparación mutua entre los datos originales. Dicha normalización se lleva a cabo dividiendo los valores {*R*, *FC*} de entrada entre los correspondientes a Optimfrog-óptimo. (La notación de los C-M se simplifica por sus iniciales, por ejemplo el caso recién mencionado se simboliza Ofr-O.) Éste es un buen parámetro, no solo porque corresponde a la mejor compresión, sino también porque al ser su rate bajo, la dispersión provocada por el retardo del sistema operativo y la interfaz gráfica se minimizan.

A modo de ejemplo, se aplicó el método a los datos de un archivo particular, "Sledgehammer" de Peter Gabriel. La Tabla 6 ilustra los valores normalizados, así como los resultados de la aplicación de los primeros pasos del algoritmo: los puntos originales están ordenados y los seleccionados aparecen sombreados. La representación gráfica de todos ellos y su correspondiente frontera se exponen en la Figura 8.

5.1 Aplicación

El análisis previo fue realizado solamente para un archivo, pero se puede extender a todos los GM, dado

que los resultados *R-FC* son esencialmente uniformes para cada C-M. Es importante recordar lo expuesto en la Figura 6, donde cada codec se distingue por su región operativa.

Para obtener la frontera del conjunto de todos los géneros musicales, se tomó una muestra representativa de archivos de cada GM. Dicha muestra consistió en un par de piezas musicales, procurando que sus resultados fueran cercanos al promedio de sus respectivos géneros. Luego, se determinó la frontera correspondiente a cada archivo analizado, resultando que las configuraciones C-M óptimas aparecen tantas veces como lo dice la Tabla 7.

Orden	C-M	*R* norm.	*FC* norm.
1	Ofr O	1	1,00
2	MP4 O	3	0,98
3	WavPack O	5	0,96
4	Monkey's O	25	0,99
5	LPAC O	47	0,96
6	FLAC O	51	0,95
7	Ofr N	54	0,98
8	Ofr R	61	0,97
9	LPAC N	71	0,95
10	TAK O	83	0,98
11	MP4 N	112	0,95
12	Monkey's N	144	0,97
13	Shorten O	170	0,92
14	LPAC R	175	0,88
15	Monkey's R	183	0,95
16	FLAC N	210	0,95
17	Shorten N	251	0,88
18	TTA	258	0,95
19	TAK N	272	0,97
20	WavPack N	283	0,95
21	WavPack R	303	0,93
22	FLAC R	311	0,87
23	TAK R	406	0,94

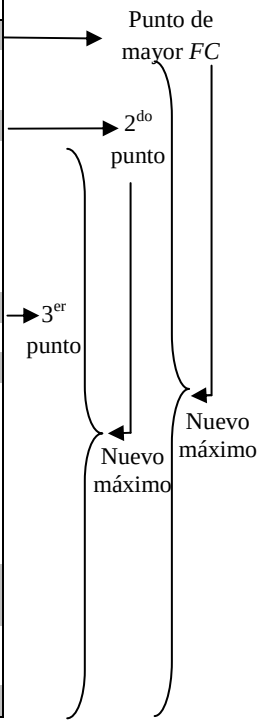

Tabla 6: Datos normalizados y ordenados por rate correspondientes al archivo “Sledgehammer” de Peter Gabriel. Se abrevió la notación de algunos codec, y los modos se identifican por sus iniciales.

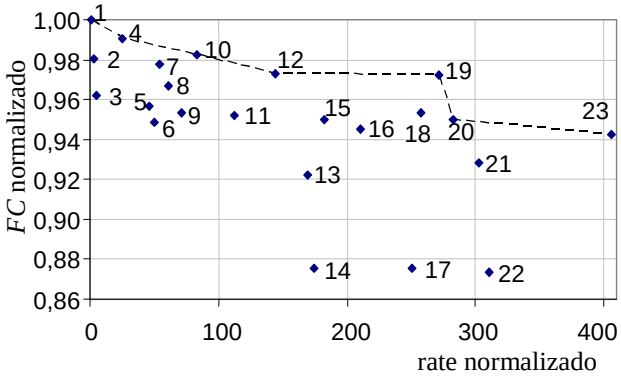

Figura 8: Datos normalizados del archivo “Sledgehammer” de Peter Gabriel y su frontera.

	Óptimo	Normal	Rápido
FLAC	0	0	1
LPAC	0	0	0
Monkey's	13	10	1
MP4	2	0	-
Ofr	14	7	3
Shorten	0	0	-
TAK	11	14	14
TTA	-	5	-
WavPack	0	5	2

Tabla 7: Frecuencia de aparición de C-M en la frontera.

Los resultados recién obtenidos permiten descartar algunas configuraciones C-M, dado que en ningún caso son óptimas. Un ejemplo de esto es Shorten en cualquiera de sus modos. Por el contrario, otras configuraciones como TAK-O son óptimas para todos los archivos. Fundamentándonos en estos resultados, decidimos continuar trabajando sólo con los C-M exitosos más frecuentes y que se sombrean en la Tabla 7. En lugar de eso, se podrían adoptar todos los casos exitosos, o incluso las 23 combinaciones C-M. Sin embargo, el criterio de selección adoptado brinda el mismo resultado, hecho que se demostrará posteriormente.

Los puntos asociados a los C-M seleccionados se grafican en la Figura 9. Es bien notable la formación de nubes compactas por cada C-M, aunque la dispersión aumenta considerablemente con el rate, probablemente debido a una mayor influencia de los retardos del sistema operativo y la interfaz. A raíz de esa dispersión, se agrupan alrededor del mismo valor de rate las nubes de TAK-N, TTA y WavPack-N. Además, TAK-O y Ofr-N comparten valores similares de compresión. A fin de superar estos inconvenientes, se determinó el promedio ponderado asociado a cada C-M (simbolizados con círculos amarillos en la Figura 9). Dicha ponderación tuvo en cuenta el tiempo de reproducción de cada archivo tratado. Esto le da mayor importancia a las piezas más largas, ya que se codifican en tiempos mayores y por ende son más inmunes a la dispersión del rate.

Posteriormente, se aplicó el algoritmo de selección de puntos óptimos a los centroides calculados, resultando una selección final que constituye la “frontera unificada”, ilustrada en la Figura 10.

Cabe aclarar que en el proceso de selección se descartaron TTA y Ofr-N por ser de menor rendimiento que TAK-N y TAK-O respectivamente. Estos casos corresponden a los de menor frecuencia de aparición en la frontera de cada GM, según lo confirma la Tabla 7. En el caso de contemplarse todas las combinaciones C-M posibles, los centroides agregados yacen debajo de la frontera unificada, tal como lo ilustra la Figura 11. Esto demuestra que el resultado del algoritmo de selección no se altera al incluir los casos con menor frecuencia de aparición en la fronte-

ra de cada GM. Por otra parte, no se encontró la cantidad óptima de C-M a descartar, ya que algunos casos tienen baja frecuencia de aparición pero aparecen en la frontera unificada. Un ejemplo de esto es WavPack-N, que mostró ser ligeramente más rápido que TAK-N aunque con una marcada menor compresión.

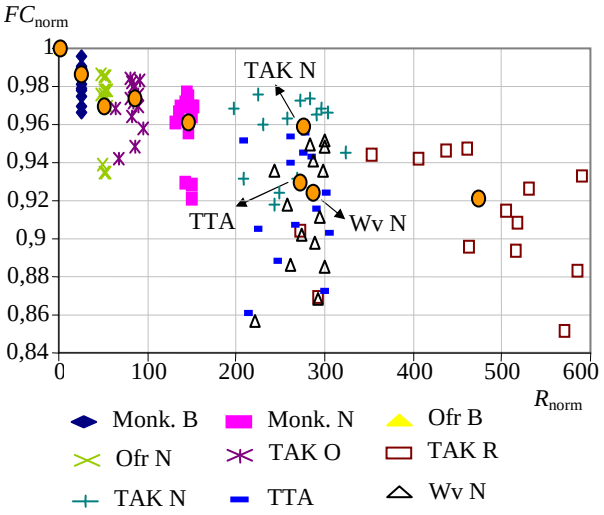

Figura 9: Representación de valores de *R* y *FC* normalizados asociados a los 14 archivos analizados en los C-M seleccionados.

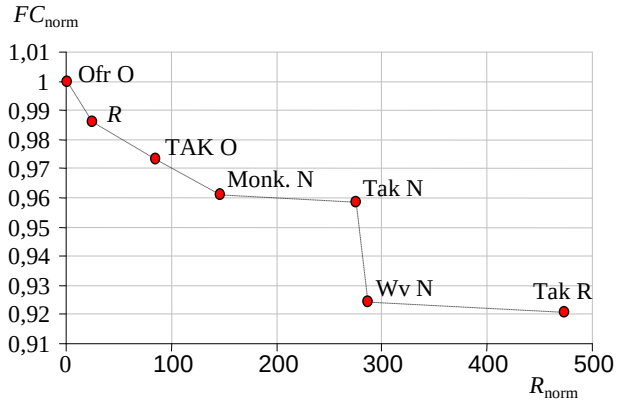

Figura 10: Frontera unificada.

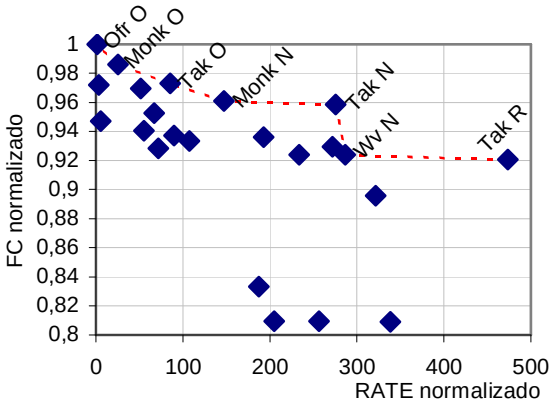

Figura 11: Frontera unificada contemplando todos los C-M disponibles.

Cabe remarcar que la escala relativa desarrollada por nuestro método no se modifica sustancialmente al cambiar de hardware. Por ello, cualquier usuario puede determinar los parámetros {*R*, *FC*} en su PC escalando la frontera normalizada con los parámetros obtenidos para un C-M ensayado en esa PC. Si ese C-M es Ofr-O, dicho escalamiento se traduce en la multiplicación de la escala normalizada por los valores de Ofr-O {$R_{Ofr\text{-}O}$, $FC_{Ofr\text{-}O}$}. En cambio, se puede seleccionar un C-M más veloz pero de reducida dispersión como Monkey's-N. En este caso, además de multiplicar la escala normalizada por los parámetros {$R_{Monkey's\text{-}N}$, $FC_{Monkey's\text{-}N}$}, hay que contemplar la relación entre los valores normalizados de Ofr-O y Monkey's-N.

Se ilustra lo antes mencionado evaluando una pieza musical particular en la computadora portátil mencionada en la sección 4. Se codificó con Monkey's-N "Todo tiempo posible" de Crucis, de 4 minutos de reproducción en 4,164 segundos. Escalando debidamente, se obtuvo la Figura 12. Se observa que la compresión con Ofr-O habría demandado alrededor de 10 minutos, más del doble del tiempo de reproducción. El *FC* también puede ser predicho, aunque en general es más claro adoptar su mejora porcentual. El desvío del valor predicho de tiempo en un caso general depende del codec con el cual se realice el ensayo preeliminar, así como del codec que se quiera predecir, debido a la dispersión de las nubes (desvío estándar sobre el valor medio del rate normalizado) que va desde el 1 % (Monkey's-O) hasta el 22 % (TAK-R).

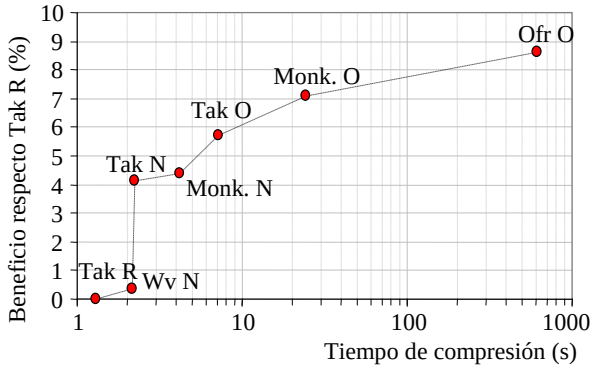

Figura 12: Ejemplo de escalado para una pieza musical de 4 min.

5.2 Comentarios finales

Debido a la reducida cantidad de codec seleccionados en la frontera unificada, estamos tentados a decir que los mejores son TAK, Ofr y Monkey's. Sin embargo, hay que complementar estos resultados con los tiempos de decodificación, la facilidad de manejo de las diferentes interfaces y el soporte en distintos programas. Estos factores tienen mucha influencia a nivel de usuario y pueden inclinar la balanza hacia el lado de otros codec no incluidos en la frontera unificada.

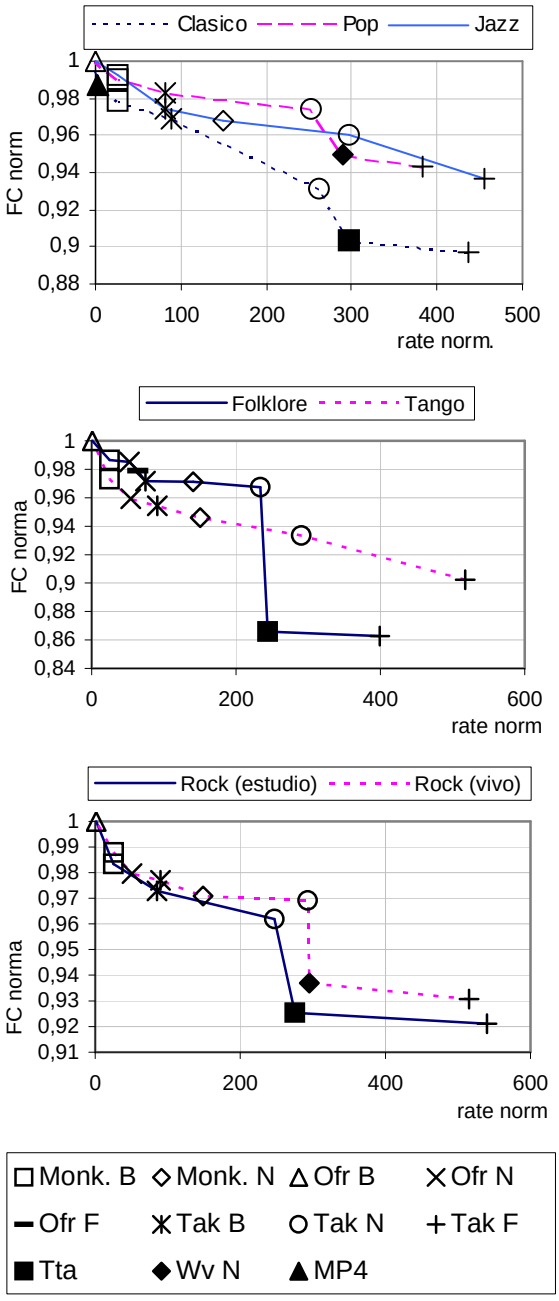

Figura 13: Fronteras individuales por género.

Otro cuestionamiento que puede surgir es si tiene sentido conservar en la gráfica final a Monkey's-N y WavPack-N, ya que al compararlos con TAK-N se sacrifica fuertemente una variable (*FC* ó *R*) a costa de un ligero aumento en la otra. Tenerlos en cuenta o no es decisión del usuario.

Como información extra, se determinó la frontera particularizada para cada GM. La misma se obtuvo aplicando el algoritmo de selección de puntos óptimos, pero muestreando un par de archivos por GM. Cabe resaltar que al tomar menos datos, la aproximación es más gruesa pero igualmente brinda una guía útil al audiófilo interesado en los GM tratados. Los resultados del método se exponen en la Figura 13. El caso MP4-O se presenta como alternativa óptima sólo en la música clásica, ya que obtiene altos *FC*. Mencionamos este caso porque la música clásica es un género ampliamente comprimido con CSP. En general las fronteras evolucionan suavemente, sobre todo si se trata de tango y jazz. En los restantes GM, esto se cumple sólo si el rate normalizado es inferior a 220. Más allá de ese valor se presentan saltos de *FC* que son más pronunciados en las gráficas de rock (en estudio y en vivo) y folklore. Como desconocemos el funcionamiento interno de todos los codec con suficiente nivel de detalle, sólo podemos conjeturar que existe alguna característica de la música que los C-M más veloces no son capaces de modelar.

6. CONCLUSIONES

En este trabajo se analizó mediante pruebas exhaustivas el desempeño de varios CSP popularmente conocidos en la comunidad audiófila. Dado que el principio de funcionamiento en todos los casos es conceptualmente el mismo, se observó que la compresión alcanzada es más dependiente del género musical tratado que del codec utilizado. En este sentido, se mostró que las piezas de jazz y tango pueden reducir su espacio de memoria inicial en un porcentaje mayor a 60. Por el contrario, las piezas de pop y rock se reducen un poco más de 35 %. Se observó también que el ruido contenido en las grabaciones en vivo disminuyó la compresión en aproximadamente un 10 %. Postulamos que estos resultados se atribuyen a las amplitudes presentes en la señal original, ya que las mismas suelen ser mayores en los géneros de rock y pop. En términos de velocidad, se pudo caracterizar cada codec por una zona operativa.

Con respecto a la codificación en modo normal, el óptimo permitió mejorar la compresión menos del 10 % pero a costa de fuertes incrementos en el tiempo de procesamiento. En general, estos tiempos son inferiores al de reproducción por más de un orden de magnitud, excepto en MPEG-4 ALS y en Optimfrog. En este caso particular, la codificación puede demandar más del doble del tiempo de reproducción, haciéndolo impráctico si se prioriza la velocidad de operación.

La codificación en modo rápido sacrifica típicamente menos de 5 % de compresión, pero con el beneficio de reducir el tiempo de procesamiento a un centésimo del tiempo de reproducción.

Los tiempos de decodificación son por lo general menores que los de codificación, excepto en el caso de Monkey's Audio. Otra excepción se dio con la codificación rápida, pero esto no representa inconveniente alguno para el usuario porque se puede decodificar en menos de un centésimo del tiempo de reproducción, excepto en MPEG-4 ALS y en Optimfrog.

Por último, se desarrolló para el usuario común una herramienta gráfica sencilla y de fácil utilización. La misma fue diseñada con un algoritmo de selección

de puntos C-M óptimos en el dominio velocidad-compresión, y sirve de guía para optar por el par C-M más conveniente según las necesidades de compresión y velocidad. Su empleo es extensible a cualquier PC de escritorio, ya que en términos generales la zona operativa de cada codec se conserva. Al cambiar de PC, se modifica la velocidad de cada codec por un factor de escala común. Cabe agregar que la herramienta gráfica se diseñó utilizando un conjunto de codec y piezas, pero se puede generalizar su uso incluyendo otros codificadores y géneros musicales. Por esta razón constituye un método flexible y adaptativo a los requerimientos de consumo del usuario.

7. REFERENCIAS

[1] Norma internacional ISO/IEC-11172-3:1993. "*Information technology - Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s - Part 3: Audio*", 1993.

[2] Fundación Xiph.org. "*Vorbis I specification*". Febrero 2010. http://xiph.org/vorbis/doc/Vorbis_I_spec.pdf

[3] Moriya Reseach Lab. "*Lossless Compression of Audio Signal. MPEG-4 ALS (Audio Lossless Coding) and its Application.*" NTT Communication Science Laboratories. Japón. 2010.

[4] Hans M.C., Schafer R.W. "*Lossless compression of digital audio*". IEEE Signal Processing Magazine. Vol 8, pp. 21–32. Julio 2001.

[5] Robinson T. "*Shorten: Simple lossless and near-lossless waveform compression*". Reporte técnico CUED/F-INFENG/TR.156. Universidad de Cambridge. Reino Unido. Diciembre 1994.

[6] Salomon D. "*Data Compression - The Complete Reference*". Cuarta edición. Springer-Verlag. Reino Unido. 2007.

[7] Spanias A., Painter T., Atti V. "*Audio signal processing and coding*". John Wiley & Sons, Inc. Nueva Jersey, EE.UU. 2007.

[8] Coalson J. "*FLAC - Free lossless audio codec*" Programa libre de código abierto. 2010. http://flac.sourceforge.net/

[9] Rabiner, L.R. Schafer R.W. "*Digital Processing of Speech Signals*", Prentice-Hall Inc., Englewood Cliffs. Nueva Jersey, EE.UU. 1978.

[10] Ashland M.T. "*Monkey's Audio*". Programa de compresión de audio sin pérdidas. http://www.monkeysaudio.com/index.html

[11] Craven P., Gerzon M. , "*Lossless coding for audio discs*", *J. Audio Eng. Soc.* Vol 44, pp. 706–720. 1996.

[12] Craven P., Law M, Stuart J. "*Lossless compression using IIR prediction filters*", Proc. 102nd AES Conv., paper 4415. Munich, Alemania, Marzo, 1997.

[13] Foro DVD. "*DVD Specifications for Read-Only Disc, Version 1.0*". 1999. http://www.dvdforum.com

[14] Liebchen T. "*MPEG-4 ALS. The standard for lossless audio coding*". The Journal of the Acoustical Society of Korea, vol. 28, pp. 618–629. Octubre 2009.

[15] Purat M., Liebchen T., Noll P. "*Lossless transform coding of audio signals*". Proc. 102nd AES Conv., paper no 4414. Munich, Alemania. 1997.

[16] Bruekers A., Oomen A., van der Vleuten R. "*Lossless coding for DVD audio*". *Proc. 101st AES Conv.*, paper no 4358. Los Angeles, California, EE.UU. 1996.

[17] Liebchen T. "*Lossless predictive audio coder (LPAC)*". Archivo ejecutable disponible en http://www.nue.tu-berlin.de/menue/mitarbeiter/tilman_liebchen/lpac_-_lossless_audio_codec_for_windows_and_linux/

[18] Ghido F. "*Ghido's Data Compression Page*". http://www.losslessaudio.org/

[19] Becker T. "Tom's audio compressor (TAK)". Archivo ejecutable en http://thbeck.de/Tak/Tak.html

[20] Djuric A. "*TTA Lossless audio codec - True audio compressor algorithms*". http://en.true-audio.com/TTA_Lossless_Audio_Codec_-_True_Audio_Compressor_Algorithms

[21] Bryant D. "*WavPack. Hybrid Lossless Audio Compression*" http://www.wavpack.com/

[22] Pawlowski P. "*Foobar2000. Freeware audio player for the Windows platform*". http://www.foobar2000.org/

[23] Rodríguez Guerrero J.M. "*Herramienta gráfica para ensayo de codificadores de audio sin pérdidas*". Disponible en http://www.fceia.unr.edu.ar/acustica/