دانشگاه شهید باهنر کرمان

دانشکده فنی و مهندسی

گروه مهندسی کامپیوتر

فشرده سازی صوت

استاد مربوطه :

جناب آقای مهندس یاسر نوروززاده

گردآورندگان :

امیر ناظمی و سهیل سیدی

hhqv@yahoo.com

نام درس :

ذخیره وبازیابی اطلاعات

خرداد ۱۳۸۷

فهرست مطالب

عنوان صفحه

چکیده ۳

مقدمه ٤

مختصری درباره فشرده سازی صوت ٥

صوت چیست؟ ۷

صوت دیجیتالی ۸

الگوریتم SHN (shorten) ۱۱

معرفی چند فرمت صوتی ۱٦

منابع 20

چکیده

فشرده سازی صوت بر مبنای خصوصیات فیزیکی صوت و همچنین توانایی و خصوصیات آدمی در درک یک صوت عمل میکند. به طور کلی الگوریتمهای فشرده سازی عمل ماسک کردن را به روشهای Temporal masking و Frequency masking انجام میدهند . در این مقاله ابتدا توضیحی دز مورد فشرده سازی صوت سپس روشهای sampling یا نمونه برداری جهت دیجیتالی کردن صوت و بررسی یکی از الگوریتمهای فشرده سازی صوت به نام SHN (shorten) که الگوریتمی lossy می باشد ، پرداخته شده است. این الگوریتم از RICE coding جهت کد گزاری و از توابع رگرسیون جهت فشرده سازی صوت استفاده می کند.همچنین در انتهای این مقاله به معرفی چند فرمت صوتی و مختصری توضیح در باره آنها پرداخته شده است.

مقدمه

از زمانی که داده ها بر روی کامپیوتر ها ذخیره گشتند به علت محدود بودن حافظه ها فشرده سازی داده ها هم مورد توجه قرار گرفت . در این راستا صوت ، تصویر ومتن از اولین داده ها ی خام فشرده سازی بودند. و بعد دیجیتالی کردن فیلم ها و پیچیده شدن ساختار فایلها ، فشرده سازی مفهومی کلی تر یافت . فشرده سازی صوت در دهه 90 مورد توجه زیادی قرار گرفت و با سرعت زیادی پیشرفت کرد واین پیشرفت تا کنون ادامه دارد. در این مقاله سعی شده است که به مفاهیم اساسی فشرده سازی صوت به طور مختصر و ابتدایی پرداخته شود . و یکی از قدیمی ترین الگوریتمهای فشرده سازی صوتی به بحث گزارده شود که امیدوارم منبع خوبی برای آشنایی با این شاخه از علم کامپیوتر باشد.

مختصری درباره فشرده سازی صوت

معمولا یک متن فضای زیادی را در یک کامپیوتر اشغال نمی‌کند؛ برای مثال یک کتاب شامل یک میلیون کاراکتر تنها یک مگابایت (Mbyte) فضا اشغال می‌کند زیرا هر کاراکتر از متن تنها ۱ بایت است.

اما در عوض یک پیکسل از یک تصویر بسته به رنگ ۱ بیت تا ۳ بایت فضا نیاز دارد و این تصدیق کننده مثل «یک تصویر به اندازه هزاران کلمه می‌ارزد» است. باید توجه کرد که یک کاراکتر اگر به صورت یک عکس دربیاید، چندین pixel را اشغال خواهد کرد. برای مثال یک عکس که امروزه با دوربین‌های دیجیتال گرفته می‌شود بین Kbyte ۵۱۲ تا Mbyte ۱۲ قبل از فشرده سازی فضا اشغال می‌کند – در دهه‌های۸۰ و ۹۰ میلادی برنامه‌های مولتی‌مدیا رونق گرفت و نیاز ذخیره سازی صدا، تصویر، متن و ... روی کامپیوترها بیش از پیش مورد اهمیت قرار گرفت. همچنین نیاز به upload کردن و download کردن صدا از طریق شبکه به جمع دلایل موجود برای فشرده سازی صوت در یک کامپیوتر پیوست. البته که تجهیزات ذخیره سازی تصویر و فیلم هم‌اکنون از صدا و متن بیشتر است. اما در آن دهه‌ها به خصوص دهه ۹۰ میلادی فشرده سازی صدا و تولید فرمت‌های جدید صوتی بیش از هر زمانی مورد توجه قرار گرفت زیرا تکنولوژی‌های خارجی و داخلی کامپیوترها در مورد تصاویر (فیلم) هنوز به پیشرفت کنونی نرسیده بود.

فشرده سازی صدا دارای ۲ خصوصیت مهم می‌باشد :

۱.می‌تواند lossy باشد.

۲.نیاز به decoding سریع دارد.

به طور کلی کلیه تکنیک‌های فشرده سازی را می‌توان به دو دسته lossy و lossless تقسیم کرد. در فشرده سازی با روش lossless کوچکترین جزئی از داده اولیه ما از بین نمی‌رود و ما داده را به طور کامل داریم و تنها از تکنیک‌هایی برای تبدیل آن‌ها استفاده می‌کنیم مانند الگوریتم فشرده سازی Huffman.

اما در فشرده سازی به روش lossy ما می‌توانیم قسمتی از داده را که از آن استفاده نمی‌کنیم و بودن و نبودن آن تأثیر محسوسی برروی کارایی و کیفیت آن نمی‌گذارد را حذف کنیم؛ به طور مثال در فشرده سازی صدا و تصویر به علت توانایی محدود انسان می‌توان قسمنی از داده را بدون ایجاد تغییر در کیفیت آن داده حذف کرد؛ برای مثال اگر ما عدد ۲۸.۹۹۹۹۹۹۹۹۹ را بخواهیم ذخیره کنیم آن را به روش lossy می‌توان به شکل ۲۹ ذخیره کرد اما در روش lessloss ما آن را به صورت 9[10].28 ذخیره می‌کنیم.

به ندرت پیش می‌آید که ما یک متن را به هنگام decode کردن و decompression کردن بخواهیم بخوانیم اما معمولا یک صدای فشرده شده باید در یک زمان عادی و منطقی decompress شود زیرا کاربر نیاز دارد به هنگام اجرای decoding به آن گوش دهد. به همین علت است که ما ساختار نامتقارنی در codecهای صوتی داریم زیرا encoding می‌تواند وقت زیادی را صرف کند اما decoding معمولا باید سریع انجام پذیرد و به این علت است که متدهای دیکشنری (dictionary) در فشرده سازی و encoding استفاده نمی‌کنند زیرا سرعت دیکد آن‌ها پایین است.

صوت چیست؟

ما از دو جنبه می‌توانیم صوت را تعریف کنیم :

۱.یک حس که توسط گوش انسان شنیده می‌شود و توسط مغز تفسیر می‌شود.

۲.یک حرکت و یک موج است که از حرکت مولکول‌ها و اتم‌ها به وجود می‌آید.

یک دستگاه برای تشخیص و یا تولید صوت باید دارای حساسیت بالایی باشد زیرا یک صدای بلند یک مولکول را به اندازه یک هزارم سانتی متر جابه جا می‌کند (از فاصله معمولی از یک صوت نه فاصله بسیار نزدیک).

همچنین بهترین عایق صدا خلأ می‌باشد زیرا هیچ ذره‌ای برای ارتعاش در آن وجود ندارد و همان‌طور که می‌دانیم صوت یک موج طولی است. در این نوع موج راستای انتشار و راستای ارتعاش هم‌جهت‌اند.

صوت دارای سه خاصیت مهم است که شامل :

۱.سرعت

۲.دامنه

۳.زمان تناوب می‌باشد.

سرعت صوت در دمای اتاق برابر است با ۳۴۳.۸ m/s. بازه فرکانسی شنوایی انسان بین ۲۰ Hz – ۲۲ KHz است و انسان می‌تواند صدایی با فرکانس ۵۰۰ Hz – ۲ KHz تولید کند.

صوت دیجیتالی

همان‌طور که می‌توان تصاویر را به واحدهای عددی مانند پیکسل تبدیل کرد، یک صوت هم برای ذخیره در کامپیوتر باید به صورت عدد دربیاید تا بتوان آن را به صورت ۰ و ۱ در کامپیوتر ذخیره کرد.

وقتی یک صوت توسط میکروفون خوانده می‌شود به یک ولتاژ متغیر تبدیل می‌گردد.

دیجیتالی کردن یک صدا این‌گونه آغاز می‌شود که ولتاژ در نقاط مختلفی از زمان در یک موج صوتی محاسبه شده و وسیله این کار ADC است که مخفف Analog-to-Digital-Convertor می‌باشد. این عمل به این علت صورت می‌گیرد که ویرایش صوت را به صورت digital امکان پذیر می‌کند. همچنین دستگاهی که این اصوات را برای پخش در speaker آماده می‌کند، DAC می‌گویند که مخفف Digital-to-Analog-Convertor است.

چیزی که واضح است اینکه هرچه نمونه برداری ما از یک صوت آنالوگ ریزتر باشد، صدای دیجیتال ما به صوت اصلی نزدیک‌تر خواهد بود.

برای یک نمونه برداری خوب از یک صوت ما با دو مشکل روبه‌رو هستیم.

۱.چه فرکانس نمونه برداری باید انتخاب شود.

همان‌طور که از شکل 1-a پیداست اگر نمونه‌های ما با فرکانس نامناسب اختیار شوند صوت ما به صورت یک خط صاف درمی‌آید. در حالت دوم با افزایش فرکانس نمونه برداری می‌بینیم که شکل فرکانس بعد از decode به شکل نقطه‌چین در b در می‌آید که باز با صوت اصلی ما تفاوت دارد و در قسمت سوم که یک فرکانس نمونه‌برداری مناسب نمایش داده شده است.

راه حل مشکل استفاده از فرکانس بیش از Nyquist frequency است – که دو برابر ماکزیمم فرکانس موجود در یک صوت می‌باشد.

از آن‌جایی که گوش انسان، فرکانس بالاتر از ۲۲ KHz را نمی‌شنود بنابراین ماکزیمم فرکانس نمونه‌برداری برابر است با ۴۴۰۰۰ = ۲*۲۲۰۰۰ که ما فرکانس ۴۴۱۰۰ Hz را به عنوان بیشترین فرکانس درنظر می‌گیریم. بنابراین دلیل این‌که ماکزیمم نرخ نمونه برداری یک صدا ۴۴۱۰۰ Hz این است.

ما می‌توانیم این نرخ نمونه برداری را به عنوان یک فیلتر پایین گذر در نظر بگیریم که فرکانس‌های بالای ۲۲۰۰۰ را حذف می‌کند. همین نرخ نمونه برداری (sample rate) برای سیستم‌های تلفنی معمولا ۸ KHz است یعنی فرکانس‌های بالاتر از ۴۰۰ Hz از بین می‌روند، بنابراین در مکالمات تلفنی معمولا تشخیص حرف P از S معمولا سخت است.

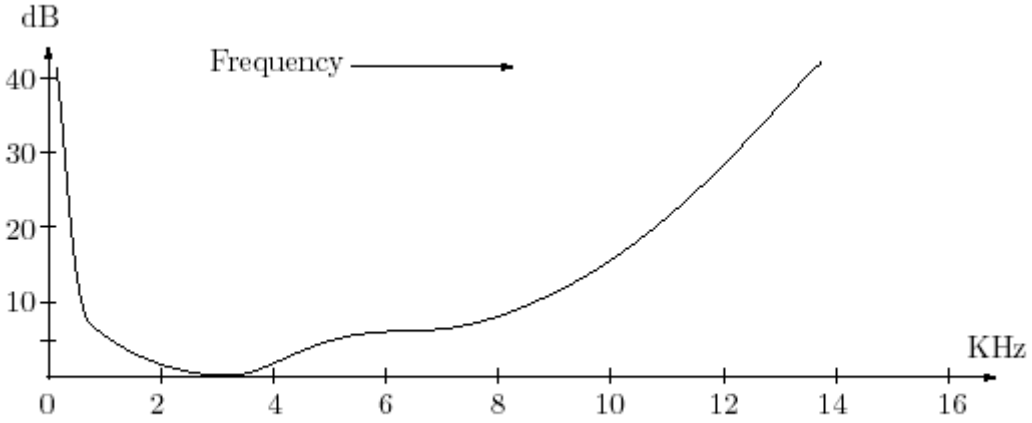
۲- مشکل دوم انتخاب اندازه sample یا نمونه است.

این sample ها معمولا ۸ یا ۱۶ و یا ۳۲ بیت اندازه دارند. فرض کنید که ماکزیمم ولتاژ در یک صدا ۱ ولت باشد. در یک sample با قالب ۸ بیتی کمترین ولتاژ می‌تواند $\frac{1}{2^8} = \frac{1}{256} = 4mv$ باشد. ولتاژهای کمتر از ۴ میلی ولت صفر به حساب می‌آیند وسکوت تلقی می‌شوند. هرچه اندازه این کمتر باشد، فشرده سازی بهتر اما کیفیت پایین‌تر می‌آید.

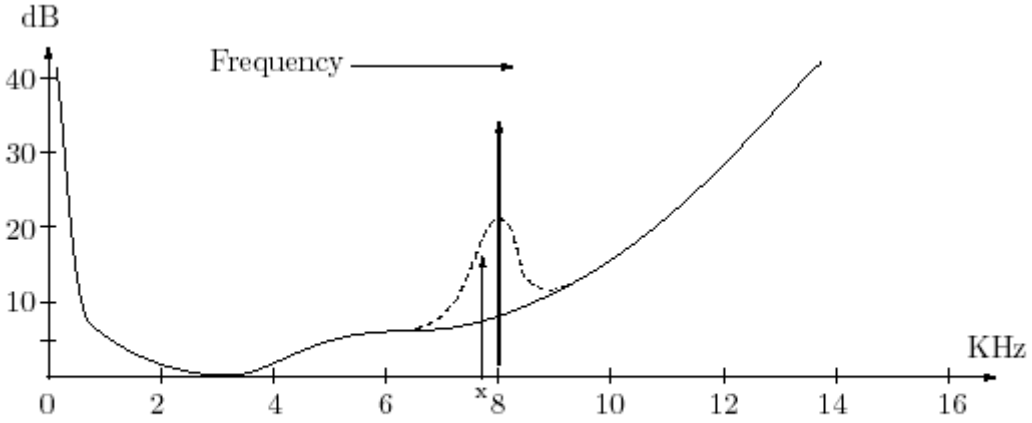
با توجه به آستاه شنوایی انسان که به فرکانس و سکوت محیط اطراف آن بستگی دارد، دو خصوصیت را می‌توان در فشرده سازی صدا به کار برد :

۱. Frequency masking
۲. Temporal masking

Frequency masking هنگامی اتفاق می‌افتد که زمانی که ما یک صدا را می‌شنویم این صدا به وسیله یک صدای دیگر که دارای فرکانس نزدیک به آن است پوشش داده شود. این عمل باعث بالا بردن سطح شنوایی ما شده و فرکانس قبلی شنیده نمی‌شود زیرا این صدا آستانه شنوایی انسان را تغییر و آن را افزایش می‌دهد. به همین علت یک الگوریتم lossy compression باید این سیگنال‌ها را حذف کند.



(a)

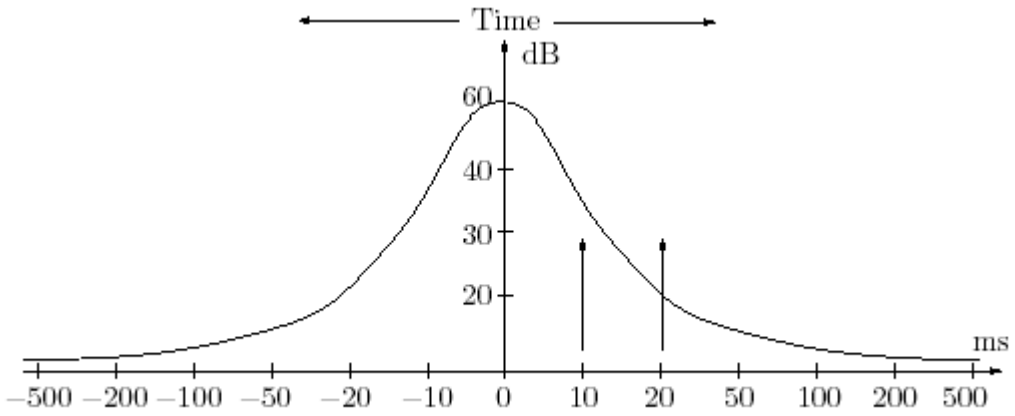


(b)

همان‌طور که در شکل می‌بینیم یک صوت مانند X که به طور نرمال بایستی شنیده شود چون بالای نمودار آستانه شنوایی انسان قرار دارد اما این صوت شنیده نمی‌شود زیرا با ایجاد یک صدای بلند آستانه شنوایی افزایش یافته و حالا این نقطه پایین نمودار (نقطه چین) قرار می‌گیرد و بنابراین باید mask شود.

Temporal masking زمانی اتفاق می‌افتد که یک صوت با فرکانس f توسط یک صوت ضعیف‌تر مانند B با فرکانس نزدیک یا برابر f دنبال شود و یا از آن پیشی افتد.

اگر زمان بین دو صوت کم باشد، صوت B قابل شنیدن نخواهد بود. شکل زیر این موضوع را نشان می‌دهد همانطور که مشاهده می کنید اگر صوت B با فاصله زمانی ۲۰ ms قرار گیرد شنیده خواهد شد.



الگوریتم *SHN (shorten)*

یک فشرده ساز lossless ساده و خاص منظوره برای فایل‌های موجی است. هر فایلی که داده‌های آن به مانند یک موج بالا و پایین بروند، می‌توانند به وسیله این متد فشرده شوند. کارایی فشرده سازی shorten به خوبی Mp3 نیست اما shorten، lossless است.

متد shorten روی فایل‌های با دامنه نوسان پایین و همچنین فرکانس نمونه گیری پایین خیلی خوب جواب می‌دهد. این فرمت روی Unix و MS-DOS پیاده سازی شده است و به طور رایگان در اینترنت موجود است.

یک فایل SHN تقریباً $\frac{1}{2}$ سایز منبع اصلی WAV یا AIFF است برخلاف codec های صوتی Lossy (نظیر MP3 و WMA) SHN قادر است سیگنالهای اصلی را بدون از دست دادن مولفه های فرکانس اش بازسازی کند.

SHN کیفیت صدایی به مراتب بهتر از MP3 ایجاد می کند اما فضای ذخیره سازی بیشتری را نیاز دارد لذا با توجه به زمان Down load یا حجم حافظه مورد نیاز فرمت چندان راحتی نیست.

مند shorten ، ابتدا نمونه های ورودی فایل را به وسیله پارتیشن بندی کردن آنها ، به بلاک‌هایی تبدیل میکند سپس هر نمونه را توسط نمونه‌های قبلی آن پیش بینی می کند ، وبا ذخیره تفاوت پیش‌بینی با نمونه اصلی موجود و کدگذاری این تفاوت (تفریق) کدهای بدست آمده را دوباره ذخیره می کند.

این متد همچنین یک حالت lossy هم دارد به طوری که نمونه‌ها ، قبل از اینکه فشرده شوند ، کوانتیزه می‌شوند. (کوانتیزه کردن به این معنی است که وقتی اعداد ما خیلی بزرگ هستند ما از یک عدد بزرگ استفاده نمی‌کنیم و معمولا می‌توانیم این اعداد را به یک عدد خاص (عدد کوانتیزه) تقسیم کنیم و خارج قسمت را به جای عدد اصلی ذخیره کنیم و در هنگام decoding عدد کوانتیزه را در عدد مربوطه ضرب می‌کنند تا اعداد اولیه به دست آیند.)

باید توجه کرد که ورودی این الگوریتم داده‌هایی با فرمت μ-low است.

μ-low یک استاندارد است که در آمریکای شمالی و ژاپن رواج دارد که ورودی encoder آن ۱۴ بیتی و خروجی آن ۸ بیتی است.یک فایل کامل در block ها قرار می‌گیرد به طوریکه معمولا ۱۲۸ یا ۲۵۶ نمونه در آن جای می‌گیرد.

نمونه‌های داخل بلاک ابتدا به اعداد صحیح تبدیل می‌شوند.

ایده این‌گونه مطرح می‌شود که نمونه‌های داخل یک فایل از یک جنسند و خصوصیات مشترک دارند و می‌توانند پیش‌بینی شوند.

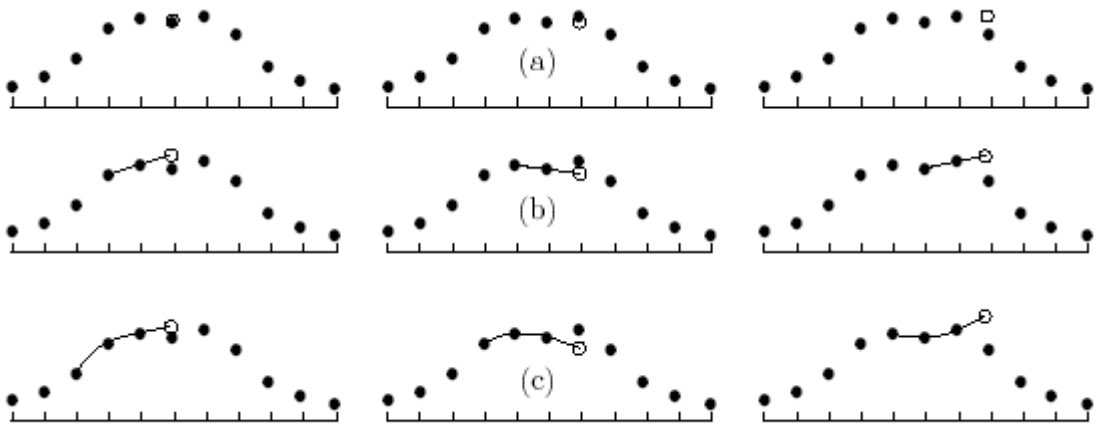
اگر فایل صوتی ما دارای شامل چندین کانال باشد – هر کانال جداگانه در بلاک‌های جداگانه کد شده و مجزا فشرده می‌شوند – هنگامی که یک block ساخته می‌شود، مقادیر نمونه‌ها پیش‌بینی شده و مقادیر تفاوت محاسبه می‌شوند.

یک مقدار پیش‌بینی شده $\hat{S}(t)$ برای نمونه جاری s(t) به وسیله محاسبه p نمونه بدون واسطه قبلی به وسیله یک ترکیب خطی به دست می‌آید.

$$\hat{s}(t) = \sum_{i=1}^{p} a_i s(t-i).$$

اگر پیش‌بینی مورد نظر به خوبی صورت گیرد، $e(t) = s(t) - \hat{S}(t)$ یک مقدار کوچک خواهد شد و e(t) خطای مورد نظر است. ضریب ai بستگی به ثابت بودن و یا متحرک بودن موجی دارد که باید انتخاب شود.

به طور کلی برای یک طبقه n تایی (n – order - predictor) یک چند جمله‌ای درجه (n-1) محاسبه می‌شود که با عبور از n نقطه (s(t-1) تا s(t-n)) s(t) را پیش‌بینی می‌کند.



همان‌طور که در شکل نشان داده شده در شکل a پیش بینی با یک نمونه قبل صورت گرفته و در شکل b پیش‌بینی با ۲ نمونه قبل صورت گرفته و در شکل ۳ پیش‌بینی با ۳ نمونه قبل صورت گرفته است.

$$
\begin{aligned}
&\hat{s}_0(t) = 0,\\
&\hat{s}_1(t) = s(t-1),\\
&\hat{s}_2(t) = 2s(t-1) - s(t-2),\\
&\hat{s}_3(t) = 3s(t-1) - 3s(t-2) + s(t-3).
\end{aligned}
$$

این توابع پیشبینی کننده هم اکنون میتوانند برای محاسبه خطاها استفاده شوند.

$$
\begin{aligned}
e_0(t) &= s(t) - \hat{s}_0(t) = s(t),\\
e_1(t) &= s(t) - \hat{s}_1(t) = s(t) - s(t-1) = e_0(t) - e_0(t-1),\\
e_2(t) &= s(t) - \hat{s}_2(t) = s(t) - 2s(t-1) + s(t-2)\\
&= [s(t) - s(t-1)] - [s(t-1) - s(t-2)] = e_1(t) - e_1(t-1),\\
e_3(t) &= s(t) - \hat{s}_3(t) = s(t) - 3s(t-1) + 3s(t-2) - s(t-3)\\
&= [s(t) - 2s(t-1) + s(t-2)] - [s(t-1) - 2s(t-2) + s(t-3)]\\
&= e_2(t) - e_2(t-1).
\end{aligned}
$$

این محاسبات بازگشتی‌اند اما تنها سه مرحله را درگیر می‌کنند و به طور ریاضی ساده‌اند و هیچ ضربی را نیاز ندارند.

برای بیشترین مقدار فشرده سازی بهتر است هر ۴ تابع تخمین زننده صدا زده شوند و کمترین خطا انتخاب شود.

البته تجربه نشان داده که اولین متد خطا یا همان متد صفر، ۴۸% موفق است؛ در صورتی که حالت سوم تنها آن را به ۵۸% بهبود می‌بخشد.

برای اغلب حالات نیازی به استفاده از توابع تخمین زننده بالا نیست و تخمین دقیق‌تر بایستی با توجه به زمان اجرا در نظر گرفته شوند.

مقدار پیش فزض معمولا $\hat{e_2}(s)$ است.

خطا یا تفاوت معمولا یک مقدار پیوسته درنظر گرفته می‌شود و با مقدار اصلی آن کد جابه‌جا می‌گردد.

یک رابطه قوی بین نمونه‌های اصلی دلالت بر این دارد که اکثر مقادیر خطا معمولا بسیار کوچک و یا حتی صفر هستند و تعدا کمی بزرگند. همچنین این مقادیر دارای علامتند. تجربه نشان داده است که توزیع خطاها خیلی نزدیک به توزیع لاپلاس‌اند.

کدهای انتخاب شده برای shorten توسط الگوریتم ارائه شده توسط Robert Rice که به کدهای Rice معروفند، encode می‌شوند. الگوریتم Rice که بستگی به انتخاب n دارد، به صورت زیر عمل می‌کند:

۱) ابتدا بیت‌های علامت را از بقیه عدد جدا می‌کند. این مهمترین بیت در Rice code است.

۲) n بیت LSB را جدا می‌کند.

۳) بیت‌های باقی‌مانده به صورت یکانی ذخیره می‌شوند و قسمت میانی Rice Code را تشکیل می‌دهند.

ذخیره به صورت یکانی به این صورت است که یک عدد مانند ۳ در دهدهی برای تبدیل به یکانی به شکل ۰۰۰۱ در می‌آید. بنابراین عدد ۵ برابر است با ۰۰۰۰۰۱ و به همین ترتیب عدد ۱۱ در Binary برای تبدیل به صورت unary تبدیل به ۰۰۰۱ می‌شود.

i	Binary	Sign	LSB	No. of Zeros	Code	i	Binary	Sign	LSB	No. of Zeros	Code
0	0	0	00	0	0|1|00						
1	1	0	01	0	0|1|01	−1	1	1	01	0	1|1|01
2	10	0	10	0	0|1|10	−2	10	1	10	0	1|1|10
3	11	0	11	0	0|1|11	−3	11	1	11	0	1|1|11
4	100	0	00	1	0|01|00	−4	100	1	00	1	1|01|00
5	101	0	01	1	0|01|01	−5	101	1	01	1	1|01|01
6	110	0	10	1	0|01|10	−6	110	1	10	1	1|01|10
7	111	0	11	1	0|01|11	−7	111	1	11	1	1|01|11
8	1000	0	00	2	0|001|00	−8	1000	1	00	2	1|001|00
11	1011	0	11	2	0|001|11	−11	1011	1	11	2	1|001|11
12	1100	0	00	3	0|0001|00	−12	1100	1	00	3	1|0001|00
15	1111	0	11	3	0|0001|11	−15	1111	1	11	3	1|0001|11

این کد تقریبا شبیه کد Huffman است که یک عدد صحیح از بیت را به هر مقدار مختلف اختصاص می‌دهد.

option lossy این الگوریتم این است که قبل از فشرده سازی اعداد ما کوانتیزه شوند اما این option دارای سرعت کم است و از آن استفاده نمی‌شود.

.

به طور کلی الگوریتم shorten ابتدا به بلاک بندی نمونه‌ها می‌پردازد. پس با استفاده از رگرسیون و تشکیل یک تابع رگرسیون نمونه‌های بعدی را توسط نمونه‌های قبل پیش‌بینی می‌کند و اختلاف بین نمونه‌های کم‌شده را به عنوان داده‌های جدید که مقداری به مراتب کمتر از داده‌های اصلی دارند ذخیره می‌کند سپس الگوریتم Rice را روی داده‌های ذخیره شده پیاده سازی می‌کند.

این مراحل در اکثر الگوریتم‌های فشرده سازی صوت دنبال می‌شوند به طوری که برای مثال در الگوریتم Mp3 ابتدا با توجه به نمودار آستانه شنوایی انسان داده‌ها به صورت lossy کاهش می‌یابند سپس با استفاده از ماسک‌های صوتی فرکانس‌هایی که شنیده نمی‌شوند حذف می‌گردند و بعد از نمونه برداری، نمونه‌ها توسط الگوریتم هافمن دیکد می‌شوند.

تفاوت عمده الگوریتم Mp3 با الگوریتم shorten این است که الگوریتم Mp3، lossy است اما الگوریتم shorten، lossless است

معرفی چند فرمت صوتی

AAC(Adavnced Audio Coding)

يك فرمت فشرده سازي براي سيگنالهای صوتي ديجيتالي است.

AAC فرمتي جديد است كه از ديدگاه كيفيت صدا و بازدهي data (data efficiency) حتي نسبت به MP3 هم ارجح است.

فرم فشرده سازي AAC كاملا جديد است و براساس برخي تستهاي شنيداري، فايل AAC در bitrate پايين تر (96 kbps) كد مي شود كه كيفيت آن بهتر از MP3 نيز هست(MP3 با 128 KBPS كد مي كند)

نسخه فصل AAC به عنوان بخشي از استاندارد mpeg4 منتشر شده است.

AAC عموما در سيستم هاي Apple براي فايلهاي صوتي بكار مي رود.

AIFF (Audio Interchange File Format)

يك فرمت صوتي براي سيستم عامل ميكنتاش است كه اغلب براي ذخيره سازي غيرفشرده صدا با كيفيت (CD-quality) بكار مي رود. AIFF شبيه به فايلهاي Wav در سيستم عامل ويندوز است.

ATRAC

بوسيله سوني در دهه ۹۰ ابداع شد. ATRAC يك Codec صوتي است كه صداي near CD-quality توليد مي كند فرمت mini Disc از ATRAC استفاده مي كند.

ATRAC 3

يك نسخه جديدتر است كه فايلهاي كوچكتري ايجاد مي كند.

Bitrate

در فشرده سازي صوتي، مقدار متوسط data كه لازم است تا يك ثانيه موسيقي را ضبط كند bitrate است كه برحسب كيلو بيت بر ثانيه بيان مي شود.

برخي از codec ها نظير AAC,WMA,MP3 مي توانند به bitrate هاي مختلف فايلهايي را كد كنند.

MP3 (Mpeg1 , Audio Layer 3)

Codec معروفی برای ذخیره و انتقال موسیقی است.

فرکانسهایی که اساساً غیرقابل شنیدنی هستند را حذف می کند اما کیفیت صدا در near-CD quelit ایجاد می کند **و فایل** آنهم دارای سایزهای حدود $\frac{1}{10}$ تا $\frac{1}{12}$ فایل WAV معادل به آن(غیرفشرده) می باشد**.**

در ایجاد فایلMP3 امکان فشرده سازیهای مختلفی است که براساس آن سایر فایل و کیفیت صدا عوض می شود.

MP3 pro

این نیز بخشی از MPEG است که نسخه پیشرفته ای از MP3 می باشد و می تواند جزئیات فرکانس بالای بیشتر را ذخیره کند.

در واقع بخش فرکانس بالای سیگنال صوتی به وسیله یک فرآیند فشرده سازی بنام Spectral Band Replicats on SBR ایجاد می شود اما بقیه قسمتهای سیگنال با فرآیند معمولی MP3 می شود.

لذا نرم افزارهای MP3 player قدیمی قادرند فایلهای کد شده در mp3 pro را پخش کنند هر چند قسمت SBR فایل را پخش نمی کنند.

اگر فایل MP 3 pro با MP3 pro player سازگار با خودش پخش می شود صدایی با کیفیت عالی و با bitrate پائین وجود خواهد داشت.

SDII (Sound designer II)

یک فرمت صوتی برای سیستم مکنیتاش (Macintosh) است و برای pro-quality sound edit بکار میرود قابلیت ضبط CD-quality Audio بصورت غیرفشرده را هم داراست.

SDMI (Secure Digital music Initiative)

SDMI یک فرمت فایل نیست. بلکه یک سیستم حفاظت Copyright برای فایلهای موسیقی دیجیتالی است.

سخت افزارها و نرم افزارهایی که با SDMI سازگار باشند شما را قادر می سازند نه تنها فایلهای MP3 مجانی اینترنتی را دریافت و پخش کنید بلکه موریکهای COPY –Protected را هم دریافت نمائید.

WAV

یک فرمت صوتی استاندارد برای ویندوز است و برای ذخیره صدای غیرفشرده با کیفیت بالا بکار می رود. فایل WAV می تواند سیشگنالهای صوتی CD-quality (44.1 KHZ-16 bit) را شامل شود.

لذا فایلهای WAV نیاز به مقادیر زیادی حافظه دارند که حدود ۱۰ مگابایت برای هر دقیقه موزیک لازم است.

WMA (windows media Audio)

فرمت WMA به وسیله میکروسافت ارائه شد و برای دستگاهها پخش صوت دیجیتال پورتابل که زمان کل پخش براساس حافظه محدودیت دارد بکار می رود استفاده از WMA در win xp پیش بینی می شود کاربران می توانند فایل WMA برای خود تولید کنند.

منابع

1- Data Compression The Complete Reference Fourth Edition by David Salomon With Contributions by Giovanni Motta and David Bryant

2- Data Compression book by Mark Nelson and Jean-Loup Gailly

3- http://en.wikipedia.org/

4-http://www.digitalpreservation.gov/formats/fdd/sound_fdd.shtml