REAL-TIME ADDITIVE SYNTHESIS OF SOUND BY TAKING ADVANTAGE OF PSYCHOACOUSTICS

Mathieu Lagrange and Sylvain Marchand

SCRIME - LaBRI, Université Bordeaux 1
351, cours de la Libération, F-33405 Talence cedex, France
[lagrange|sm]@labri.u-bordeaux.fr

ABSTRACT

In this paper we present an original technique designed in order to speed up additive synthesis. This technique consists in taking into account psychoacoustic phenomena (thresholds of hearing and masking) in order to ignore the inaudible partials during the synthesis process, thus saving a lot of computation time. Our algorithm relies on a specific data structure called "skip list" and has proven to be very efficient in practice. As a consequence, we are now able to synthesize an impressive number of spectral sounds in real time, without overloading the processor.

1. INTRODUCTION

Spectral sound models provide general representations for sound well-suited for intuitive and expressive musical transformations [1, 2]. However, most of them are based on additive synthesis and thus usually require the computation of a large number of sinusoidal oscillators, since they are dealing with sounds made of a large number of partials or harmonics.

We have shown in [3, 4] that a very efficient synthesis algorithm – based on a recursive description of the sine function – can reproduce sound in real time from the additive parameters, with a nearly optimal number of operations per partial. In this article, we propose to consider psychoacoustic phenomena such as masking in order to reduce the number of partials to be synthesized.

After a brief introduction to additive synthesis and psychoacoustics in, respectively, Sections 2 and 3, we present in Section 4 the algorithm which allows us to reduce on the fly the number of partials to be synthesized as well as some results obtained with our software implementation of this algorithm in Section 5.

2. ADDITIVE SYNTHESIS

Additive synthesis (see [5]) is the original spectrum modeling technique. It is rooted in Fourier's theorem, which states that any periodic function can be modeled as a sum of sinusoids at various amplitudes and harmonic frequencies.

2.1. Sinusoidal Modeling

For pseudo-periodic sounds, these amplitudes and frequencies continuously evolve slowly with time, controlling a set of pseudo-sinusoidal oscillators commonly called *partials*. The audio signal a can be calculated from the additive parameters using Equations 1 and 2, where n is the number of partials and the functions f_p, a_p, and ϕ_p are the instantaneous frequency, amplitude, and phase

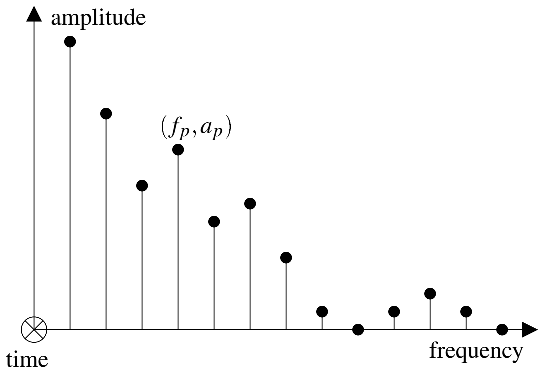

Figure 1: *Partials of an harmonic sound.*

of the p-th partial, respectively. The n pairs (f_p, a_p) are the parameters of the additive model and represent at time t points in the frequency-amplitude plane, as shown in Figure 1. This representation is used in many analysis / synthesis programs such as SMS [1] or *InSpect* [3, 6].

$$a(t) = \sum_{p=1}^{n} a_p(t) \cos(\phi_p(t)) \tag{1}$$

$$\phi_p(t) = \phi_p(0) + 2\pi \int_0^t f_p(u)\, du \tag{2}$$

2.2. Real-Time Synthesis

The real-time synthesis has been implemented in our *ReSpect* software tool [3, 6]. We are indeed able to synthesize hundreds – if not thousands – of sinusoidal oscillators in real time. As the number of oscillators grows, the volume of many partials is very low and the average distance in frequency between two consecutive partials is getting smaller. Psychoacoustic considerations (see below) show that many partials are then inaudible, and could safely be ignored at the synthesis stage in order to save processor time.

An harmonic sound with a $F = 50$ Hz fundamental frequency requires a maximal number of 440 partials (since 22 kHz is the highest audible frequency). We believe that a number of oscillators not much larger than that is enough for nearly all possible sounds, even polyphonic ones. The reason for this is that as the number of oscillators grows, the smallest distance – in frequency – between two oscillators is going to get smaller. When this distance is sufficiently small, the masking phenomenon occurs (see 3). Since the partials under the masking threshold cannot be heard, we can safely omit them at the synthesis stage.

3. PSYCHOACOUSTIC PHENOMENA

The fast fluctuations – from tens to thousands per second – of the air pressure at the level of the ears generate an auditory sensation. The word "sound" stands for both the physical vibration and the sensation this vibration produces. This section explains the basic knowledge in psychoacoustics [7] required to fully understand this article. Further information can be found in [8, 9].

If, at time t, we can measure the instantaneous amplitude and frequency of a sound, the corresponding perceptive parameters are loudness and pitch, respectively. The Fechner law applies to every sensory organ and claims that the sensation is proportional to the logarithm of the excitation. As a consequence, human beings perceive these parameters on logarithmic scales.

3.1. Perceptive Scales

The dB (decibel) scale is commonly used to represent the volume. The relation between the volume in dB and the linear amplitude is given by Equation 3. If we consider that the maximal amplitude (1.0 in the linear scale) should correspond to a volume of 120 dB in order to match the standard dB SPL (Sound Pressure Level) scale, then we set $A_{0\text{dB}} = 10^{-6}$ corresponding to an acoustic pressure of $P_{0\text{dB}} = 2 \cdot 10^{-5}$ Pa (Pascals). Anyway, amplitude and pressure being proportional, it is just a matter of translation of the volume origin (0 dB) in the logarithmic scale.

$$V(a) = 20\log_{10}\left(\frac{a}{A_{0\text{dB}}}\right) \tag{3}$$

A very convenient scale for representing frequencies is the Bark scale (after Barkhausen), which is very close to our perception [9]. Equation 4 allows us to go from the Hertz scale to the Bark scale.

$$B(f) = \begin{cases} f/100 & \text{if} \quad f \leq 500 \\ 9 + 4\log_2(f/1000) & \text{if} \quad f > 500 \end{cases} \tag{4}$$

3.2. Threshold of Hearing

Human beings can hear frequencies in the range of 20 Hz to 22 kHz approximatively, but the sensibility threshold in amplitude $\mathcal{S}_a$ is a function of frequency. Equation 5 provides us with a good approximation for this threshold. Partials with volumes below this threshold will not be heard, and thus can safely be ignored at the synthesis stage.

$$\mathcal{S}_a(f) = \begin{cases} 3.64(f/1000)^{-0.8} - 6.5e^{-0.6(f/1000-3.3)^2} \\ \qquad +10^{-3}(f/1000)^4 \end{cases} \tag{5}$$

3.3. Frequency Masking

Physically, the addition of two signals of the same amplitude is ruled by a nonlinear addition law and gives a maximum of +6 dB. However, from a perceptive point of view, there is a modification of the perception threshold for a sound m (masked sound) when it is played together with a louder sound M (masking sound). This

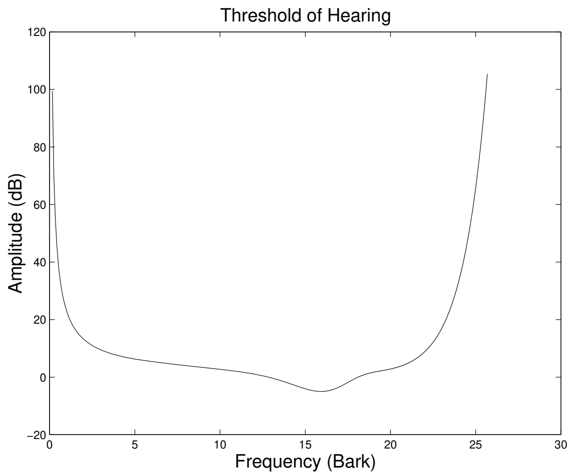

Figure 2: *Threshold of hearing $\mathcal{S}_a$.*

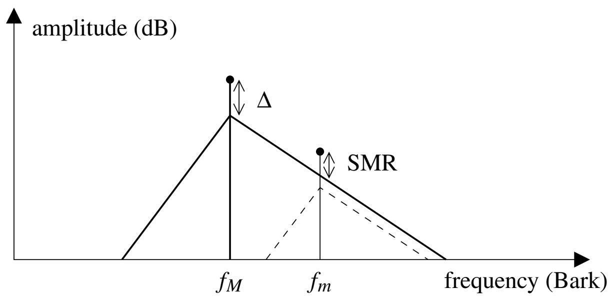

Figure 3: *Masking of a sinusoid of frequency f_m by another sinusoid of frequency f_M. The masking effect is maximal when f_m and f_M are close. As a first approximation we can consider that the masking threshold is close to a triangle in the Bark-dB scale, although it is not exactly the case in practice, especially for the top of the triangle.*

phenomenon is known as the frequency masking. Consider the case of M and m being two sinusoids of frequencies f_M and f_m, and amplitudes a_M and a_m, respectively. Assume that $a_M > a_m$. If f_m is close to f_M, the sound m is masked by the sound M and becomes inaudible. This phenomenon can be used to lower the number of sinusoids to be computed during the additive synthesis process by filtering out the masked (inaudible) partials. Garcia and Pampin use in [10] this masking phenomenon to reduce the number of sinusoids of a sound in the additive model. We propose to use a similar technique, but this time not for compression purposes, but for synthesis efficiency only. The masking phenomenon is also used in the MPEG-II Layer 3 audio compression [11, 12]. We do not want to compress the spectral sounds, because we need all the partials – even the inaudible ones – for musical transformations. But masked partials can be removed at the synthesis stage, since they will not be heard. As a first approximation we can consider that the masking threshold is close to a triangle in the Bark-dB scale. Garcia and Pampin [10] use a simple masking model to evaluate the signal-to-mask ratio (SMR) of each partial. This model consists of:

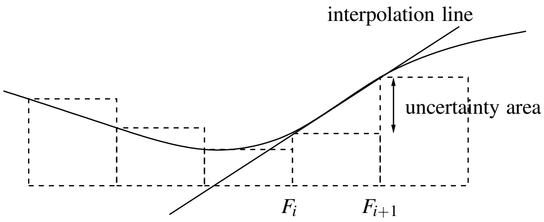

Figure 4: *Linear interpolation applied to the discrete version of the threshold of hearing.*

- The difference Δ between the level of the masker and the masking threshold (-10 dB);
- The masking curve towards lower frequencies (left slope: 27 dB/Bark);
- The masking curve towards higher frequencies (right slope: -15 dB/Bark).

Another interesting phenomenon is temporal masking. There are two kinds of temporal masking. The post-masking occurs when the masking sound disappears. In fact, its effect persists in time during some milliseconds. As a consequence, even if the masking sound is not present the masking effect is still present, although it decreases with time. Perhaps more surprisingly, pre-masking also exists. More precisely, the masking effect is active a few milliseconds before the masking sound really appears. However this phenomenon is less significant.

4. REAL-TIME SYNTHESIS ALGORITHM

Before the classic additive synthesis algorithm itself, another algorithm can be added in order to reduce the number of partials to be synthesized by filtering out inaudible partials.

This algorithm decides whether a partial can or cannot be detected by a listener in a noiseless environment. More precisely, for each partial we first compare its amplitude a_p to the threshold of hearing $\mathcal{S}_a$ at the corresponding frequency f_p. If it turns out that this partial could be heard, then we check if it is not masked by some other stronger partial.

4.1. Threshold of Hearing

Since the expression of the threshold of hearing given in Equation 5 is rather complicated to compute, we choose to deal with a discrete version of this threshold. More precisely, we first sample the threshold of hearing, thus precomputing the value of $\mathcal{S}_a$ for a large number of discrete frequencies in the Bark scale in order to achieve a sufficient precision. Then, for each partial p, if f_p is between the frequencies F_i and F_{i+1} then:

- If $a_p < \min(\mathcal{S}_a(F_i), \mathcal{S}_a(F_{i+1}))$, then p will not be heard;
- If $a_p > \max(\mathcal{S}_a(F_i), \mathcal{S}_a(F_{i+1}))$, then p could be heard;
- If $\mathcal{S}_a(F_i) < a_p < \mathcal{S}_a(F_{i+1})$, then we have to decide by linear interpolation.

Indeed, in the last case p is in an uncertainty area due to the sampling of the threshold. To decide whether the partial could be heard or not, we approximate $\mathcal{S}_a$ between the sampling points F_i and F_{i+1} by a line segment (see Figure 4) and we simply compare a_p with the value of the segment at ordinate f_p.

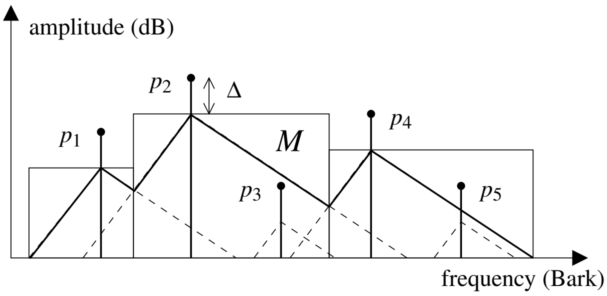

Figure 5: *Five partials and the associated mask M (bold polyline). p_1, p_2, and p_4 are masking partials and contribute to M. The frequency areas of their contributions are represented by rectangles. p_5 is neither masking nor masked, and p_3 is masked (by p_2).*

4.2. Frequency Masking Overview

After that, our algorithm decides whether a partial is masked or not, while computing the global mask M incrementally. First, the partials are sorted by decreasing amplitudes. Then, for each partial p:

- If $M(f_p) + \Delta < V(a_p)$, then p is a **masking** partial and M must be updated with its contribution;
- If $M(f_p) < V(a_p) \leq M(f_p) + \Delta$, then p is neither masking nor masked;
- If $V(a_p) \leq M(f_p)$, then p is simply **masked**.

We use a list in order to store the contributions of the masking partials to the global mask M. Since this list is ordered by increasing frequencies, only the left and right neighbors are to be considered when inserting the contribution of a new masking partial. The new partial cannot mask them, since its amplitude is lower than theirs (remember that the partials have been previously sorted by decreasing amplitudes), but it can shorten the frequency area where they contribute to M.

The contributions of the masking partials to the global mask M are stored in a double-linked list, sorted by increasing frequencies. In order to decide if a new partial p is masking, neither masking nor masked, or simply masked, we need to search the list for the two contributions surrounding its frequency f_p. If it turns out that p is a masking partial, then its contribution must be inserted into the list at the right position – in order to maintain the order of the list – and the contributions of its neighbors are to be updated.

4.3. Using a Skip List

As a consequence, we need a data structure ordered in frequency, with the notion of left and right neighbors, and where searching and inserting is as fast as possible. Thus, we choose a tree-like structure, the skip list, introduced by Pugh in [13] as an alternative to balanced trees. Both data structures show a logarithmic complexity for searching.

Let us explain the mechanism of the skip list on the example of Figure 6. To find out the node with a key value equal to 6 in this skip list, we begin at the top of the head pointer array (leftmost node). The node pointed is `NIL` (rightmost node), which contains a key value greater than those of all the nodes in the list. So, we have to go down in the array of pointers. The pointed node has now

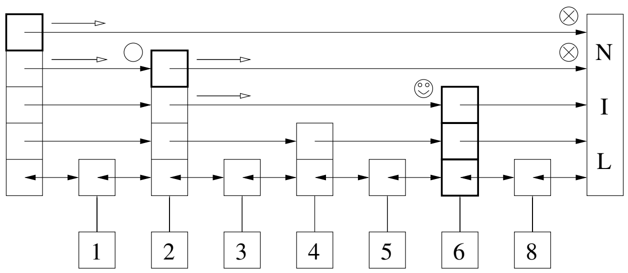

Figure 6: *Example of a skip list. Bold rectangles indicate addresses that would be updated during the insertion of value 7 in the skip list.*

the key value 2, which is lower than 6. We can safely skip to node 2 (without going through node 1) and start again the same process on this new node. Doing so, we jump directly to node 6. With this mechanism, we can search very efficiently a list for a value. At the end of a search, if the current node does not contain the desired key value, then the searched value was simply not present in the list.

Concerning insertion, balanced trees algorithms explicitly balance the data structure during insertions – which is time consuming – whereas skip lists algorithms balance it probabilistically, by choosing the size of the array of pointers of the new node randomly. Thus, insertion is a simple process, consisting mainly in searching for the first node with a key value greater than the one of the new node, followed by pointer updates. The cost of pointer updates is constant, that is why insertion and searching are both $O(\log(n))$ in terms of complexity – n being the number of elements in the list – which is optimal.

4.4. Global Mask Generation Algorithm

We first allocate enough memory to be able to store at least the contributions of all the partials and initialize the head and `NIL` nodes. Then, for each partial p which has an amplitude a_p greater than the threshold of hearing (see 3.2), we try to insert its contribution into the global mask M.

We first search M for the lowest node with a frequency greater than f_p – the right neighbor. During this search, we store all the addresses of the nodes whose frequency is greater than f_p. We obtain the left neighbor with the backward link of the right neighbor. With these local information (left and right neighbors), we check whether the contribution of the partial p has to be inserted or not (see 4.2).

In case of an insertion, we choose randomly the size of the array of pointers of the new node – lower than the size of the head array though. We copy in this array all the addresses previously stored. Finally, we update the pointers structures with the address of the new element. At the end of our algorithm, we know which partials has to be synthesized and we can send them to our efficient algorithm for additive synthesis.

4.5. Towards an Adaptive Masking Algorithm

If the masking phenomenon is strong, the number of masked partials n_m is high and the skip list remains very short. The global mask computation is fast and efficient – since a lot of partials will not be synthesized in the end. On the contrary, if n_m is low, the skip list is bigger and the insertion of each new contribution is slower. A lot of time is lost in the generation of the global mask M whereas only a few partials will not be synthesized in the end.

The challenge is then to be able to tune the duration of the generation of the global mask, depending on the evolution of some efficiency criterion without knowing the exact number of masked partials. For example, in the first case above (strong masking), this criterion will be high and low in the second case (weak masking).

Further research will include the capability to approximate – as precisely as desired – the global mask and to determine the best tuning strategy.

5. RESULTS

The amplitude sort and frequency masking algorithms in this software architecture (see Figure 7) have both a complexity in $O(n \cdot \log(n))$, where n is the number of partials.

Yet, the additive synthesis itself is only $O(n)$ in terms of complexity, its duration being proportional to the number of partials. Trying to speed up this algorithm by the mean of a more complex one could have been an utopia.

5.1. Monophonic Sources

In the case of a single harmonic source showing poor masking among partials, our algorithm may seem of little interest, because it takes much time for basically removing a very few partials. It would have been faster to forget about our algorithm and just compute the inaudible partials anyway...

Hopefully, the global mask is very slow time-varying and M has not to be rebuilt at every synthesis step. Currently, the global mask is recomputed only every 16 synthesis steps, each synthesis step occurring every 64 samples at 44100 Hz. This value is based on psychoacoustic experiments. This allow our algorithm to be still valuable – or at least acceptable – when the masking phenomenon is not strong enough. The error – depreciated partials state – is very low ($<1\%$). Further research will include error reduction using a global mask interpolation in time.

5.2. Polyphonic Sounds

The masking phenomenon is stronger when synthesizing several sources simultaneously. In the example illustrated in Figure 8, 50% of the partials are removed, with a very small computation time overhead. As a consequence, the synthesis time is divided by almost 2. Practical experiments confirm that our algorithm becomes even more valuable as the number of partials increases. This is even more encouraging that the current implementation of our algorithm – used for these measurements – is still an early version, far from being fully optimized.

6. CONCLUSION AND FUTURE WORK

We are actually implementing this masking algorithm on the top of our *ReSpect* real-time synthesis software tool. For 3 harmonic sources played simultaneously – a voice, a saxophone, and a guitar – 50% of the partials were removed. As the number of sources increases, the algorithm becomes more and more valuable: Most partials are removed, and *ReSpect* manages to synthesize complex sounds in real time where it had failed before.

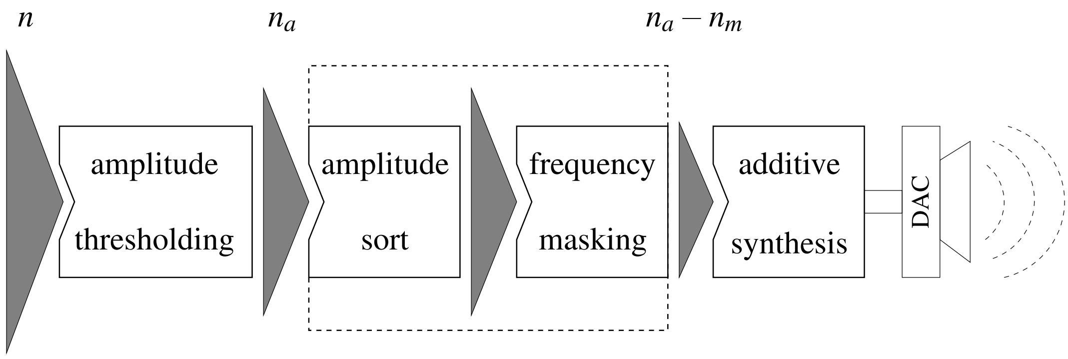

Figure 7: *Software architecture overview. Among the n original partials, only n_a are above the threshold of hearing. These partials are sorted by decreasing amplitudes, then the n_m masked partials are filtered out by taking advantage of the efficiency of the skip-list data structure. Eventually, only $n_a - n_m$ partials are effectively sent to a classic algorithm for additive synthesis.*

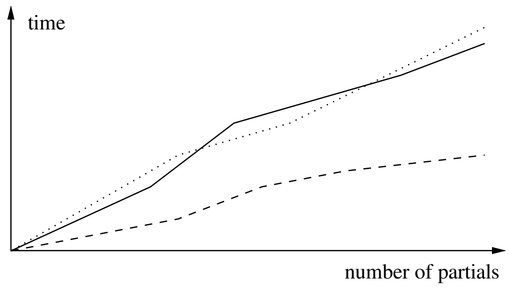

Figure 8: *Polyphonic sound synthesis. The durations of the complete synthesis (solid line), synthesis with systematic global mask computation (dotted line), and synthesis with spaced global mask computation (dashed line) are displayed in the case of a polyphonic sound consisting of several harmonic sources (singing voice, saxophone, and guitar) played simultaneously.*

Further research includes exact determination of other psychoacoustic phenomena such as temporal masking and other related ones that may help us to decrease the number of oscillators needed, and thereby increasing performance.

As a consequence, in a few years most people will have a PC with sufficient power on their desks to generate any sound in real time based on our method.

7. REFERENCES

[1] Xavier Serra, *Musical Signal Processing*, chapter Musical Sound Modeling with Sinusoids plus Noise, pp. 91–122, Studies on New Music Research. Swets & Zeitlinger, Lisse, the Netherlands, 1997.

[2] Sylvain Marchand, *Sound Models for Computer Music (analysis, transformation, synthesis)*, Ph.D. thesis, University of Bordeaux 1, LaBRI, December 2000.

[3] Sylvain Marchand and Robert Strandh, "InSpect and ReSpect: Spectral Modeling, Analysis and Real-Time Synthesis Software Tools for Researchers and Composers," in *Proceedings of the International Computer Music Conference (ICMC)*, Beijing, China, October 1999, International Computer Music Association (ICMA), pp. 341–344.

[4] Robert Strandh and Sylvain Marchand, "Real-Time Generation of Sound from Parameters of Additive Synthesis," in *Proceedings of the Journées d'Informatique Musicale (JIM)*, Paris, May 1999, CEMAMu, pp. 83–88.

[5] James A. Moorer, "Signal Processing Aspects of Computer Music – A Survey," *Computer Music Journal*, vol. 1, no. 1, pp. 4–37, Spring 1977.

[6] Sylvain Marchand, "InSpect+ProSpect+ReSpect Software Packages," Online. URL: `http://www.scrime.u-bordeaux.fr`, 2000.

[7] E. Zwicker and H. Fastl, *Psychoacoustics Facts and Models*, Springer Verlag, 1990.

[8] Eberhard Zwicker and Richard Feldtkeller, *Psychoacoustique – L'oreille, récepteur d'information*, Masson, Paris, 1981, In French.

[9] Eberhard Zwicker and Ulrich Tilmann Zwicker, "Audio Engineering and Psychoacoustics: Matching Signals to the Final Receiver, the Human Auditory System," *Journal of the Audio Engineering Society*, vol. 39, no. 3, pp. 115–126, March 1991.

[10] Guillermo Garcia and Juan Pampin, "Data Compression of Sinusoidal Modeling Parameters Based on Psychoacoustic Masking," in *Proceedings of the International Computer Music Conference (ICMC)*, Beijing, China, October 1999, International Computer Music Association (ICMA), pp. 40–43.

[11] Ted Painter and Andreas Spanias, "A Review of Algorithms for Perceptual Coding of Digital Audio Signals," Online. URL: `http://www.eas.asu.edu/~spanias`, 2001.

[12] P. Papamichalis, "MPEG Audio Compression: Algorithm and Implementation," in *Proceedings of the International Conference on DSP*, June 1995, pp. 72–77.

[13] William Pugh, "Skip Lists: A Probabilistic Alternative to Balanced Trees," *Communications of the ACM*, pp. 668–676, 1990.