SCALABLE METADATA AND QUICK RETRIEVAL OF AUDIO SIGNALS

Nancy Bertin, Alain de Cheveigné
Equipe Audition
CNRS UMR 8581 - ENS (DEC)
29, rue d'Ulm
75005 PARIS
Nancy.Bertin@ens.fr, Alain.de.Cheveigne@ens.fr

ABSTRACT

Audio search algorithms have reached a degree of speed and accuracy that allows them to search efficiently within large databases of audio. For speed, algorithms generally depend on precalculated indexing metadata. Unfortunately, the size of the metadata follows the same exponential trend as the audio data itself, and this may lead to an exponential increase in storage cost and search time. The concept of *scalable metadata* has been introduced to allow metadata to adjust to such trends and alleviate the effects of forseeable increases of data and metadata size. Here, we argue that scalability fits the needs of the hierarchical structures that allow fast search, and illustrate this by adapting a state-of-the-art search algorithm to a scalable indexing structure. Scalability allows search algorithms to adapt to the increase of database size without loss of performance.

Keywords: Search, indexing, scalability, audio retrieval, scalable metadata.

1 INTRODUCTION

This paper investigates the usefulness of scalable content-based metadata for quick retrieval of music and audio data. In agreement with the well-known law of Moore and its avatars (Odlyzko, 1999), we observe an exponential growth of multimedia content available on the web, cable networks, DVDs, hard disks, etc. This constitutes a problem for those who manipulate the content (producers and consumers alike), and also for the algorithms and tools that deal with such data. The concept of *metadata* was introduced as an attempt to alleviate problems associated with manipulating large quantities of data. Metadata are designed to summarize data in a format that is compact so as to minimize storage costs, and optimized for manipulations such as search and retrieval. Metadata are however likely to grow at the same rate as the data itself. Growth of metadata may be addressed by designing new metadata formats, either more compact descriptions of the data, or else second-order descriptions of the metadata. However new formats introduce interoperability problems as well as ergonomy problems for the user who must learn new tools to deal with these new formats.

The concept of *scalable metadata* was introduced to address these issues (de Cheveigné, 2002). A definition of scalability is given in Sect. 2; in brief, metadata are scalable if they have the same semantics at all scales, can be converted from high- to low-resolution scale (rescaled), and do not depend on the intervening rescaling operations. Scalable metadata can thus be adjusted to fit databases of any size, and the needs of any application, while retaining consistent semantics across scale. Metadata storage formats, and tools to manipulate them, thus remain the same whatever the future increases in data size. Arguably, metadata *must* be scalable to follow future increases in data size and so non-scalable formats will eventually be superseded (de Cheveigné, 2002).

A crucial function in any system that handles large quantities of data is *search*. Search is needed for query operations, but also for signal processing, display, housekeeping, etc. Without adequate attention to efficiency and asymptotic properties, the cost of search operations can very easily grow out of bounds. For example a search time linear in size is obviously prohibitive if size increases exponentially. Efficient algorithms have been developed for e.g. strings or trees, but relatively fewer are effective for multimedia data. Examples are (Luettgen and Willsky, 1995; Spence and Parra, 2000; Jain et al., 1999; Crestani et al., 1998; Kashino et al., 1999, 2003). Efficient search methods usually involve construction of an index structure within which search proceeds. The indices can be understood as metadata, and this raises the issue of their scalability. It also raises the question of the usefulness of metadata for search in general. For all practical purposes, search within a multimedia database is search within the *metadata* associated with the content. The effectiveness of scalable metadata as a substrate for search is an essential question.

Here, we start from a state-of-the-art audio search algorithm (Kashino et al., 1999, 2003) and show that it can be adapted to take advantage of scalable metadata. Scalability eases the implementation of a hierarchical metadata structure that speeds search. Efficient search involves *pruning* of the search space at the earliest possible stage. The statistical operations involved in the definition of scalable metadata (extrema, mean, variance, histogram, etc.) are effective for this purpose, as these statistics allow inference of the presence (or better: the absence) of a search token within the subset of data that they summarize. Such statistics have been exploited previously (Gaede and Günther, 1998; Spence and Parra, 2000; Sivakumaran et al., 2001) for search within audio or image domains.

The notion of scalability is defined in the next section. Principles of efficient search are outlined in Sect. 3. Experiments to test the effectiveness of hierarchical search are described in Sect. 4.

2 SCALABLE METADATA

2.1 Properties and design

Three properties characterize scalable metadata. First, they may be instantiated at any resolution. Second, an existing description may be converted automatically to a lower-resolution one. Third, the description at a given resolution depends only on the resolution, and not on the history of scaling operations that led to it. Scaling entails loss of information: low resolution metadata contain less information than high resolution, and conversion from lower to higher resolution is not possible. Lower resolutions are usually imposed by application or system-dependent storage or processing constraints. The scalability property allows low and high resolution metadata to be compared, thus ensuring interoperablity.

Scalability is provided by a set of well-defined operations that transform series of numerical descriptor values from one resolution to the next. Among them are: extrema (*min* and *max*), *sum*, *mean* and *histogram*. *Variance* and *covariance* are also scalable if associated with the mean. Others are described in (de Cheveigné, 2002; de Cheveigné and Peeters, 1999a; ISO/IEC_JTC_1/SC_29, 2001), in particular a "scalewise variance" that measures variance across scale, as well as "weighted" versions of the above operations. Weighted operations allow certain samples to be discounted before scaling, for example if they are known to be locally undefined or unreliable. A scalable data structure for metadata consists of the metadata (represented as a series of scalars, vectors or matrices) together with information that describes its "geometry": name of the scaling operation that was (or should be) applied to it, number of samples, scale ratio, etc. The structure contains all that is needed by an application to make sense of the data. It also contains all that that is needed to *rescale* it. Metadata storage or transfer systems can thus adjust to the resolution according to storage capacity or bandwidth constraints.

The scalable structure is designed to be "plugged in" to a content-based descriptor, to serve as a container for the series of descriptor values. Typically, an audio descriptor such as spectrum or power is calculated from audio content and then stored with on one hand descriptor-specific parameters (frame rate, spectral resolution, etc.) and on the other a series of descriptor values. The latter is stored in a scalable data structure. In some cases it is useful to think of the scaling operation as part of the definition of the descriptor. For example, the power spectrum of a segment of audio is equal to the mean of the power spectra of the frames within that segment. Power can be defined with whatever granularity. In other cases, the scaling operations are best understood as statistics that describe the distribution of values that they summarize. For example, mean and covariance can be used to parametrize a gaussian distribution. A given descriptor may eventually be scaled according to several different operations, each at its own scale, all stored within the same description.

Scalability was introduced as a design principle of the audio part of the MPEG-7 metadata standard. Content-based audio descriptors are built using a "Scalable-Series" datatype that stores the series of descriptors values, and confers to each MPEG-7 audio description the scalability property (de Cheveigné and Peeters, 1999a,b, 2000; ISO/IEC_JTC_1/SC_29, 2001). Audio descriptors used in the MPEG-7 standard are defined in (ISO/IEC_JTC_1/SC_29, 2001). A useful property of the MPEG-7 data structures is that they can accommodate a descriptor instantiated at multiple scales, and with multiple scaling rules. Some examples of scalable audio descriptors and associated scaling operations are given in Tab. 2.1, others are described in (de Cheveigné, 2002).

Table 1: Descriptor examples.

Descriptor	Typical operations	Useful for
Waveform	Min, Max	Display, search
Power	Mean, variance	Search
Fundamental frequency	Weighted mean, Histogram	Query by humming
Power spectrum	Mean, Covariance, Histogram	Display, search

2.2 Scalability and the life-cycle of metadata

As evoked previously, scalability addresses issues related to the life-cycle of multimedia content and metadata. It enhances the useful properties of *interoperability* and *reusability* of metadata descriptions, that are further enhanced by being standardized. Metadata resolution requirements typically vary across applications and across time, and the same is true of storage and transport constraints. Metadata produced at one time for one application may need to be reused at another time by another application. A level of detail that is appropriate at one time may later be excessive, when the volume of data to describe has increased. Scalability allows the same metadata formats to be used for every application. It frees the application designer from the difficult decision of the "right" resolution. It also allows rescaling operations to be scheduled at the level of the transport or storage system, so that metadata may be thinned rather than deleted when space comes to lack. Scalability thus extends the life-cycle of metadata.

2.3 Typical applications

Scalable metadata are useful for a range of applications. They can be used for display of audio data within a user interface (music browser or editor). For example, a "waveform" descriptor may be defined as a time-series of min/max pairs. This is sufficient to produce a full-resolution graphical display of the waveform within an audio file. As long as there are as many pairs as pixels horizontally, the result is identical to plotting all samples of the original waveform, and yet cheaper in terms of storage, transport and drawing. Likewise a sonagram-like spectrum descriptor (time-frequency representation). Descriptors may be combined, for example a low-resolution spectrogram associated with fundamental frequency and harmonicity measure, to approximate a high-resolution spectrogram. Descriptors may serve also to produce icons.

Scalable audio metadata may also be used to provide audio feedback for browsing. For example, a combination of a low-resolution power spectrum with descriptors of fundamental frequency, harmonicity and roughness allows a rough rendering of the texture of audio content to be restituted. By focusing on the *large scale* structure of documents or collections, such auditory or visual feedback is complementary to the more common practice of providing a human-edited clip, for instance. A synthetic "earcon" may thus be used to characterize file content

The most important function, however, is *search*. The importance of search increases as content grows, and manual operations become less feasable. Search is needed to respond to user demands to "find" specific content. It is also needed to support "house-keeping" operations at the system level, or to organize data into useful synthetic representations for a user interface. For example, a display that represents the content of a hard disk with duplicates colored in red would be of great use to a user. Search supports such functionality.

3 SCALABILITY AND SEARCH

Efficient search is based on *pruning*, that is, early elimination of the parts of the search space where the query is known not to be. This is typically done by organizing the search space hierarchically as a tree and attaching information to each node, such that by consulting that information the algorithm can know that the token is *not* within the subtree spanned by the node.

Organizing the data in hierarchical search structures is a principle shared in numerous domains: string matching (Fredriksson, 2004), image browsing and search (Spence and Parra, 2000; Chen et al., 2000), bioinformatics and DNA analysis (Eisen et al., 1998), speech processing (Zotkin and Duraiswami, 2004), etc. Many standard search algorithms use structures such as binary trees, red-black trees, B-trees (Cormen et al., 1990), hierarchical clustering (Krishnamachari and Abdel-Mottaleb, 1999). Multiple-resolution representations of the data may be used to label such trees to perform hierarchical search, either deterministic (Chen et al., 2000; Li et al., 1996) or probabilistic (Luettgen and Willsky, 1995; Spence and Parra, 2000).

Scalable metadata are well suited for this purpose. The index at each node is a scaled summary of the indices of lower nodes. The scaling operations mentioned above (extrema, mean, variance, etc.) offer statistics that describe the set of data that they summarize, and thus each node allows inference as to whether a search token is included in the subtree.

Extrema statistics support deterministic pruning: a node is pruned if the search token is out of bounds. As an example, an audio waveform comparison algorithm based on minima and maxima of the waveform is described in (de Cheveigné, 2002; ISO/IEC_JTC_1/SC_29, 2001). Other statistics such as mean and covariance allow probabilistic inference based on a parametric model of the data set, for example multivariate gaussian. The "scalewise variance" statistics described in (de Cheveigné, 2002; ISO/IEC_JTC_1/SC_29, 2001) allows yet finer characterization of the distribution, as do various combinations of these statistics.

Search proceeds from the root of the tree (lowest resolution) to the leaves (highest resolution). At each node, the decision is made whether to prune the subtree as a result of an inference based on the index of that node. Search is fast near the root, and more slow but reliable as it proceeds towards the leaves. For a very large database (or one that is distributed across the network), only the high-level nodes may be available on line, the off-line lower-level nodes being retrieved on demand, if available. Early pruning reduces the number of times that this potentially costly operation needs to be performed. Scalable metadata thus provide the hierachical search structures needed to support efficient search.

4 EXPERIMENT

The aim of this experiment is to demonstrate that scalable metadata can support effective content-based audio search. For that purpose we took a state-of-the-art search algorithm, adapted it to make use of a hierarchical structure based on scalable metadata, and compared its performance with that of the original.

4.1 Baseline algorithm

An algorithm that achieves efficient pruning *without* a hierarchical search structure is the "active search" algorithm of Kashino et al. (1999, 2003). The algorithm, based on histograms of vector-quantized spectra, allows a segment of audio (query token) to be found within a database of audio documents. In brief, the database to be searched is indexed by calculating power spectra at a given frame rate. Spectra warped to a logarithmic frequency axis are vector-quantized using a code book to form a time-series of VQ code indices. This series constitutes an index within which the search proceeds.

At search time, indices of the database are accumulated over a running window to form a time series of *histograms*. At each frame, the histogram is compared to a similar histogram formed from the query token, until a match is found. The efficiency of search stems from the fact that, after each comparison between the query histogram and the current histogram, the algorithm can skip forward a number N of index samples equal to the largest mismatch between corresponding bins of token and database histograms. Indeed, a time shift of *at least* N frames is required to resorb that mismatch. The algorithm is thus fast.

As just described, the algorithm is sure to find a match if the analysis frames of query and database are temporally aligned. However, perfect alignment is not always guaranteed, and misalignment may cause the VQ series to differ slightly. Taking this into account, the algorithm uses a measure of similarity defined as:

$$S(h_Q, h_C) = \frac{1}{L}\sum_{k=1}^{L} min(h_Q(k), h_C(k))$$

where $h_Q(k)$ is the k-th coefficient of the query histogram, $h_C(k)$ the current histogram (taken from the database), and L the codebook size. The token is considered to have been found when the similarity exceeds a certain threshold.

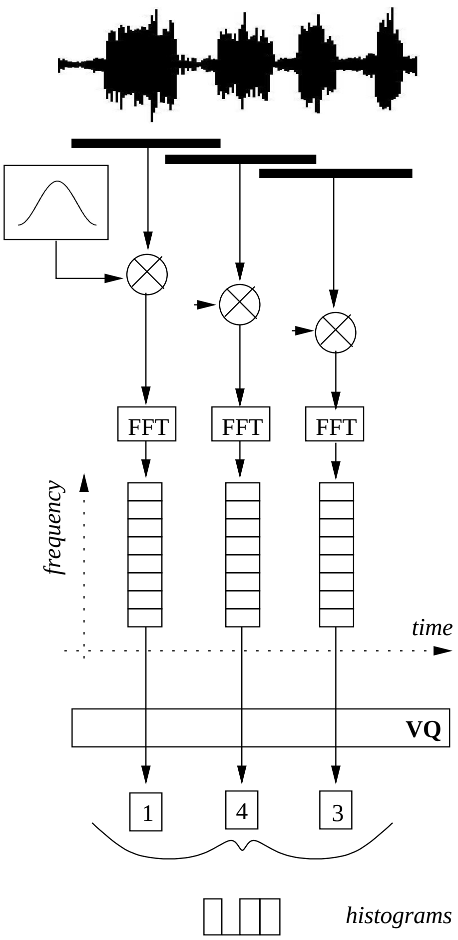

Figure 1: Schematic diagram of the indexing phase. Overlapping frames are windowed and Fourier-transformed to obtain power spectra that are then warped to a logarithmic frequency axis and vector-quantized to produce a series of codes that are then aggregated into histograms.

A useful feature of the "active search" algorithm of Kashino et al. (1999, 2003) in our context is that it is based on histograms. The histogram operation is scalable, and therefore this efficient algorithm is a good starting point to develop a scalable search algorithm also based on histograms. The similarity between the two histogram-based algorithms makes it easy to draw insights from their comparison.

4.2 Hierarchical histogram-based search

The first steps of the scalable algorithm (Fig. 1), are similar to the active search algorithm of Kashino et al. (1999, 2003). Log-frequency spectra are vector-quantized to form a series of VQ codes that are then grouped into a series of histograms. In contrast to the active search algorithm, a series of histograms, rather than VQ codes, forms the searchable index. The histogram series are then scaled by successive powers of two to form a hierarchical search structure (binary tree) that indexes the database (Fig. 2). At search time, a histogram is similarly calculated from the query token and compared to the histogram attached to each node of the database index structure, starting from the root. Histogram comparison determines whether the query token is *included* within the segment spanned by a node. More precisely, if at frame j and for all codebook indices k we have:

$$h_Q(k) \le h_C(k, j),$$

then all spectra within the query token are also found within the database segment spanned by this node. If the test fails, that segment of the database may be pruned. If the test succeeds, it is repeated at successively finer resolutions until it fails or the token is found.

Actually the algorithm is slightly more complex than just described. So far, a match is guaranteed only if the query token happens to be temporally aligned with a same-sized database interval spanned by a node, an unlikely event. To allow arbitrary aligment, the above comparison is replaced by $h_Q(k) \le h_C(k, j) + h_C(k, j+1)$. If the segment spanned by the latter two histograms is shorter than twice the query token, the query token is split in two and search proceeds with each half. A further complication arises because, as in the active search algorithm of Kashino et al. (1999, 2003), the temporal aligment of search token analysis frames and database analysis frames is not guaranteed. To allow for slight misalignments, the perfect match criterion must be replaced by an approximate match criterion involving a threshold.

4.3 Database and implementation

Experiments were performed using the RWC database (Goto et al., 2002, 2003) that contains a total of approximately 24 hours sampled at 44100 kHz with 16-bit resolution of various types of music (classical, Japanese popular music...).

Spectral features were calculated every 16 ms by applying an FFT to a 32 ms window shaped as a raised cosine. The frequency axis was warped to a logarithmic scale by grouping power spectrum bins into 8 one-octave bands ranging from 65.2 to 16000 Hz. Spectrum values were raised to a power of 1/3 to improve the shape of their distribution. The advantage of a logarithmic scale is that it parallels the spectral resolution of the ear (although with cruder resolution) and distributes signal power evenly among bands (most spectra are low-pass). The 1/3 exponent simulates partial loudness (Hartmann, 1997). This relatively crude resolution (about three times poorer than the ear) is motivated by the need for concision. The procedure is similar to that specified in the MPEG-7 standard.

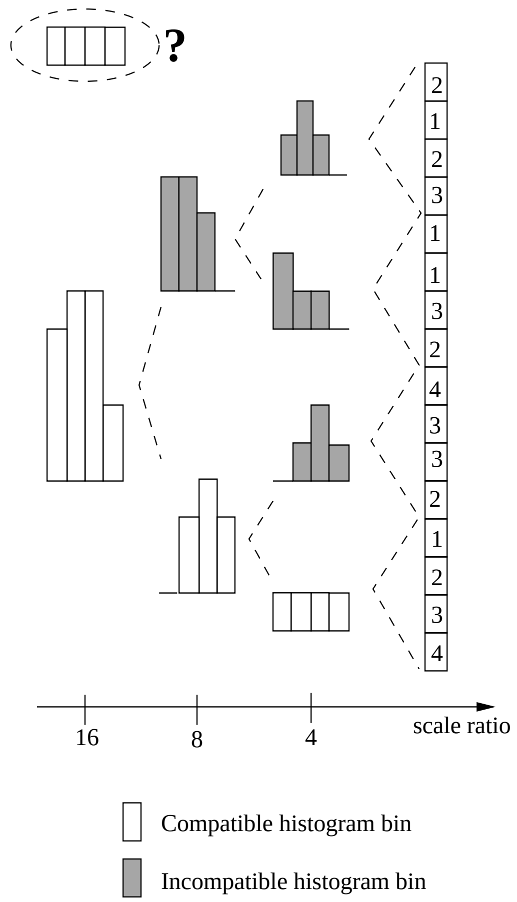

Figure 2: Schematic diagram of the hierachical search algorithm. The database is indexed by a series of codevectors (right) from which is derived a binary tree structure of aggregate histograms. Search proceeds from the root. The query (represented by a histogram) is compared to each histogram in turn. In case of failure, the subtree spanned by that node is pruned (gray). In case of success (white), search proceeds within the subtree until the token is found, or the search returns unsuccessful.

A codebook was prepared by applying the LBG vector quantization algorithm (Linde et al., 1980) to a subset of the database (about 20%). The codebook size was 512, adequate for a 8-dimension feature space. Codebook size and spectral resolution result from a compromise between discriminative power and practical constraints. For comparison, 2-bin quantization of each dimension of a 1/3 or 1/4 octave spectrum would require a codebook of size 2^{24} or 2^{32} which is clearly unweildy. The codebook (dictionary) must be included with the indexing metadata attached to the database to be searched (or else it must be "well-known", for example defined by a standard). A codebook prepared in this way will obviously depend upon the database from which it was prepared. Ideally one would like to have a sort of "universal" codebook, built on a very large database and re-usable. Whether such a goal is feasable remains to be determined.

The time series of spectral features was quantized using the VQ code book and used to implement both the active search algorithm of Kashino et al. (1999, 2003) and the hierarchical algorithm. To implement the hierarchical search algorithm, series of VQ codes were aggregated to form histograms, and then scaled repeatedly and stored within a hierarchical data structure. For simplicity, both algorithms were implemented so that analysis frames of query and database were aligned, and thresholds (see above) were set to the ideal value corresponding to an exact match without any tolerance. This affects both algorithms equally. A third algorithm, exhaustive search, was also implemented for comparison. It simply consists in a direct comparison of the sequence of vector-quantized spectra of the query with same-sized subsequences in the database. This naive algorithm is useful as an upper bound for calculation time.

Subsets of the database of varying size were extracted in order to investigate the relation between search speed and database size. A set of 48 query excerpts of 10s duration was used as search tokens for both algorithms. Both algorithms were set up to search for multiple occurrences (i.e. search did not terminate after the first match). Search speed was quantified by counting the average number of histogram comparisons, or average CPU time necessary to find a query token. Simulations were performed using Matlab 7.0.1 on a PC with a Pentium 4 - 3 GHz processor.

5 RESULTS AND DISCUSSION

5.1 Results

Table 2 and Fig. 3 summarize the results. The size of the search space is varied as a parameter. Cost of exhaustive search (top line) varies linearly with space size. Active search (middle line) also follows an approximately linear trend, but is over two orders of magnitude faster than exhaustive. Hierarchical search (bottom line) follows a less-than-linear trend, with a speedup ratio of 7 to 40 over active search. Results for CPU time are similar.

Table 2: Search time and speed-up ratio (SU) for active search (AS) and hierarchical search (HS).

	1h	2h	6h	13h	24h	48h
AS	525	803	2726	5043	9794	19588
HS	69	72	123	158	272	481
SU	7,6	11,2	22,2	31,9	36	40,7

The hierarchical algorithm based on scalable metadata is faster than the baseline algorithm, itself known to be competitive. This shows that scalable metadata can support efficient search. The result is welcome, as we argued earlier that scalability is a necessary property of metadata.

Both algorithms attained 100% accuracy (defined as the mean between precision and recall rate). This high

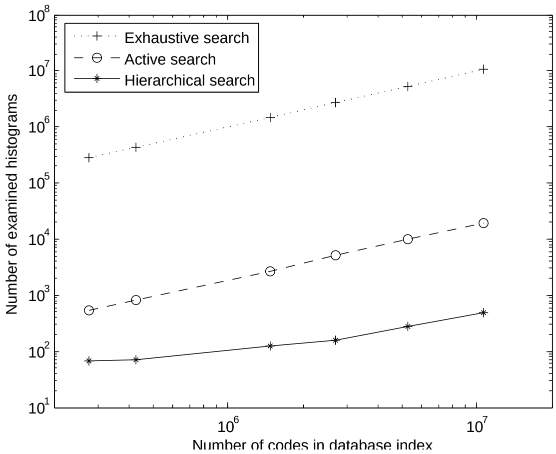

Figure 3: Search time as a function of database size.

level of accuracy was obtained for the case where analysis frames of query and database were temporally aligned. While this can be assumed in some applications, in others there is no guarantee of temporal alignment. Misalignment produces a mismatch that must be compensated by allowing histogram comparisons to be tolerant. This in turn increases the proportion of "false positives" and causes subtree pruning to be less effective, thus increasing search time. The active search and hierarchical search algorithms are equally affected. Similar "fuzzy-match" schemes are required if search is expected to tolerate distortions (such as produced by lossy coding).

5.2 Discussion

Scalability puts strong constraints on the semantics and structure of metadata. Here we found that these constraints are compatible with the important function of search, and that scalability may actually help to implement hierarchical structures that support very efficient search. Efficiency stems from the hierarchical organization of the search space induced by scaling.

Scalability implies that the temporal order of labels at each level must be conserved. This leads to a hierarchical structure that less efficient than, say, hierarchical clustering by similarity. It is conceivable that efficiency could be enhanced by reorganizing the search tree at search time. More research is needed to develop index schemes that are both scalable and efficient.

Speedups may also be expected by improving the distributional properties of the descriptor, for example by normalization by power (separately vector-quantized), or transformation to a "cepstrum-like" representation with a covariance matrix closer to diagonal. The index may also be improved by including additional descriptors to capture features not well represented by the octave-band spectrum: fundamental frequency, harmonicity, roughness, etc.

An attractive feature of histogram-based search is that it can be applied to any descriptor. Useful examples are fundamental frequency, chroma, event counts, or even text-based descriptors. All can be quantified and stored as histograms. A constraint is that the coding (quantizing) stage requires a dictionary. Scalability requires that the same dictionary be used to quantize all data, and this implies the existence of a well-known and accepted (i.e. standardized) dictionary. Whether a universal dictionary can be found for each descriptor that can cover all needs is an important issue.

The present paper examined only algorithms based on the histogram. Other scalable operations as extrema, mean and covariance, scalewise variance, etc. produce statistics that can be used to quantify the distribution of values that they summarize. These too are expected to support efficient search. More research should be devoted to the issue of search within scalable representations.

6 CONCLUSIONS

In this paper, we investigated search within audio content based on scalable metadata. We verified that a standard search algorithm can be applied to scalable metadata, and found additionally that search could be speeded by making use of the hierarchical structure offered by scalable metadata. While we did not test other search algorithms, it is reasonable to believe that they too can be implemented and benefit from scalability. Scalability is a property that is necessary to allow metadata to follow future trends in data size. It is a welcome result that it can support essential operations such as search.

ACKNOWLEDGEMENTS

The authors wish to thank Daniel Pressnitzer, Dan Gnansia and Maria Chait for their useful comments and friendly support during the conduct of this research.

REFERENCES

J.-Y. Chen, C. Bouman, and J. Dalton. Hierarchical browsing and search of large image databases. In *IEEE transactions on Image Processing*, volume 9, pages 442–455, march 2000.

T. Cormen, C. Leiserson, and R. Rivest. *Introduction to algorithms*. The MIT Press, 1990.

F. Crestani, C. Rijksbergen, and I. Campbell. Is this document relevant ?... probably: a survey of probabilistic models in information retrieval. In *ACM Computing Surveys*, volume 30, pages 528–552, 1998.

A. de Cheveigné. Scalable metadata for search, sonification and display. In *Proceedings of the 2002 International Conference on Auditory Display*, Kyoto, Japan, July 2002.

A. de Cheveigné and G. Peeters. Core set of audio signal descriptors. Technical Report JTC1/SC29/WG11, MPEG00/m5885 technical report, ISO/IEC, 2000.

A. de Cheveigné and G. Peeters. Scale tree. Technical Report JTC1/SC29/WG11, MPEG99/m5076 technical report, ISO/IEC, 1999a.

A. de Cheveigné and G. Peeters. Scale tree update. Technical Report ISO/IEC JTC1/SC29/WG11, MPEG99/m5443 technical report, ISO/IEC, 1999b.

M. Eisen, R. Spellman, P. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. In *Proc. Natl. Acad. Sci. USA, Genetics*, volume 95, pages 14863–14868, december 1998.

K. Fredriksson. Metric indexes for approximate string matching in a dictionary. In *Proceedings of the 11th International Symposium on String Processing and Information Retrieval (SPIRE'2004)*, LNCS 3246, pages 212–213. Springer–Verlag, 2004.

V. Gaede and O. Günther. Multidimensional access methods. In *ACM Computing Surveys*, volume 30, pages 170–231, 1998.

M. Goto, H. H., N. T., and R. Oka. RWC music database: Popular, classical and jazz music databases. In *Proc. of International Conference of Music Information Retrieval (ISMIR)*, pages 287–288, 2002.

M. Goto, H. H., N. T., and R. Oka. RWC music database: Music genre database and musical instrument sound database. In *Proc. of International Conference of Music Information Retrieval (ISMIR)*, 2003.

W. Hartmann. *Signals, Sounds and Sensation*. AIP Press, 1997.

ISO/IEC_JTC_1/SC_29. Information technology - multimedia content description interface - part 4: Audio. Technical Report ISO/IEC FDIS 15938-4, 2001.

A. Jain, M. Murty, and P. Flynn. Data clustering: a review. In *ACM Computing Surveys*, volume 30, pages 265–321, 1999.

K. Kashino, G. Smith, and H. Murase. Time-series active search for quick retrieval of audio and video. In *Proc. of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 6, pages 2993–2996, March 1999.

K. Kashino, T. Kurozumi, and H. Murase. A quick search method for audio and video signals based on histogram pruning. *IEEE Transactions on Multimedia*, 5:348–357, Sep. 2003.

S. Krishnamachari and M. Abdel-Mottaleb. Hierarchical clustering algorithm for fast image retrieval. In *IS&T/SPIE Conference on Storage and Retrieval for Image and Video Databases VII*, pages 427–435, San Jose, California, January 1999.

C.-S. Li, P. Yu, and V. Castelli. Hierarchyscan: A hierarchical similarity search algorithm for databases of long sequences. In *Proc. of IEEE Conference on Data Engineering*, 1996.

Y. Linde, B. A., and R. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, pages 702–710, January 1980.

M. Luettgen and A. Willsky. Likelihood calculation for a class of multiscale stochastic models, with application to texture discrimination. In *IEEE Transactions on Image Processing*, volume 4, pages 194–207, 1995.

A. Odlyzko. The current state and likely evolution of the internet. In *Proc. Globecom '99, IEEE*, pages 1869–1875, 1999.

P. Sivakumaran, J. Fortuna, and A. M. Ariyaeeinia. On the use of the bayesian information criterion in multiple speaker detection. In *Proc. Eurospeech*, pages 795–798, 2001.

C. Spence and L. Parra. Hierarchical image probability (HIP) models. In *Proc. Advances in Neural Information Processing Systems*, pages 848–854, 2000.

D. Zotkin and R. Duraiswami. Accelerated speech source localization via a hierarchical search of steered response power. In *IEEE Transactions on Speech and Audio Processing*, volume 12, pages 499–508, 2004.