Indexation scalable de documents sonores

Nancy BERTIN

Mémoire pour le Master de Sciences et Technologies
Université Pierre et Marie Curie
Mention SDI, Spécialité MIS, Parcours ATIAM

sous la direction d'Alain de CHEVEIGNÉ

Equipe Audition
Département d'Etudes Cognitives, ENS
Laboratoire de Psychologie Expérimentale, CNRS, UMR 8581

Mars-Juin 2005

Table des matières

Remerciements

En tout premier lieu, j'aimerais remercier Alain de Cheveigné, mon directeur de stage, de m'avoir proposé de le rejoindre. Enthousiaste, disponible (même à des milliers de kilomètres !), il m'a laissé une grande autonomie de travail tout en sachant rester présent quand il le fallait.

Merci à toute l'équipe Audition pour la chaleureuse ambiance de travail dans le laboratoire : Dan et ses péripéties immobilières, Daniel et ses mésaventures au troisième étage, Maria et son regard frais et salutaire sur nos mauvaises manies françaises ! Tous ont permis de faire de ce stage un moment non seulement pédagogiquement profitable mais aussi agréable et humainement riche.

J'aimerais également remercier particulièrement Denis Matignon pour son soutien lors de ma candidature, l'année passée.

Enfin, je profite de cette occasion pour remercier les enseignants et étudiants du Master ATIAM pour cette année à la hauteur de mes rêves d'adolescente. Sans oublier tous les chercheurs de l'IRCAM, ayant fait naître ma vocation lors des journées Portes Ouvertes de 1995, pour leur passion communicative, leur enthousiasme, leur patience, leurs encouragements, leurs conseils. C'est en grande partie grâce à eux que j'ai pu arriver jusqu'ici aujourd'hui.

Résumé

Ce document décrit la mise au point d'un algorithme de recherche efficace dans les données sonores, fondé sur la notion de scalabilité.

Pour s'adapter à la croissance exponentielle de la quantité de données sonores, et combattre les difficultés de manipulation qui en résultent, les métadonnées utilisées pour l'indexation audio doivent être scalables. La scalabilité est définie par la possibilité pour les métadonnées d'être instantiées à n'importe quelle échelle, et d'être converties de façon ordonnée d'une résolution fine vers une résolution plus grossière. Les métadonnées scalables permettent de mettre au point des outils de visualisation et de recherche viables à très grande échelle.

L'objectif du travail de stage était, en prenant pour acquise la contrainte de scalabilité, de démontrer que les métadonnées qui respectent cette contrainte supportent efficacement les opérations de recherche dans les très grandes bases de données, la visualisation et la navigation, etc. Nous avons réussi à mettre au point un algorithme très efficace de recherche dans une base de données sonores qui démontre à la fois la compatibilité des structures scalables avec la recherche d'information musicale, et le gain apporté par leur utilisation.

Le facteur d'accélération par rapport à un algorithme représentatif de l'état de l'art (l'algorithme de «recherche active» de Kashino et al., 2003) atteint 30 pour des bases grandes. De plus ce facteur croît avec la taille de la base, ce qui constitue une propriété prometteuse pour les systèmes de recherche d'information musicale futurs, qui auront à gérer des données largement plus massives qu'aujourd'hui.

Avant-propos

L'équipe d'accueil

J'ai effectué mon stage au sein de l'équipe *Audition : psychophysique, modélisation, neurosciences*. Cette jeune équipe possède une double affiliation : au Département d'Etudes Cognitives (DEC) de l'ENS [1] et au Laboratoire de Psychologie Expérimentale [2] (LPE, UMR CNRS 8581).

L'équipe compte actuellement quatre membres statutaires et un membre attaché. Elle possède déjà de nombreuses collaborations internes, internationales et universitaires.

Les recherches de l'équipe portent majoritairement sur les bas niveaux de traitement auditif de l'information acoustique. Différents thèmes sont étudiés à l'aide d'approches distinctes et complémentaires : psychophysique, électrophysiologie, modélisation computationnelle, neuroimagerie, audiologie, et neuropsychologie. Le point central de ces recherches consiste en la mise en évidence et l'étude des mécanismes périphériques (cochlée, nerf auditif) et centraux (tronc cérébral, cortex) impliqués dans l'analyse et l'intégration temporelles réalisées à différentes échelles (de la microseconde à plusieurs secondes) par le système auditif. Ces études sont majoritairement appliquées à la compréhension des capacités d'analyse de scènes auditives, de calcul d'attributs auditifs (tels que hauteur et timbre), de catégorisation des sons, de reconnaissance de la parole ou de perception de la musique chez des sujets entendants, ou malentendants appareillés ou implantés. Des applications dans le domaine des algorithmes et des prothèses et implants sont développées et testées dans le cadre de ces recherches.

L'équipe possède également des liens avec le milieu hospitalier. Un Groupement de Recherche en Audiologie Expérimentale et Clinique (GRAEC) dirigé par les équipes Audition et Psycholinguistique du LPE fédère plusieurs services hospitaliers Parisiens d'ORL et de Neurologie (7 hôpitaux) et plusieurs industriels de la prothèse auditive et de l'implant cochléaire. Des recherches plus cliniques sur

[1] DEC, 45 rue d'Ulm, 75230 Paris cédex 05 ; Directeur : Pr. D. Andler ; andler@hippo.ens.fr ; Tél : 01.44.32.36.50 ; Fax : 01.44.32.36.10

[2] UFR-Institut de Psychologie, Université René Descartes Paris 5, 71 avenue Edouard Vaillant, 92774 Boulogne Billancourt cédex ; Directeur : K. O'Regan - DR1 CNRS : oregan@idf.ext.jussieu.fr ; Tél. 01.55.20.59.26 ; Fax : 01.55.20.58.54

la surdité et l'intelligibilité dans le bruit sont menées dans le cadre du GRAEC.

Sujet du stage

Le sujet du stage m'a été présenté ainsi :

> Le stage vise à explorer le concept de scalabilité des données d'indexation de documents sonores. La loi de Moore (sous un de ses avatars) prévoit que le volume des données (sur le web, sur les supports de stockage) augmente exponentiellement. En comparaison, la «bande passante» cognitive et comportementale de l'utilisateur varie peu. Il en résulte une difficulté croissante pour naviguer et manipuler les données sonores. Le concept de *métadonnée* a été inventé en réponse à ce constat, mais les métadonnées elles-mêmes suivent aussi une croissance exponentielle, ce qui risque de reproduire le même problème. La *scalabilité* des métadonnées est une propriété qui leur permet de s'adapter à cette croissance. Les métadonnées scalables ont une résolution ajustable, et permettent ainsi de construire des structures d'indexation hiérarchiques qui facilitent la navigation et la recherche. Il s'agit d'implémenter un petit ensemble de descripteurs de signal (spectre, fréquence fondamentale, etc.), sous forme scalable, et d'explorer les possibilités qu'elles offrent pour la navigation et la recherche de données (sur disque ou sur web).

Déroulement du stage

La première phase du stage a consisté à mettre en place un outil d'indexation (extraction et remise à l'échelle) utilisant des descripteurs scalables, et tester leur utilité sur une application simple de visualisation.

La seconde phase s'est portée sur la mise au point d'un algorithme de recherche performant, utilisant les multiples échelles disponibles pour un même descripteur, dans le cadre d'un application de localisation d'un court extrait dans une base de données. Cette phase a été l'occasion d'étudier différentes approches et algorithmes du domaine dit de *Music Information Retrieval*.

Enfin, cet algorithme a été intégré dans un outil de visualisation et de recherches de doublons dans une grande base de données sonore.

Afin de souligner l'apport principal de mon travail, le présent document ne suit pas chronologiquement ce déroulement. J'ai ainsi choisi d'insister sur la mise au point et les résultats de l'algorithme de recherche hiérarchique.

Chapitre 1

Introduction : l'indexation audio et ses applications

1.1 Qu'est-ce que l'indexation audio ?

L'opération d'indexation vise à ajouter à un contenu des données supplémentaires qui permettent de classer ce contenu et de le manipuler. Ces données supplémentaires apportent une information sur les données brutes qu'elles concernent ; on les appelle *métadonnées*. Plus concises ou mieux structurées que les données de départ, elles visent à faciliter leur manipulation.

Deux exemples historiques : le livre et la bibliothèque

Deux exemples permettent de saisir intuitivement en quoi consiste l'indexation : l'index d'un livre, et l'étiquetage des livres dans une bibliothèque.

L'indexation d'un texte consiste à repérer dans celui-ci certains mots ou expressions significatifs, et à créer un lien entre ces termes et le texte original. Par exemple, les pages d'index d'un livre reprennent les termes significatifs apparaissant dans le livre, et les relient aux pages du livre où ces termes (ou leurs synonymes) apparaissent. Ceci facilite pour le lecteur la localisation des pages ou sections où l'on mentionne un sujet particulier. De même, la table des matières d'un livre est une forme (assez grossière) d'indexation.

Le second exemple est le système de classement des ouvrages d'une bibliothèque. L'étiquette accolée au livre apporte une information concise permettant de le localiser : outre par exemple, les trois premières lettres du nom de l'auteur, un code chiffré pourra signaler la catégorie (littérature, histoire, poésie...), un sous-code préciser cette catégorie (littératures de langue française, anglaise, allemande...). L'ensemble de ces données *indexe* la bibliothèque entière.

Indexation audio

Le cas de la musique est évidemment différent de celui du texte. La musique, les sons ne contiennent pas une signification univoque et consensuelle, tel ou tel morceau pourra être jugé triste ou gai par deux auditeurs. Si certaines informations objectives peuvent lui être attachées, comme un titre ou le nom de l'artiste, ces informations ne seront pas forcément suffisantes pour les manipulations qu'on souhaite en faire. Tandis que la quantité de musique disponible ne cesse de croître, on a en particulier besoin de descriptions pouvant être produites et comprises par des machines.

Indexation manuelle vs. indexation par le contenu

La typologie des différentes sortes d'indexation repose sur la manière de les produire. On distingue l'indexation «éditoriale» de l'indexation «par le contenu».

Les métadonnées éditoriales sont produites manuellement. Les plus courantes sont le titre du morceau, le titre de l'album qui le contient, l'artiste, mais on peut y ajouter un grand nombre d'autres informations. Outre des informations factuelles (date, lieu d'enregistrement...) on peut imaginer une étiquette de genre (rock, électro, baroque...) ou d'autres informations subjectives.

A l'inverse, les métadonnées basées sur le contenu sont extraites automatiquement, à partir du signal audio lui-même ; ce sont des métadonnées objectives. Des exemples de telles métadonnées sont donnés dans la section 1.4.

L'indexation textuelle pose des problèmes d'homogénéité des descriptions, de con-sensus sur les termes, et surtout de coût de production. En effet, produites à la main par des experts, elles sont coûteuses et peu reproductibles. Compte-tenu de la croissance exponentielle du volume de données, cette approche ne peut pas suffire.

L'avantage principal de l'indexation par le contenu est son caractère automatique : les algorithmes d'extraction fonctionnent sans intervention humaine et produisent des métadonnées uniformes et fiables. A supposer qu'on dispose de descripteurs qui prédisent des attributs perceptifs tels que le timbre, ces descripteurs remplaceraient avantageusement des étiquettes verbales subjectives et peu reproductibles.

L'indexation par le contenu présente cependant de nombreux problèmes qui sont autant de défis pour la recherche. Par exemple, dans les cas d'applications qui nécessitent de rechercher des sons similaires (et non seulement identiques), l'indexation doit avoir une pertinence psychoacoustique, afin que deux documents perceptivement proches soient décrits par des descripteurs de valeurs semblables. Un simple *hashage* du signal serait sensible à la moindre différence entre des signaux très similaires pour l'auditeur. L'indexation par le contenu rencontre également le problème de l'inter-opérabilité des systèmes ; sa normalisation fait d'ailleurs l'objet de nombreuses activités notamment dans le standard MPEG-7

[12, 10, 11].

Bien qu'il soit très différent des précédents, il paraît intéressant de mentionner un troisième type d'indexation, évoqué notamment dans [29] qui le baptise «indexation culturelle». Il s'agit ici d'utiliser les informations fournies par le contexte, l'environnement. En réalité ce terme recouvre deux choses : des systèmes basés sur des informations collectées auprès des utilisateurs (par exemple leurs profils) et sur un «filtrage collaboratif» ; des systèmes collectant des informations de co-occurrence (par exemple grâce aux moteurs de recherche sur Internet).

1.2 Schéma d'un système d'indexation

La figure 1.1 résume et illustre les différentes étapes d'un système d'indexation. Ces étapes ne sont pas toutes obligatoires et de nombreux choix peuvent être faits à chaque étape. Quelques exemples sont cités ici.

Le système se décompose en deux parties : une partie «frontale» (en anglais *front-end*) qui est la partie «acoustique et traitement du signal» du système, et une partie de «modélisation» des index obtenus, visant à obtenir une description plus concise et organisée.

Lors de la phase frontale, différentes étapes sont réalisées. Un pré-traitement peut être appliqué au signal acoustique suivant sa nature initiale (une conversion analogique - numérique, une conversion de fréquence d'échantillonnage, un filtrage destiné à simuler un éventuel canal...). Ensuite, le signal est régulièrement fenêtré, à une période telle qu'on peut raisonnablement espérer que le signal soit quasi-stationnaire sur une fenêtre (intervalles de quelques millisecondes). Les fenêtres et leur recouvrement sont choisies de manière à limiter les problèmes de discontinuité. Chaque trame donnera lieu ensuite à un «vecteur acoustique» (*feature vector*) décrivant la trame. Les étapes suivantes, de transformation et d'extraction, sont spécifiquement dévolues au calcul de ce vecteur. Les transformations sont généralement un passage à une représentation temps-fréquence ou temps-échelle. Elles ont pour but de réduire la redondance ou d'obtenir une description plus aisément manipulable lors des étapes ultérieures. Les étapes d'extraction sont très diverses tant du point de vue du résultat que des algorithmes déployés pour les obtenir. Enfin, une étape de traitement supplémentaire peut compléter cette phase en vue d'améliorer la sémantique du vecteur ou son utilisation future, par exemple une normalisation.

L'étape suivante de modélisation, a pour but principal, si elle a lieu, de réduire la dimension de l'espace des vecteurs acoustiques (*feature space*) ou de les organiser dans un modèle compact faisant sens. Selon que les données sont fixes ou que l'on peut en rajouter au fur et à mesure, les transformations ne seront pas toutes possibles. Elles vont de transformations linéaires comme l'analyse en composantes principales (ACP), l'analyse discriminante linéaire (LDA) ou des raffinements de celles-ci [4, 2], jusqu'aux modèles probabilistes évolués comme les

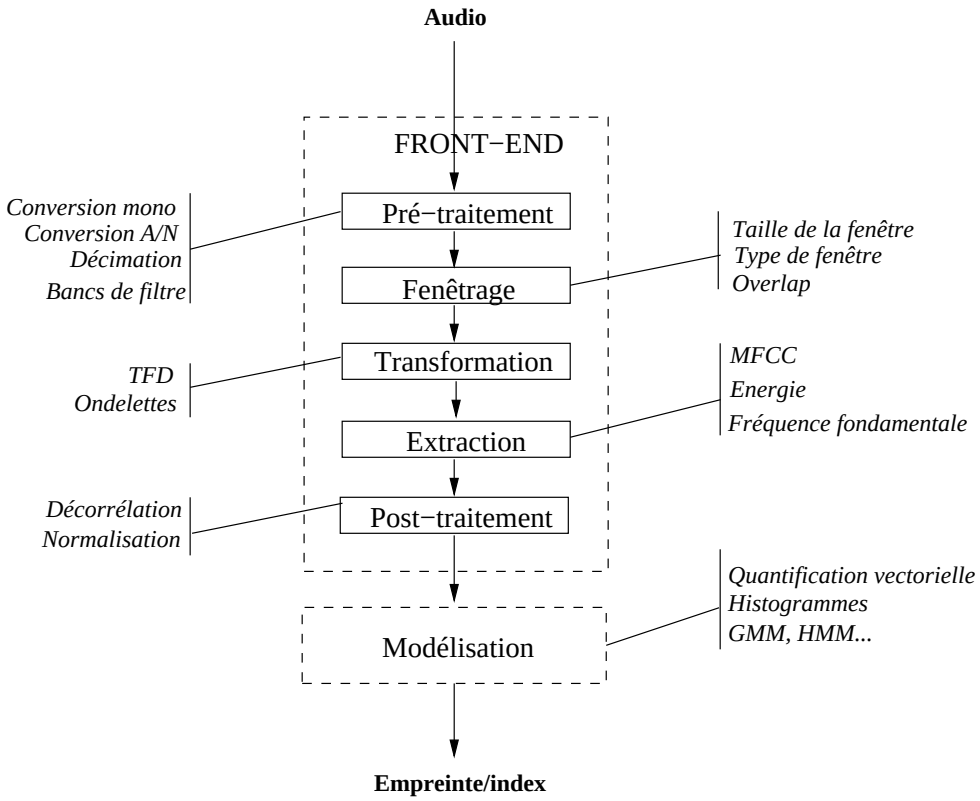

FIG. 1.1 – Les différentes étapes de l'indexation. D'après [5].

modèles de Markov cachés (HMM) très utilisés dans le domaine du traitement de la parole.

Ces principes sont par exemple décrits en détails dans [5].

1.3 Applications de l'indexation

Les applications de l'indexation audio et ses enjeux industriels et commerciaux sont extrêmement nombreux. En voici quelques exemples ; davantage de détails pourront par exemple être trouvés dans [23].

1.3.1 Navigation et visualisation

La navigation dans les bases de données est un problème de plus en plus aigu compte-tenu de l'explosion de la quantité de données disponibles, qu'il s'agisse de musique ou de tout autre type de contenu (textes, images...). Elle porte un double défi, à la fois du point de vue informatique (faisabilité et rapidité computationnelles) et ergonomique (le but majeur étant d'obtenir des outils facilement manipulables par les utilisateurs). La recherche d'interfaces innovantes, par exemple dans le domaine du texte ou de l'image, est très active : navigateurs «hyperboliques», interfaces zoomables, visualisations sous formes de graphes, utilisation de liens hyper-média... On en trouvera des exemples dans [7, 18, 6].

A ceci, s'ajoute la difficulté de visualiser sur un écran un contenu audio, par nature non visuel. La question de la représentation du son n'est pas nouvelle. Elle a reposé pendant longtemps sur des représentations temporelles ou temps-fréquence qui semblent inadaptées à une très grande quantité de données.

Dans ce contexte, l'indexation par le contenu est une voie prometteuse puisqu'elle offre de nombreux avantages : réduction de la dimension de l'espace, enrichissement des informations disponibles (ne se limitant plus aux sempiternels auteur-album-titre), métadonnées pouvant être exploitées pour des mesures de similarité... D'autre part elle pourrait permettre une navigation par requêtes, l'utilisateur spécifiant le type de contenu qu'il recherche.

1.3.2 Classification des sons

Une autre famille d'applications est celle qui relève de la classification des sons en différentes catégories prédéfinies : voix/musique, familles des instruments de l'orchestre, etc.. Ces applications s'apparentent à de nombreux champs de recherche dans les domaines dits de classification automatique, reconnaissance des formes ou apprentissage automatique (*Machine Learning*). De nombreux modèles de classification existent : modèles de Markov cachés, machines à vecteurs de support, réseaux de neurones... Tous reposent sur une extraction préalable de descripteurs qui formeront les vecteurs acoustiques décrivant les sons.

1.3.3 Recherche d'un extrait, identification

Une intense activité commerciale et industrielle ne cesse de se développer dans le domaine musical. L'explosion des systèmes «peer-to-peer», du format mp3 ou des appareils comme le récent *I-pod* témoignent de l'immense demande des consommateurs. En plus de l'accumulation de grandes quantités de données, les acteurs de l'industrie musicale veulent faire la différence en offrant au consommateur des services innovants.

Dans cette optique, des applications d'identification ont vu le jour. Le «query-by-humming » ou «query-by-example» désigne l'application qui consiste à fredonner la mélodie d'une chanson, ou en transmettre un extrait enregistré (par exemple via un téléphone portable) pour en obtenir l'interprète, le titre... Les opérations d'identification reposent sur l'indexation de la base dans laquelle on va rechercher l'extrait souhaité. Cette indexation doit évidemment être très robuste aux bruits, distortions, etc. entre l'extrait et le morceau original. De telles applications sont déjà disponibles sur Internet[1].

Une autre application commerciale dans l'air du temps est la «recommandation musicale» : connaissant les morceaux que l'utilisateur a déjà achetés, le système peut lui proposer d'autres morceaux similaires. L'idée de cette application reposerait sur des mesures de similarité entre descripteurs de type tempo, rythme... La constitution automatique de «playlists» assurant un enchaînement homogène des morceaux relève des mêmes idées. Ces applications sont encore au stade de projets de recherche.

1.3.4 Protection de la propriété intellectuelle

Il existe une forte demande d'outils performants pour lutter contre le piratage de la musique. Les outils d'identification et de recherche offerts par les descripteurs audio pourraient par exemple permettre de chercher sur un disque dur, ou de monitorer les transferts, de manière à détecter la présence de contenus acquis illégalement. De même, les sociétés protégeant le droit d'auteur comme la SACEM en France pourraient utiliser un tel outil pour monitorer les programmes radiophoniques.

Une autre technique développée en parallèle pour ces applications est le tatouage (*watermarking*), qui consiste à insérer dans les données elles-mêmes des marques inaudibles mais qu'on peut aisément retrouver dans le signal.

Une autre application à la protection du droit d'auteur pourrait être la recherche automatique de plagiats, basée par exemple sur des distances de similarité calculées à partir des descripteurs.

[1]Par exemple : Musart, http ://musen.engine.umich.edu

1.4 Descripteurs

Parmi les différents types de métadonnées, on appelle *descripteurs* les métadonnées basées sur le contenu et généralement de «bas niveau», c'est-à-dire relativement proche de la description acoustique du signal.

1.4.1 Critères de choix

S'il s'agit de calculer automatiquement une description à partir du signal sonore, les possibilités semblent infinies. Différents critères vont permettre d'orienter le choix :

- le pouvoir de discrimination : toute indexation doit caractériser au mieux le contenu indexé, garantir le mieux possible qu'on ne puisse le «confondre» avec un autre.
- l'efficacité computationnelle : un des enjeux de l'indexation est un gain en temps et en espace mémoire. On souhaite réduire le coût de production et de stockage. L'existence d'algorithmes permettant d'extraire rapidement les métadonnées, et la concision de la description, sont donc des critères importants de choix du descripteur.
- la dimension de l'espace : on a souvent besoin d'une réduction de la dimension de l'espace vectoriel de description (en anglais *feature space*), par exemple dans les applications de classification. Par exemple, le nombre de bandes de fréquences dans un descripteur de spectre sera un choix important, un nécessaire compromis devant avoir lieu entre la concision de la description et son pouvoir discriminant. A ce sujet, de nombreux auteurs évoquent *the curse of dimensionality*, la malédiction de la dimension, car un espace trop petit ne permettra pas une bonne discrimination, tandis qu'un espace trop grand sera coûteux à manipuler.
- la robustesse : elle n'est pas toujours requise. Dans certaines applications, on va vouloir identifier un contenu qui a subi des dégradations éventuellement sévères (compression et décompression, transmission par téléphone...), la robustesse du descripteur à ce type de dégradations pourra être requise. Dans d'autres, on ne recherche pas un extrait identique mais des extraits musicalement similaires. Les descripteurs devront être compatibles avec cette notion de similarité.
- la sémantique : quelle est la pertinence perceptive du descripteur, a-t-elle un sens du point de vue de l'auditeur, du point de vue musical ? S'il semble naturel de choisir une quantité portant un certain sens musical (comme la fréquence fondamentale, qui apporte une information au moins partielle sur la hauteur tonale), cette approche ne sera cependant pas forcément optimale du point de vue computationnel. Toutefois, suivant les applications, on ne pourra faire l'économie d'une approche psychoacoustique, car des signaux de formes d'onde très différentes peuvent avoir une grande similarité musi-

cale ; dans une application de recherche, par exemple, il sera donc essentiel que le descripteur soit compatible avec une certaine notion de similarité perceptive.

Le compromis entre ces différents critères se fera souvent suivant l'application envisagée, le compromis le plus critique étant souvent entre la précision de la description et sa concision.

1.4.2 Exemples de descripteurs

Il est pratiquement impossible de faire un inventaire exhaustif de tous les descripteurs possibles. Chaque système emploie sa propre version, ses propres paramètres, ses propres algorithmes d'extraction. Les tentatives de standardisation, dont la norme MPEG-7 [12, 10, 11], sont indispensables mais encore incomplètes (la norme MPEG-7 par exemple, ne traite pas du tout de la façon dont on doit extraire les descripteurs retenus, mais essentiellement de l'aspect syntaxique et sémantique). Quelques exemples sont présentés ici.

Certains descripteurs traitent de l'évolution des signaux dans le domaine temporel. Le plus élémentaire, retenu dans MPEG-7, est la forme d'onde du signal elle-même. Il peut évidemment paraître étrange de parler dans ce cas de *méta-données*, mais ce choix prendra son sens lorsqu'on introduira la scalabilité et les opérations de mise à l'échelle (voir section 2.1).

D'autres descripteurs plus directement manipulables se situent également dans le domaine temporel. L'enveloppe temporelle du signal, qui traduit ses fluctuations lentes, joue un rôle essentiel dans la perception auditive [17].

Le taux de changement de signe (*zero-crossing rate*) du signal est défini sur une séquence $\{s(n) : n \in \{0, ..., N\}\}$ par :

$$Z = \sum_{n=1}^{N} \frac{sign(s(n)) - sign(s(n-1))}{2}$$

où $sign$ désigne la fonction signe usuelle :

$$sign(x) = \begin{cases} 1 & \text{si } x \geq 0 \\ -1 & \text{si } x < 0 \end{cases}$$

On utilise également les taux d'ordre supérieur (taux de changement de signe des dérivées successives du signal). C'est quantités, peu coûteuses à calculer, ont prouvé leur bon pouvoir de discrimination [32].

L'énergie, la modulation, le facteur de crête, la sonie (*loudness*) ou l'impulsivité sont d'autres exemples de descripteurs du domaine temporel.

De nombreuses autres descriptions se situent dans le domaine spectral. Les premières d'entre elles sont de type spectrogramme : descriptions vectorielles (par bandes de fréquence) du spectre de puissance sur des trames successives. Elles

sont multiples car différents choix de «découpage» de la dimension fréquentielle sont possibles, ainsi que différentes opérations de transformation pouvant être appliquées aux valeurs obtenues. Les échelles et transformations s'inspirent des propriétés de l'oreille humaine. Les bandes de fréquence utilisées lors de l'analyse du signal s'inspirent des modèles de «filtres auditifs» décrits par les psychoacousticiens. Parmi eux, on peut citer l'échelle des Barks, les ERB et les mels [17]. On peut choisir une résolution fréquentielle plus ou moins fine suivant les performances attendues. L'échelle logarithmique approche grossièrement ces filtres, mais sa simplicité en fait souvent le choix final. On peut ensuite appliquer aux valeurs obtenues certaines opérations : passage au logarithme, à la racine carrée ou cubique, au cepstre. Ces transformations visent à améliorer la distribution des vecteurs, et éventuellement à simuler partiellement la sensation d'intensité [17].

Un exemple classique est l'exemple des *MFCC* pour *Mel Frequency Cepstrum Coefficients*. Le signal est d'abord fenêtré puis pour chaque trame, on calcule sa transformée de Fourier. Les coefficients sont alors convertis à l'échelle mel (ce qui correspond à un filtrage par un banc de filtres triangulaires, dont les fréquences centrales sont non linéairement espacées). Pour chaque coefficient S_k (dans le domaine de Fourier), le coefficient cepstral correspondant est donné par :

$$C_k = FFT^{-1}\left[\log|S_k|\right]$$

L'intérêt de cette représentation est à la fois l'échelle fréquentielle approchant le fonctionnement de l'oreille, et la transformation cepstrale qui a de bonnes propriétés de décorrélation (par exemple pour séparer les contributions dans un modèle source-filtre).

D'autres quantités peuvent décrire le spectre du signal. Le centroïde spectral (également appelé chroma, centre de gravité spectral, brillance) défini par :

$$CGS = \frac{\sum_{k=1}^{K} kA_k}{\sum_k = 1^K A_k},$$

où A_k désigne l'amplitude de la k^e bande de fréquence, contribue à la sensation de hauteur. Le rayon de giration spectral (dispersion du spectre autour de son centroïde) complète cette information. La platitude spectrale (*flatness*) définie par :

$$F = \frac{(\prod_{k=1}^{K} A_k)^{1/K}}{\frac{1}{K}\sum_{k=1}^{K} A_k}$$

est un autre exemple.

Deux autres descripteurs sont fréquemment utilisés : la fréquence fondamentale du signal et son harmonicité. L'extraction de la fréquence fondamentale est un problème compliqué de traitement du signal. De nombreux algorithmes existent, par exemple [8] pour une seule fréquence fondamentale. Dans le cas de l'estimation de fréquences fondamentales multiples, le problème reste encore largement

ouvert. L'harmonicité permet notamment d'adjoindre à la fréquence fondamentale estimée une valeur de confiance. Les descripteurs de fréquence fondamentale sont utiles dans de nombreuses applications dont la comparaison, la classification... Ils sont complémentaires des descripteurs de spectre, puisqu'ils apportent une information sur la structure temporelle fine du signal, ce que ne font pas les descripteurs de spectre qui n'ont pas une résolution suffisante.

Enfin, d'autres descripteurs de plus haut-niveau peuvent être visés, comme des descripteurs de rythme ou de tempo [31], de tonalité, de genre... Ils sont de nature bien différentes des précédents. Si ces descripteurs paraissent perceptivement plus explicites, leur extraction pose cependant de très nombreux problèmes pas toujours résolus à l'heure actuelle.

Un domaine également connexe est l'extraction d'empreinte audio (*fingerprint*), qui désigne un ensemble réduit de descripteurs ou construit à partir de descripteurs, permettant la discrimination.

Chapitre 2

Scalabilité

2.1 Le concept de scalabilité

Le concept de scalabilité est de plus en plus évoqué dans la littérature. Il désigne la capacité d'un système, conçu pour fonctionner à une échelle donnée, à fonctionner à plus grande échelle, c'est-à-dire pour un volume très important de données. On utilise ici une définition plus précise de cette propriété.

2.1.1 Intérêt

La loi de Moore et ses avatars [28] prévoit une croissance exponentielle du contenu multimedia disponible sur Internet, les réseaux, les supports matériels comme les DVDs ou les disques durs de nos ordinateurs personnels... Cela pose un problème à la fois pour ceux qui manipulent les contenus (qu'il s'agisse de producteurs ou de consommateurs) et pour les outils et algorithmes qui les utilisent. Si le concept de métadonnées a émergé en réponse à ces problèmes, il ne constitue qu'une solution incomplète, le volume des métadonnées étant susceptibles de croître suivant la même tendance exponentielle que les données. Concevoir de nouvelles métadonnées, de nouveaux formats, des descriptions plus concises, des *méta-métadonnées*, est une fuite en avant, qui pose de surcroît des problèmes de compatibilité, d'inter-opérabilité des anciens systèmes, ainsi que des problèmes d'ergonomie pour l'utilisateur, qui doit à chaque fois apprendre la manipulation de nouveaux formats et outils.

Le concept de *scalabilité* a été introduit en réponse à ce problème [9]. L'idée est simple : les métadonnées *scalables* peuvent s'ajuster à la taille des bases de données et aux besoins des différentes applications, via des opérations statistiques, en conservant une certaine sémantique à travers les différentes résolutions auxquelles elles peuvent exister. Les formats de stockage et les outils de manipulation peuvent ainsi être préservés, quelle que soit la croissance ultérieure de la quantité de données.

Chaque passage à une résolution plus faible cause évidemment une perte d'information. Le pari est que seules les métadonnées scalables deviennent utiles passée une certaine quantité de données, en raison d'une constance de la «bande-passante cognitive» de l'utilisateur.

On attend du concept de scalabilité qu'il réponde aux exigences liées au cycle de vie des données et des métadonnées, et qu'il convienne aux utilisations de ces dernières. Parmi les propriétés attendues, on souhaite qu'elles soient évolutives, flexibles, interopérables...

Les métadonnées sont rarement à la fois produites et consommées par le même système. Elles sont destinées à pouvoir être réutilisées dans d'autres contextes. Notamment, la résolution temporelle nécessaire peut varier d'une application à l'autre et changer au cours du temps. Une échelle appropriée à un certain moment pourra devenir excessive, notamment lorsque le volume de données aura augmenté. La scalabilité permet de conserver le même format de métadonnées dans tous ces contextes : cela évite d'avoir à calculer de nouveaux index pour des données qui sont peut-être temporairement inaccessibles ou tout simplement en très grand nombre donc coûteuses à réindexer. Dans ces moments, si les descripteurs sont scalables, il suffira d'appliquer la bonne opération de remise à l'échelle aux métadonnées, sans modifier le format et sans avoir besoin d'accéder aux données complètes. Ainsi, la scalabilité prolonge la vie des métadonnées et, sous condition d'une bonne standardisation, aide à l'interopérabilité des systèmes.

Evidemment, la scalabilité ne peut résoudre tous les problèmes puisque le passage à une résolution inférieure est irréversible et entraîne une perte d'information. C'est le prix de la concision. Cependant, les descriptions à basse résolution peuvent être considérées comme des approximations de descriptions plus fines et demeurer utiles aux applications, même si elles n'en remplissent pas toutes les exigences.

2.1.2 Définition

Deux propriétés fondamentales caractérisent la scalabilité. Tout d'abord, le format des métadonnées doit permettre leur instantiation à n'importe quelle résolution temporelle. Deuxièmement, une description existante doit pouvoir être automatiquement convertie vers une description de résolution plus faible. Le résultat ne doit dépendre que de la résolution finale, et pas des opérations intermédiaires de remise à l'échelle ayant permis de le produire.

Evidemment, les opérations de remise à l'échelle causent une perte d'information : si l'on veut une description plus concise, remplissant les contraintes de stockage, de bande passante ou d'ergonomie, c'est inévitable.

La scalabilité doit nécessairement s'accompagner d'opérations de conversion d'une échelle à l'autre. Un grand nombre d'opérations mathématiques conviennent : minimum et maximum, somme, moyenne (éventuellement pondérée), variance, covariance, histogramme...

Les métadonnées scalables consistent en deux types de données : les valeurs des descripteurs, résumées par une ou plusieurs des opérations pré-citées, accompagnées d'un petit nombre d'informations assurant l'autonomie de la structure pour une utilisation ultérieure, comme la fréquence d'échantillonnage initiale, le rapport de mise à l'échelle et l'opération appliquée...

2.2 Descripteurs scalables

2.2.1 Opérations de remise à l'échelle

Les opérations de remise à l'échelle ont pour but de convertir une description à une échelle donnée en une autre de résolution plus faible. La description résultante résume de façon plus concise, et donc inévitablement plus grossière, les données de départ.

Supposons que la description de résolution maximale contient N valeurs du descripteur choisi. Cet ensemble est partitionné en sous-ensemble de N_i valeurs chacun, et chacun des sous-ensembles est résumé en une seule valeur, obtenue en appliquant aux N_i valeurs l'opération de mise à l'échelle. L'indexation par le contenu produisant généralement des séries chronologiques, il est utile et commode de conserver cette structure lors de la remise à l'échelle ; en conséquence, les sous-ensembles de la partition seront contigus.

Les opérations peuvent consister en un résumé statistique comme la moyenne ou la variance, ou en un sous-échantillonnage des valeurs (déterministe ou aléatoire).

La plus élémentaire des opérations de remise à l'échelle est la somme, qui résume M valeurs x_i par :

$$s = \sum_{i=1}^{M} x_i$$

M est le facteur de mise à l'échelle (ou *scale ratio*). La description est d'autant plus concise que M est grand, et l'on peut choisir une résolution arbitrairement grande. D'autre part, l'associativité de la somme garantit de pouvoir obtenir la description à la résolution souhaitée en une ou plusieurs étapes. Ainsi, les descripteurs mis à l'échelle par l'opération de somme sont bien scalables au sens défini précédemment.

Plusieurs variantes s'apparentent à ce premier exemple : la somme pondérée, la moyenne et la moyenne pondérée. La moyenne pondérée, par exemple, s'obtient par :

$$m = \frac{\sum_{i=1}^{M} \omega_i x_i}{\sum_{i=1}^{M} \omega_i}$$

L'ajout de poids permet de moduler la contribution de chaque valeur, par exemple en fonction de sa fiabilité. Dans ce cas, les poids devront être stockés en même

temps que les valeurs, pour permettre une utilisation ultérieure et garantir la scalabilité de la structure.

Des descriptions plus raffinées comme la variance (ou la covariance pour les descripteurs vectoriels) peuvent être ajoutées à la moyenne. Les variances pondérées de sous-ensembles de valeurs scalaires :

$$v = \frac{\sum_{i=1}^{M} \omega_i (x_i - m)^2}{\sum_{i=1}^{K} \omega_i}$$

seront remises à l'échelle par l'opération :

$$v = \frac{\sum_{i=1}^{M} \omega_i v_i + \sum_{i=1}^{M} \omega_i (m_i - m)^2}{\sum_{i=1}^{M} \omega_i}$$

L'opération de passage à l'histogramme consiste à *compter* le nombre de descripteurs tombant dans certaines fourchettes de valeurs ou classes prédéfinies. Si les descripteurs de départ évoluent dans un espace qu'on a partitionné en N classes, la k^e valeur de l'histogramme décrivant un horizon temporel de M descripteurs sera donné par :

$$h(k) = card\,\{x_i \in C_k : i \in \{0, ..., M-1\}\}$$

C_k étant la k^e classe de la partition. La partition est obtenue par quantification (scalaire ou vectorielle) ou un clustering de l'espace initial. Une fois cette description obtenue, les passages à l'échelle ultérieurs se feront par simple somme des histogrammes contigus.

Toutes ces opérations sont de nature statistique et utilisent toute l'information disponible à l'échelle de départ. D'autres opérations relèvent du sous-échantillonnage. Une séquence peut par exemple être résumée par sa valeur maximale et/ou minimale, sa première ou sa dernière valeur. Une telle décimation est déterministe. On peut également réaliser une décimation aléatoire, la séquence étant résumée par une de ses valeurs prise au hasard. L'accent est alors mis sur les propriétés statistiques de l'ensemble des séquences. Toutes ces descriptions sont scalables.

D'autres exemples de mise à l'échelle sont présentés dans [9, 12, 10, 11, 19].

La plupart des descripteurs se présentent sous la forme de séries chronologiques. Cette structure est conservée lors des remises à l'échelle. Le ratio de mise à l'échelle peut être uniforme (on résume des sous-séquences ayant toutes la même taille) ou offrir une résolution non-uniforme. Par exemple, on pourrait choisir d'appliquer l'opération de min-max avec un rapport de mise à l'échelle plus grand là où le signal varie lentement, et plus faible (meilleure résolution) aux endroits de fortes variations.

La variété et le nombre d'opérations possibles permettent une grande souplesse dans les combinaisons, suivant le descripteur associé et l'application visée. Plusieurs opérations et plusieurs résolutions peuvent être stockées simultanément pour un même descripteur.

2.2.2 Exemples de descripteurs scalables

La section 1.4.2 présente des exemples de descripteurs. On va détailler ici quelques exemples particuliers de descripteurs *scalables* issus de [19], en montrant quelles opérations de remise à l'échelle leur sont adaptées et pour quel usage. La table 2.1 résume ces exemples.

Soulignons le fait que deux cas peuvent se produire : parfois, le descripteur conservera sa sémantique lors du passage à l'échelle. C'est par exemple le cas de la fréquence fondamentale. Dans d'autres cas, la sémantique est perdue mais une autre vient la remplacer, le descripteur mis à l'échelle pouvant alors s'interpréter comme la paramétrisation d'une distribution.

TAB. 2.1 – Exemples de descripteurs scalables.

Descripteur	**Opérations typiques**	**Utilité**
Forme d'onde	Min, Max	Affichage, recherche
Puissance	Moyenne, variance	Recherche
Fréquence fondamentale	Moyenne pondérée Histogrammes	Query-by-humming
Spectre de puissance	Moyenne, covariance Histogrammes	Affichage, recherche

Forme d'onde

Il peut paraître étonnant de parler de métadonnées quand les descripteurs sont les données elles-mêmes. Toutefois, ce choix fait sens lors du passage à l'échelle. Résumés par les opérations de minimum et maximum, ils permettent un affichage rapide et économique de la forme d'onde, sans avoir à accéder directement aux fichiers audio, potentiellement très volumineux. En effet, étant donné le nombre limité de pixels disponibles sur l'écran, il est lent et inutile de charger les valeurs de tous les échantillons, qui ne pourront pas tous être affichés : il suffit d'une paire minimum-maximum par pixel, l'affichage consistant à tracer un segment vertical limité par ces deux valeurs, à chaque pixel.

La description de la forme d'onde par les minimum et maximum peut également être le support d'une comparaison rapide entre formes d'onde.

Puissance moyenne

La définition de la puissance moyenne la rend homogène pour l'opération de moyenne. Ainsi, ces descripteurs, remis à l'échelle par le calcul de la moyenne, conservent une sémantique constante quelle que que soit la résolution temporelle.

Fréquence fondamentale

La fréquence fondamentale est un assez bon indicateur de la sensation de *hauteur* musicale et présente donc des intérêts pour de nombreuses applications de recherche, de classification, d'affichage ; elle constitue une description intermédiaire entre le contenu lui-même et des descriptions mélodiques haut-niveau comme la norme MIDI. Sa mise à l'échelle par des histogrammes peut par exemple donner une information précieuse sur la tonalité du morceau (dans la cas de la musique tonale).

Cependant, elle n'est pas toujours définie à chaque instant du signal, c'est pourquoi les opérations de remise à l'échelle (moyenne, histogrammes, min et max...) qu'on lui appliquera devront nécessairement être *pondérées*. Un poids nul affecté aux valeurs peu fiables permettra de ne pas corrompre la description. La détermination des poids initiaux se fera en même temps que l'extraction de la fréquence fondamentale elle-même.

Spectre

Le spectre de puissance conserve sa sémantique lors du passage à la moyenne, à condition qu'il n'ait pas subi de transformation comme le passage au logarithme ou à la racine, la transformation cepstrale ou des transformations linéaires comme l'analyse en composantes principales. Dans le cas contraire, la description par la moyenne et la (co)variance sera cependant utile, par exemple pour paramétrer un modèle statistique de la distribution des valeurs du spectre.

Autres

De très nombreux descripteurs comme l'harmonicité, la modulation, le centroïde spectral, l'étalement spectral... sont également scalables. De manière peut-être plus inattendue, des descripteurs ne décrivant pas des informations continues dans le temps, comme des descripteurs «d'événements» pourront être mis à l'échelle grâce à l'opération d'histogrammes.

2.2.3 Structure de données

Elle doit répondre à deux contraintes fondamentales : la flexibilité (ajout simple de nouveaux descripteurs ou de nouvelles opérations de remise à l'échelle) et l'interopérabilité (possibilité de réutilisation dans d'autres contextes et pour d'autres applications, sans autre information que celle contenue dans la structure de données).

Ainsi, une structure de données scalables réunit à la fois les données mises à l'échelle (sous la forme de séries chronologiques de scalaires, vecteurs ou matrices) et des informations complétant ces données : quelle opération de mise à l'échelle a été (ou doit être) utilisée, le facteur de mise à l'échelle, la fréquence

d'échantillonnage initiale... Cette structure est donc parfaitement autonome. Les systèmes de stockage et de transport pourront, suivant les contraintes de capacité et de bande-passante, ajuster leur résolution. La norme MPEG-7 [19] spécifie quelles sont les données à ajouter suivant les descripteurs en question.

La structure est un «conteneur» pour les données. Elle peut recevoir différents descripteurs, à différentes échelles, ayant subi différentes opérations de remise à l'échelle. En particulier, les différentes échelles d'un même descripteur pourront être organisées de manière hiérarchique, ce qui présente un avantage certain notamment pour les algorithmes de recherche.

Chapitre 3

Algorithmes de recherche dans des bases de données sonores

3.1 Introduction

L'indexation automatique et la recherche d'information musicale (*Music Information Retrieval*) qui l'utilise requièrent le développement d'algorithmes mathématiques permettant d'extraire automatiquement l'information à partir d'un enregistrement numérique, puis de comparer efficacement un nouvel extrait à la base de départ. [33], [36] et [13] offrent des revues détaillées de ces enjeux et des solutions envisageables.

Une fonction essentielle de tout système manipulant de grandes quantités de données est la recherche, et si l'on ne prête pas attention à l'efficacité des algorithmes et à leurs propriétés asymptotiques, le coût des opérations peut devenir rapidement prohibitif. Des algorithmes efficaces ont été mis au point par exemple pour le traitement de données textuelles ou pour des données organisées en arbres, mais il en existe moins pour les données multimédia et notamment sonores. Quelques exemples sont donnés ici.

Il est important de souligner qu'il existe une différence notable entre les algorithmes de recherche à l'identique (on cherche à identifier et à localiser l'occurrence d'un extrait donné dans une base) et les algorithmes de recherche de similarités (étant donné un extrait, on recherche un contenu similaire selon une certaine métrique).

3.2 Recherche exhaustive

L'idée la plus naturelle et la plus naïve est d'effectuer des comparaisons exhaustives du signal recherché avec toute la base de recherche. On peut ainsi imaginer comparer directement les formes d'onde, ou de calculer des intercorrélations entre signaux.

Cette idée est naïve : le coût d'une telle comparaison devient très rapidement prohibitif. Son seul intérêt est d'offrir une borne supérieure à la complexité des algorithmes de recherche : on ne doit pas faire pire.

3.3 Quelques exemples

Contours mélodiques

[3] et [1] donnent plusieurs exemples de recherche basée sur l'extraction d'un contour mélodique (*pitch contour*). L'application majeure de ces systèmes est la recherche par chantonnement (*query-by-humming*) ; le contour mélodique vu comme caractéristique extraite du signal est bien adapté à la forme particulière de la requête.

Dans [1], l'algorithme présenté repose sur deux étapes : une première consiste à extraire les hauteurs et les convertir en un contour mélodique lissé ; ensuite, la phase de recherche s'appuie sur des algorithmes d'alignement inspirés du traitement de la parole, comme la programmation dynamique (*Dynamic Time Warping* ou *DTW*).

Histogrammes de hauteur

L'algorithme présenté dans [34] s'appuie sur deux types d'histogrammes de hauteurs, pour des applications de classification par genre musical. Il s'applique à la fois à des fichiers au format MIDI (pour lesquels l'étape de pré-traitement est triviale) et à du signal audio (auquel des algorithmes d'estimation de fréquences fondamentales sont appliqués). Une fois les fréquences extraites, les valeurs sont accumulées dans deux types d'histogrammes :

- un histogramme brut, quantifié sur 128 valeurs (correspondant aux notes définies dans MIDI) ;
- un histogramme replié (*folded*) dans lequel les valeurs sont ramenées dans une octave et ordonnées selon le cycle des quintes.

Les histogrammes sont normalisés par le nombre total de valeurs qu'ils résument. Quatre quantités sont ensuite dérivées à partir de ces histogrammes :

- le maximum de l'histogramme replié, qui correspond à la note la plus fréquente (en général, la tonique dans un morceau tonal simple) ;
- le nombre d'occurrences de cette note ;
- le maximum de l'histogramme non-replié, qui apporte une information sur la tessiture majoritaire des instruments du morceau ;
- la distance (nombre de demi-tons) entre les deux maxima de l'histogramme replié.

Ces quatre quantités sont ensuite utilisées comme paramètres pour entraîner un modèle de classification supervisé, par l'algorithme des k-plus-proches-voisins. L'idée intéressante ici est d'utiliser les histogrammes comme représentatifs de la

distribution des hauteurs tonales dans le morceau, et donc de la répartition des tonalités. Le lien entre tonalités et genre musical s'appuie sur des considérations musicologiques simples : par exemple dans le Jazz, les modulations vers d'autres tonalités sont très fréquentes et on peut s'attendre à un maximum d'amplitude relativement faible ; dans des morceaux de structure harmonique simple, la distance entre les deux maxima sera le plus souvent d'une quinte ou une quarte (relation tonique-dominante, ou modulation au ton voisin). La matrice de confusion, nettement au-dessus du hasard, obtenue dans [34] confirme l'utilité de cette approche tout en soulignant qu'elle n'est pas suffisante. [1] cite également un exemple de recherche par histogrammes de hauteurs.

D'autres exemples

[27] propose deux méthodes de recherche, l'une utilisant des spectres, l'autre la dérivée de la puissance du signal. De tels algorithmes choisissent de se focaliser sur un type de descripteur.

[26] présente une étude systématique de 4 sous-ensembles de descripteurs balayant un grand nombre de possibilités : propriétés très bas niveau du signal (puissance RMS, largeur de bande), MFCC, attributs psychoacoustiques (rugosité, sonie) et enfin un modèle auditif des fluctuations d'enveloppe temporelle du signal.

Une autre approche consiste plutôt à extraire un grand nombre de descripteurs, et à les agréger par diverses méthodes : sélection automatique des plus significatifs [30], combinaisons par tranformations de type ACP en un petit nombre de descripteurs dérivés maximisant l'information transmise [2]...

3.4 Recherche active

L'algorithme de recherche active, décrit sous différentes variantes dans [32, 21, 20] est l'algorithme de référence pour la suite de ce travail.

Cet algorithme permet de localiser un court extrait (de quelques secondes) dans un flux audio ou une base de données de documents sonores. Il s'appuie sur une série chronologique d'histogrammes construits à partir du spectre de puissance du signal. Le principe général, illustré figure 3.1, repose sur une mesure de similarité entre l'histogramme résumant la requête, et des histogrammes de même taille indexant la base de recherche.

La base de recherche est préalablement indexée par le calcul du spectre de puissance du signal sur des trames régulièrement espacées. Le spectre est ramené à une échelle logarithmique de fréquences, puis quantifié suivant un dictionnaire précalculé. La base est ainsi indexée par une série chronologique d'indices renvoyant aux mots correspondants dans le dictionnaire. Lors de la phase de recherche, ces

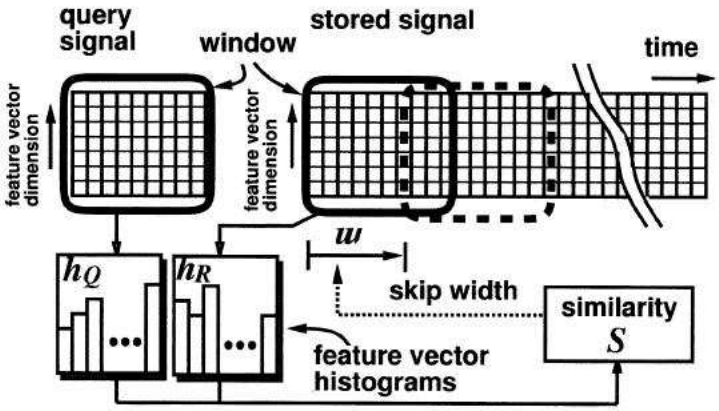

FIG. 3.1 – Schéma-bloc de l'algorithme d'active-search. Extrait de [21]

indices sont accumulés dans des histogrammes. L'histogramme de chaque trame est comparé à celui de la requête jusqu'à la détection de la présence de la requête.

L'efficacité de la recherche repose sur le fait que, après chaque comparaison «ratée», l'algorithme peut faire un «saut en avant» (en anglais *skip*) d'un nombre d'échantillons correspondant au mismatch entre les deux histogrammes. En effet, s'il manque N indices pour que les histogrammes soient similaires, il est inutile de comparer la requête avec un histogramme décalé de moins de N échantillons.

Cette description correspond au cas idéal où la requête est présente exactement à l'identique dans la base, et notamment que son début correspond effectivement au début d'une fenêtre d'analyse du spectre de la phase d'indexation. Un désalignement des fenêtres d'analyse entre la requête et la base introduit une erreur lors de la quantification vectorielle. Afin de prendre en compte ce bruit, on introduit une mesure de similarité entre histogrammes, définie par :

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

où $h_R(k)$ est le k^e coefficient de l'histogramme décrivant la requête, $h_C(k)$ le même coefficient dans l'histogramme courant, et L la taille du dictionnaire de quantification vectorielle (donc le nombre de coefficients dans chaque histogramme). On considère la requête localisée lorsque cette mesure dépasse un certain seuil θ. Entre deux échantillons, la similarité ne s'accroîtra au maximum que de :

$$\Delta S_{max} = S(h_R, h_C(n_2)) - S(h_R, h_C(n_1)) = \frac{n_2 - n_1}{L}$$

Le nombre d'échantillons que l'on peut négliger est alors donné par :

$$w = \begin{cases} E[L(\theta - S)] + 1 & \text{si } S < \theta \\ 1 & \text{sinon} \end{cases}$$

Cet algorithme a été choisi comme point de départ car l'histogramme est une représentation scalable : une description à une échelle plus grossière est obtenue par simple somme de la suite chronologique des histogrammes à échelle plus fine. La similarité entre algorithmes basés sur des histogrammes facilite l'évaluation comparative de l'algorithme mis au point dans la suite de ce travail. Il s'agit de plus d'une bonne référence pour l'évaluation, car il représente l'état de l'art.

Chapitre 4

L'algorithme de recherche hiérarchique

4.1 Principe de la recherche hiérarchique

Cette section décrit un algorithme de recherche rapide appuyée sur des métadonnées scalables. La scalabilité est au départ une contrainte que l'on s'impose, sur le pari qu'elle est indispensable pour épouser la croissance des données et qu'à terme, seules les métadonnées scalables seront utiles. Mon objectif initial était de montrer que la scalabilité est compatible avec les exigences d'une recherche efficace. Cela m'a amenée sur la piste d'un algorithme nouveau, d'une efficacité bien supérieure à notre algorithme de référence, la recherche active.

Tous les algorithmes efficaces de recherche reposent sur le principe d'élagage (*pruning*), celui-ci devant être réalisé le plus tôt possible, tout en garantissant au maximum de ne pas éliminer le résultat attendu. Un bon moyen d'effectuer cet élagage est d'organiser l'espace de recherche hiérarchiquement, sous la forme d'un arbre, dans lequel chaque nœud contient suffisamment d'information pour déterminer si la requête peut être ou n'est pas dans le sous-arbre dont la racine est le nœud examiné. Les métadonnées scalables permettent de construire une telle structure, les données attachées à chaque nœud étant obtenues en passant à l'échelle les descriptions de ses fils. La recherche commence à la racine de l'arbre (à la résolution la plus grossière) et à chaque nœud, on détermine grâce à la description qui lui est attachée si le sous-arbre correspondant peut être élagué ; la recherche continue sur les fils des nœuds compatibles avec la présence de la requête, jusqu'aux feuilles. La recherche est rapide près de la racine et précise près des feuilles.

L'algorithme est ici appliqué à des histogrammes de descripteurs spectraux codés par quantification vectorielle. Il est important de souligner qu'il s'applique aussi bien à tout autre descripteur codé par histogramme (fréquence fondamentale, centroïde spectral, etc.), voire à des histogrammes de classes obtenues par ap-

prentissage statistique (k-plus-proches-voisins, machines à vecteurs de support...).

4.2 Recherche hiérarchique basée sur les histogrammes

L'algorithme mis au point lors de ce stage utilise une hiérarchie d'*histogrammes* résumant la distribution des spectres de puissance des signaux. Les premières étapes de l'indexation sont similaires à l'algorithme de recherche active présenté précédemment. Les spectres de puissance subissent une quantification vectorielle puis sont accumulés dans une série chronologique d'histogrammes. Cette série est ensuite mise à l'échelle par facteurs 2 successifs, formant ainsi un arbre binaire qui forme l'index de recherche. Lors de la recherche, un histogramme similaire est calculé à partir du signal-requête et comparé successivement à chaque noeud, depuis la racine. La comparaison des histogrammes permet de déterminer si la requête est *incluse* dans le segment résumé par le nœud. En effet, si pour la trame j et pour chaque vecteur k du dictionnaire, on a :

$$h_R(k) \leq h_C(k, j)$$

alors tous les spectres contenus dans la requête sont également contenus dans le segment résumé par l'histogramme qu'on examine. Dans ce cas, l'opération est répétée en descendant dans l'arbre, jusqu'à ce que le test échoue ou qu'on localise la requête. En revanche, si la relation n'est pas vérifié, alors la requête n'est pas contenue dans le segment et cette partie de la base peut être élaguée.

En pratique, l'algorithme est un peu plus compliqué, car il n'est pas du tout certain que la requête soit *alignée* avec un intervalle de la base de même longueur, résumé par un seul nœud. Pour permettre un alignement arbitraire, la comparaison est remplacée par :

$$h_Q(k) \leq h_C(k, j) + h_C(k, j + 1)$$

ce qui permet à la requête d'être «à cheval» sur deux nœuds.

Un autre problème provient du possible non-alignement des trames d'analyse de la requête et de la base. Une réponse à ce problème est proposée en 4.6.2.

La figure 4.3 résume le déroulement de l'algorithme.

4.3 Réalisation

4.3.1 Indexation

Cette étape concerne aussi bien l'extrait recherché que la base de données entière. C'est la phase d'indexation. Elle se déroule en quatre étapes : l'extraction

des spectres, la quantification vectorielle, l'accumulation de ces valeurs dans des histogrammes et le stockage de la structure de données d'index.

L'extraction des spectres se déroule suivant les prescriptions du MPEG-7. En vue de la quantification vectorielle, on doit se limiter à un nombre de bandes de fréquences réduit. Le choix de 8 bandes de fréquence logarithmiquement espacées entre 62.5 et 16 000 Hz offre un bon compromis entre le pouvoir de discrimination et la concision. L'échelle logarithmique de fréquences, qui approche grossièrement les propriétés de l'oreille, permet une distribution de l'énergie relativement homogène entre les différentes bandes de fréquence. L'application d'une racine cubique aux valeurs obtenues permet d'obtenir une meilleure distribution des valeurs dans l'espace (sans toutefois éliminer toute corrélation).

Afin de pouvoir être accumulés dans des histogrammes, ces valeurs doivent être regroupés en catégories, de manière à pouvoir compter le nombre de vecteurs qui tombent dans chacune d'elles. Une façon de réaliser cela est la *quantification vectorielle*. Un dictionnaire de valeurs, chacun représentatif d'une classe, est déterminé selon un algorithme d'apprentissage non supervisé comme l'algorithme dit «LBG»[24]. Ensuite, chaque spectre est remplacé par le numéro du vecteur du dictionnaire qui est le plus proche (selon une distance euclidienne dans le cas du LBG).

Dans notre implémentation, le dictionnaire contient 512 vecteurs, appris sur environ 300.000 vecteurs de spectre extraits aléatoirement de la base de données totale. La taille du dictionnaire réalise un bon compromis entre la complexité et la précision, puisqu'il faut représenter un espace à 8 dimensions.

A ce stade, on peut s'interroger sur les propriétés des codes en tant que descripteurs audio. Ils présentent un certain nombre d'avantages : d'une part, ils réduisent significativement la dimension de l'espace (un vecteur, le spectre, étant représenté par un scalaire, le code correspondant). D'autre part (cela dépend de la taille du dictionnaire) ils permettent une bonne discrimination entre les morceaux. La figure 4.1 illustre cette propriété : étant donné un extrait de la base décrit par une séquence de 600 codes, on mesure le pourcentage de codes communs entre cette séquence et des séquences successives de même longueur de la base. On constate un pic lors de l'occurrence de l'extrait, le pourcentage moyen restant très faible en dehors de ce pic. La discrimination du descripteur est donc très bonne[1].

La suite chronologique des codes est ensuite accumulée dans une structure binaire d'histogrammes de plus en plus longs. Pour fixer les idées, imaginons une

[1] Ce résultat est relativement prévisible. En effet, l'algorithme LBG étant construit pour que les spectres soient aussi uniformément distribués que possible dans les classes, deux codes pris au hasard ont une probabilité d'environ $p = 1/512$ d'être égaux. Deux séquences aléatoires de même longueur L ont donc une probabilité extrêmement faible d'être égales : $P = p^L$. Ce n'est évidemment qu'une première approximation puisque dans une séquence de codes extraite de musique, les codes consécutifs sont fortement corrélés.

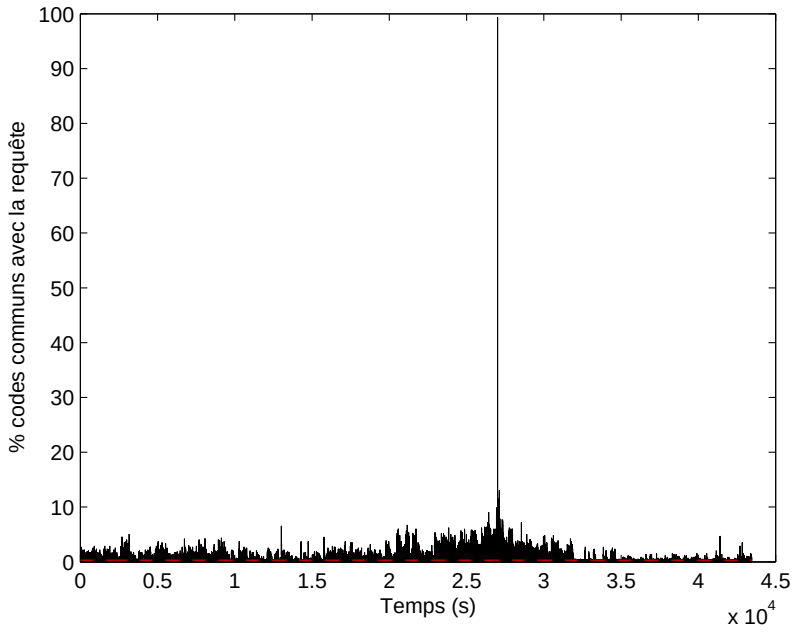

FIG. 4.1 – Similarité d'une séquence avec des séquences de même longueur de la base.

séquence de valeurs quantifiées sur 4 niveaux, valant [1 2 1 4 3 1 2 2]. Elle va être indexée comme sur la figure 4.2 par un arbre binaire d'histogrammes. Au niveau de la racine, un seul histogramme résume les 8 valeurs, puis 2 histogrammes résument chacun 4 valeurs, etc..

En pratique, la structure finale est légèrement différente. Tout d'abord, pour des raisons de concision, tous les niveaux ne sont pas stockés, mais seulement un nombre réduit à partir de la racine. Ensuite, les histogrammes ne sont pas effectivement stockés sous la forme d'un arbre, mais à la suite les uns des autres dans une seule matrice. Enfin, la longueur de la séquence à indexer étant rarement une puissance exacte de 2, l'arbre ne sera que quasi-binaire. Le nombre (et la position dans la matrice) des histogrammes d'une profondeur donnée peuvent être facilement recalculés à partir de la longueur totale de la séquence. D'autre part, puisque l'arbre est tronqué, on stocke également la séquence «brute» de manière à pouvoir recalculer dynamiquement les niveaux manquants. (Dans un système réel en ligne, cela pourrait correspondre à récupérer à la demande sur un serveur les niveaux manquants). Dans notre exemple simpliste, l'index de la

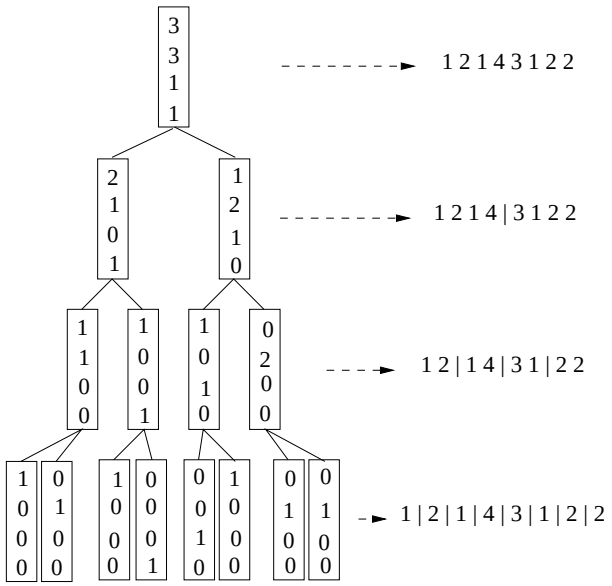

FIG. 4.2 – Arbre binaire d'histogrammes décrivant la séquence 1-2-1-4-3-1-2-2.

séquence sera finalement :

$$\begin{bmatrix} 2 & 1 & 3 \\ 1 & 2 & 3 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

Une fois cette indexation réalisée, la recherche peut avoir lieu. Le choix de la profondeur de troncature (profondeur à partir de laquelle les histogrammes ne sont plus disponibles) peut être fait en accord avec la taille de la base de données et des contraintes de réduction de place en mémoire que l'on s'impose. Une solution informatique pourrait être de tronquer relativement tôt la série fournie à l'utilisateur, les niveaux de résolution plus fine étant stockés sur un serveur et fournis à la demande seulement lorsque nécessaire.

4.3.2 Recherche multi-échelles

A chaque profondeur dans l'arbre de recherche, on a une liste de paires d'histogrammes compatibles avec la présence de la requête. Soient $h(k, j)$ et $h(k, j+1)$ deux histogrammes de même profondeur et temporellement adjacents. Sauf éventuellement à la fin de la base, chacun de ces deux histogrammes possède deux fils $h(k+1, 2j-1)$ et $h(k+1, 2j)$ pour le premier, et $h(k+1, 2j+1)$ et $h(k+1, 2j+2)$ pour le second. Les deux paires à tester au niveau suivant sont alors : $h(k+1, 2j-1) + h(k+1, 2j)$ et $h(k+1, 2j) + h(k+1, 2j+1)$. (Il est

inutile de tester la paire $h(k+1, 2j+1) + h(k+1, 2j+2)$ car si la requête s'y trouve, la paire $h(k, j+1) + h(k, j+2)$ a été retenue à l'étape précédente.) On procède ainsi de la racine vers les feuilles.

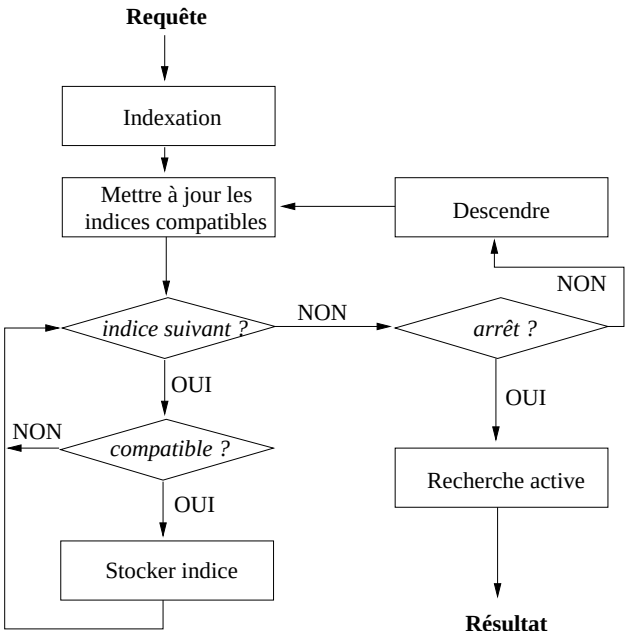

FIG. 4.3 – Schéma-bloc de l'algorithme de recherche scalable.

4.3.3 Phase finale

La phase de recherche hiérarchique se termine lorsque l'une de ces conditions d'arrêt est atteinte :

- tout l'arbre a été élagué, la recherche renvoie un échec (requête non trouvée) ;
- la taille des histogrammes (nombre de codes résumés dans un histogramme) devient inférieure à la moitié de la taille de la requête. Dans ce cas deux histogrammes consécutifs ne peuvent contenir la requête ;
- la profondeur suivante dans l'arbre n'est pas disponible dans l'index.

Dans les deux derniers cas, on pourrait choisir de s'arrêter là. La localisation de l'extrait serait alors imprécise (la résolution de la recherche étant limitée par la résolution maximale disponible dans les index). Si l'on souhaite une localisation précise, on peut alors procéder à une recherche active sur la séquence des codes, mais seulement bien sûr sur les segments non encore éliminés.

4.4 Expérience

Dans cette étude, on s'est limité à la recherche des occurrences *exactes* d'un extrait dans une base de données sonores. Une telle recherche correpond à un grand nombre de cas d'applications : par exemple dans un outil de nettoyage de disque dur (recherche et élimination de doublons dans un but d'économie de mémoire) ou de protection du droit d'auteur (recherche de copies illégales de matériel audio).

4.4.1 Tâche expérimentale

On se donne un court extrait de signal audio. Cet extrait est préalablement indexé, suivant la même procédure que celle qui a servi à indexer toute la base de données. Ensuite, l'algorithme de recherche est appliqué, afin de déterminer la présence et éventuellement la localisation précise de l'extrait cherché dans la base. L'objectif est d'évaluer les performances du système (indexation + algorithme de recherche) du point de vue de la précision et de la rapidité.

La tâche est effectuée sur des bases de données de taille différentes, afin d'étudier l'évolution des performances en fonction de ce paramètre.

Notons que pour la simplicité du format de données et la complétude de l'étude sur la structure binaire, on a choisi de concaténer tous les morceaux de la base en un seul très long flux de données indexé. L'adaptation de ce travail à une base découpée et hiérarchisée est discutée dans la section 4.7.

4.4.2 Base de données

La base de données est la base du RWC (*Real World Computing*), conçue et distribuée dans des buts de recherche. En dehors d'enregistrements de gammes chromatiques jouées sur différents instruments, que nous n'avons pas utilisés, elle réunit environ 24h de morceaux de musique libres de droits, de différents genres, échantillonnés en stéréo à 44.100 Hz. Elle est décrite dans [14, 15]. A partir de cette base, on a extrait un ensemble de bases de taille plus petite afin d'évaluer l'effet de ce paramètre sur le temps de recherche. Dans chaque sous-ensemble tous les genres disponibles (classique, jazz...) sont représentés. La base a été indexée comme décrit dans la section 4.3.1.

4.4.3 Requêtes-test

50 requêtes-test ont été extraites de la base. Elles partagent toutes la même durée (9.6 s) et proviennent de morceaux de tous les genres musicaux disponibles.

4.4.4 Algorithmes de référence

Pour comparaison, la recherche des extraits est également réalisée par l'algorithme de recherche active de [21]. Pour son application on utilise directement les suites de codes stockés dans l'index. Son implémentation utilise les mêmes suites de codes que l'algorithme hiérarchique. C'est un algorithme représentatif de l'état de l'art. On a également implémenté pour comparaison un algorithme simple de recherche exhaustive.

4.5 Résultats

4.5.1 Métriques

Afin de s'affranchir d'une dépendance sur l'implémentation, le coût de la recherche est mesuré en nombre de comparaisons d'histogrammes évaluées au décours de la recherche. Dans le cas de l'algorithme de recherche active, il s'agit d'intersections d'histogrammes, et dans le cas de la recherche hiérarchique, de comparaisons entre l'histogramme requête et une paire d'histogrammes de l'arbre de recherche. Le coût est moyenné sur les 50 extraits et calculé pour chacune des 6 bases.

D'autre part, on doit garantir l'efficacité de l'algorithme par des mesures de faux positifs et faux négatifs. On utilise les deux quantités usuelles :

- la précision définie par :

$$Precision = \frac{nombre\ de\ résultats\ corrects}{nombre\ total\ de\ résultats\ renvoyés}$$

  qui donne une mesure du nombre de faux positifs (extraits mal localisés) ;
- le taux de rappel (*recall*) :

$$Recall = \frac{nombre\ de\ résultats\ corrects}{nombre\ de\ résultats\ attendus}$$

  qui traduit le nombre de faux négatifs (extraits présents dans la base et non localisés par l'algorithme)

On a également cherché à évaluer le pouvoir de discrimination des histogrammes en fonction de leur profondeur de recherche. On l'exprime par la rapport :

$$DP = \frac{nombre\ d'histogrammes\ éliminés\ à\ la\ profondeur\ n}{(nombre\ d'histogrammes\ examinés\ à\ la\ profondeur\ n) - 1}$$

ce pourcentage vaut donc 1 si l'étape n de recherche a éliminé tous les histogrammes ne contenant pas la requête.

4.5.2 Vitesse

Avec les mêmes extraits, bases et réglages de seuils que précédemment, on mesure le nombre moyen (sur les 50 extraits) de comparaisons entre histogrammes évaluées au décours de l'algorithme. Les valeurs sont résumées dans la table 4.1 et une visualisation est donnée dans 4.4. L'accélération est le rapport du nombre de similarité évaluées dans la recherche active et du nombre de compatibilités évaluées dans la recherche hiérarchique.

TAB. 4.1 – Vitesse.

Taille de la base	1h	2h	6h	13h	24h	48h
Rech. exhaustive	$2{,}8.10^5$	$4{,}3.10^5$	$1{,}5.10^6$	$2{,}7.10^7$	$5{,}3.10^7$	$10{,}6.10^7$
Rech. active	1077	1394	3628	6169	11832	23665
Rech. hiérarchique	316	354	363	427	548	798
Accélération	3,4	3,9	10	14,5	21,6	29,7

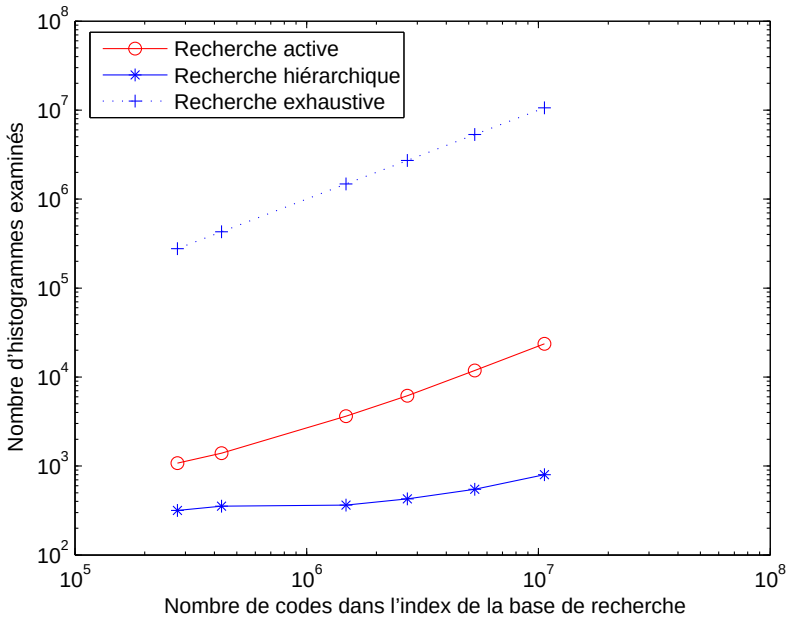

FIG. 4.4 – Vitesse de recherche en fonction de la taille de la base de recherche.

Outre l'accélération importante obtenue, on retiendra surtout un résultat très intéressant quant à la scalabilité de l'algorithme. En effet, on observe un infléchissement dans l'augmentation du nombre de compatibilités évaluées dans l'algorithme de recherche hiérarchique. Alors que les deux algorithmes de référence

(recherche exhaustive et recherche active) poursuivent une croissance linéaire en fonction de la taille de la base, l'algorithme de recherche hiérarchique permet d'obtenir une vitesse croissant moins vite au moins sur la partie centrale de la courbe.

Cependant, la tendance linéaire semble reprendre lorsqu'on atteint les bases testées les plus grandes, ce qui suggère que le gain apporté par la recherche hiérarchique va s'infléchir pour de très grandes bases, tout en ayant visiblement une pente plus faible que celle des algorithmes de référence.

4.5.3 Précision et taux de rappel

Pour les deux algorithmes, on mesure dans des conditions similaires (mêmes extraits et bases) les taux de précision et de rappel moyens. Les valeurs des seuils de détection pour la recherche active et pour la phase finale de recherche hiérarchique sont égales pour une paire extrait-base. Les résultats sont résumés dans la table 4.2.

TAB. 4.2 – Précision et taux de rappel.

Algorithme	**Précision**	**Rappel**
Recherche active	93.7 %	92.8 %
Recherche hiérarchique	86.2 %	97.9 %

Les seuils ont été grossièrement réglés. Un réglage plus fin permettrait sans doute d'obtenir de meilleurs résultats (tels que ceux annoncés dans [32, 21, 20]). Le but était surtout de garantir que les mesures de vitesse ultérieures avaient un sens, donc d'obtenir des résultats de précision et de rappel raisonnables sinon parfaits.

Une explication supplémentaire aux faux négatifs vient de la méthode de choix des extraits. En effet ils sont choisis au hasard dans la séquence concaténée des morceaux de la base de plus petite taille. Si un extrait tombait entre deux morceaux, il est possible que ces deux morceaux ne soient plus adjacents dans les bases de taille supérieure, auquel cas ils n'y sont plus présents. Il s'agit donc d'un simple artefact.

4.6 Analyse des propriétés de l'algorithme

4.6.1 Pouvoir de discrimination

Afin de diagnostiquer plus finement la réelle efficacité de l'algorithme de recherche hiérarchique, on mesure au décours de l'exécution de l'algorithme le pouvoir de discrimination des histogrammes en fonction de leur profondeur dans

l'arbre de recherche, comme défini dans la section 4.5.1. Ceci permet de mieux visualiser quelles sont les étapes les plus efficaces de l'algorithme en terme d'élagage. A des fins de comparaison on évalue également à chaque niveau la discrimination maximale théorique. Elle est définie par le pourcentage qu'on atteindrait si, à un niveau donné, on éliminait tous les histogrammes ne contenant pas la requête. Ces deux quantités (pouvoir de discrimination réel et maximum théorique) sont représentés sur la figure 4.5.

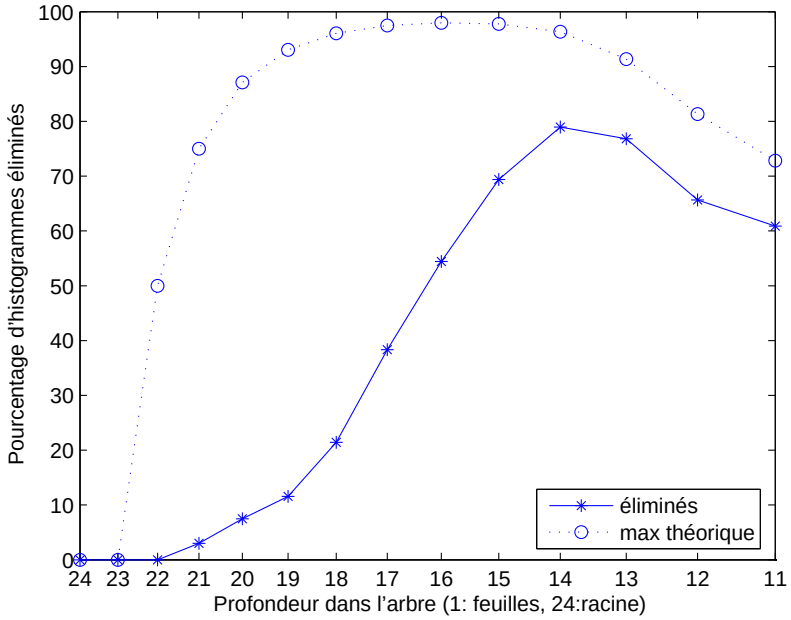

FIG. 4.5 – Pourcentage d'histogrammes éliminés. (Le pourcentage mesure le pouvoir de discrimination à chaque échelle).

On voit que la discrimination est bonne aux niveaux intermédiaires (pour des histogrammes 2 à 32 fois plus grands que la requête, elle dépasse les 50 %). La décroissance aux derniers niveaux traduit le fait que seuls un petit nombre d'histogrammes sont encore en lice ; la mesure par pourcentage induit que le bon histogramme (celui qui contient effectivement la requête) prend relativement plus de poids à mesure que le nombre de candidats se réduit.

En revanche, le pouvoir de discrimination reste assez loin du maximum aux niveaux de recherche les plus près de la racine, ce qui suggère de chercher des améliorations à ces niveaux élevés. Cela peut expliquer l'inflexion de l'accélération de la recherche à partir d'un certain volume de données, observée précédemment.

4.6.2 Influence des seuils

Tous les résultats précédents sont obtenus dans le cas idéal où les extraits sont alignés avec les fenêtres d'analyse initiales dans la base. Cela ne correspond évidemment pas au cas pratique d'application, l'information d'alignement n'étant pas disponible en situation réelle.

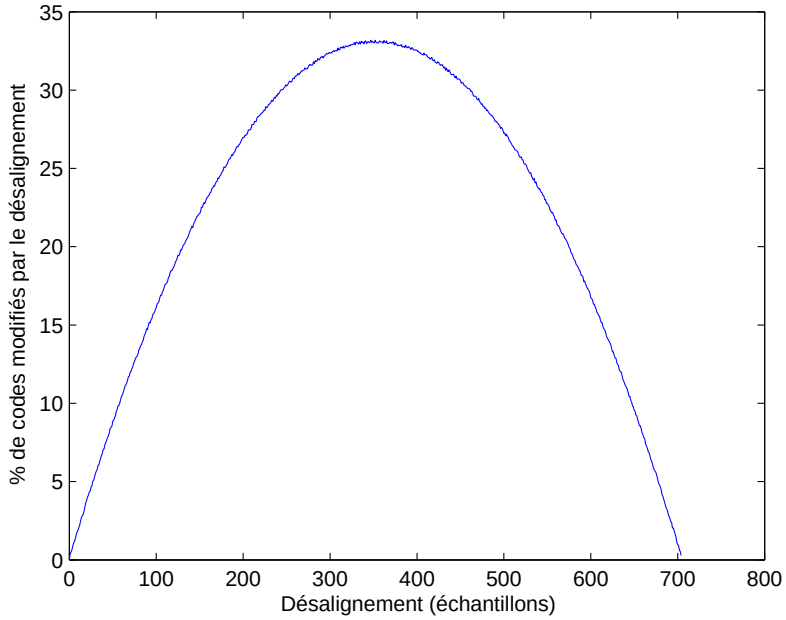

FIG. 4.6 – Effet du désalignement des fenêtres d'analyse sur la quantification. (La taille de la fenêtre est de 700 points.)

Le désalignement des fenêtres d'analyse a un effet important sur la phase d'indexation (extraction des spectres et quantification vectorielle). Cette différence est représentée sur la figure 4.6 qui représente le pourcentage de codes incorrects en fonction du nombre d'échantillons de décalage entre les fenêtres d'analyse, pour une séquence de 600 trames quantifiées. Un code est jugé incorrect s'il n'est égal ni au code de la fenêtre initiale précedente, ni au code suivant. Ce pourcentage de différence atteint plus 30% dans le pire des cas.

Toutefois, la similarité entre l'extrait non aligné et la séquence correspondante dans la base reste suffisamment discriminante, comme le montre la figure 4.7 qui représente des quantités comparables à celles de la figure 4.1, mais pour un extrait fortement non aligné. On peut donc espérer conserver l'algorithme en introduisant un facteur de tolérance dans le calcul de la compatibilité.

Pour adapter l'algorithme à ce cas, on introduit donc un facteur de tolérance f qui traduit la proportion de codes incorrects que l'on accepte pour juger tout

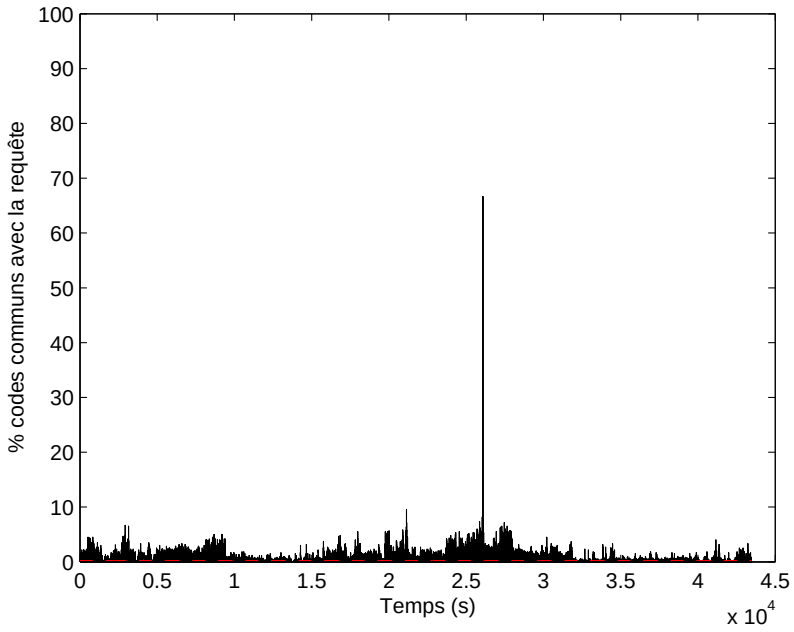

FIG. 4.7 – Similarité d'une séquence non alignée avec des séquences de même longueur de la base.

de même les histogrammes compatibles.

Cette tolérance a évidemment une forte influence sur le déroulement de l'algorithme. En effet, à chaque niveau, cela implique que l'algorithme laisse passer davantage d'histogrammes que dans sa version sans tolérance. Cela a pour effet d'augmenter le nombre d'histogrammes examinés à chaque niveau et cela réduit donc l'accélération observée précédemment. La figure 4.8 illustre ce phénomène pour différentes valeurs du seuil. Sur cette figure, la courbe pointillée représente le nombre total d'histogrammes disponibles dans l'index à la profondeur donnée. On voit que plus le facteur de tolérance augmente et plus on se rapproche de ce maximum. Passé une certaine valeur, on n'observe plus la décroissance qui signifiait que l'on avait éliminé une majorité d'histogrammes.

L'introduction du facteur de tolérance augmente donc le nombre de faux positifs au niveau d'arrêt de la recherche hiérarchique. La phase finale de recherche pourra éventuellement corriger ce problème. En revanche, si la tolérance est trop faible, on risque d'élimine dès la phase de recherche hiérarchique le ou les histogrammes cibles. Ces faux négatifs précoces sont irrécupérables par la phase finale. Il est donc important de choisir une bonne valeur du facteur de tolérance, réalisant un compromis acceptable entre faux positifs et faux négatifs.

La figure 4.9 illustre l'évolution du nombre de ces erreurs à mesure que la tolérance croît. On mesure le nombre total d'erreurs pour 50 extraits de tests, dé-

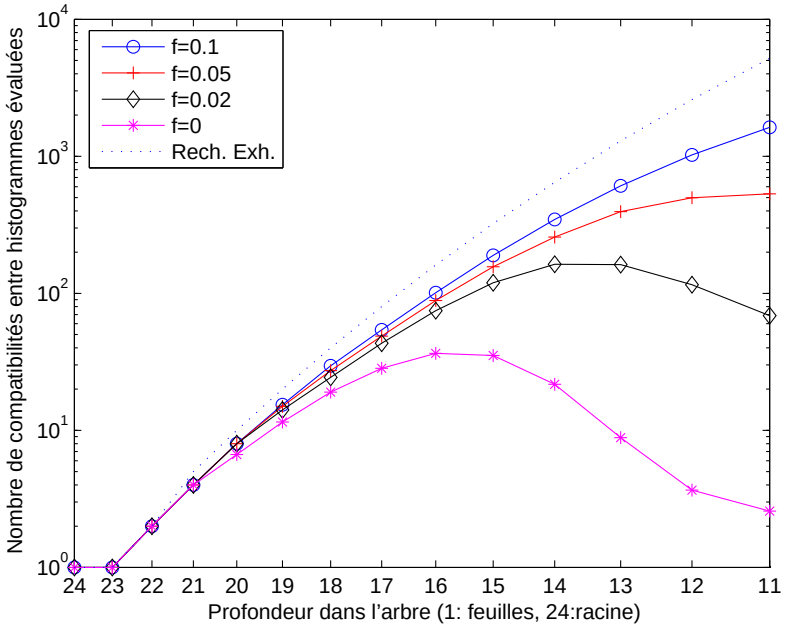

FIG. 4.8 – Influence du seuil sur le nombre de compatibilités évaluées.

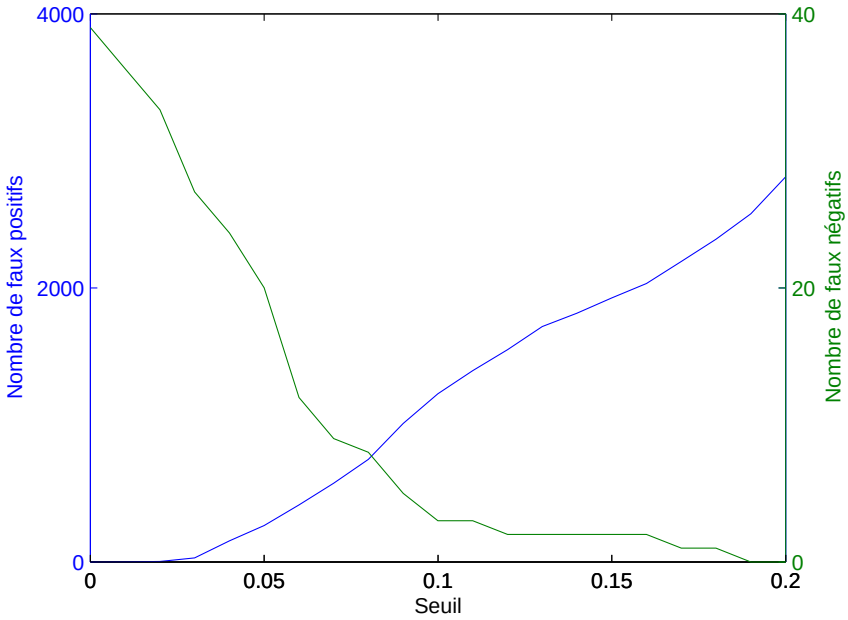

FIG. 4.9 – Influence du seuil sur le nombre d'erreurs.

calés d'une demi fenêtre d'analyse par rapport aux extraits précédemment utilisés. Cette mesure ne prend pas en compte l'effet de la phase finale, mais seulement le nombre de faux positifs et de faux négatifs à l'issue de la dernière passe de l'algorithme hiérarchique. Pour ces extraits très désalignés, il faut aller jusqu'à une très importante tolérance pour éliminer tous les faux négatifs. Le nombre de faux positifs est alors très important et le gain en vitesse obtenu pour les extraits alignés est considérablement réduit.

4.7 Perspectives

Amélioration de l'indexation

Les valeurs décevantes du pouvoir de discrimination des histogrammes aux échelles les plus grandes (cf. section 4.6.1) suggèrent de chercher une amélioration dans cette direction. On peut faire l'hypothèse qu'à ces échelles, des histogrammes similaires sont disséminés dans toute la base, ce qui ne permet pas d'élaguer sélectivement les branches qui les contiennent. Une amélioration possible pourrait être d'envisager une classification (*clustering*) des histogrammes, rendant plus probable un élagage très précoce.

Robustesse

Dans cette étude, les bons résultats obtenus concernent des extraits alignés avec les fenêtres d'analyse initiales. L'introduction d'un facteur de tolérance conduit à une dégradation des performances.

De manière générale, la robustesse de l'algorithme à des distortions diverses devrait être testé pour garantir la possibilité de son application dans des conditions moins restrictives. Plusieurs phases de l'algorithme sont critiques de ce point de vue : l'étape de quantification vectorielle (potentiellement très sensible à des dégradations), et la mesure de compatibilité entre le signal cherché et les extraits de la base.

Bases hiérarchisées

L'algorithme de recherche hiérarchique a été appliqué à des données concaténées et contenues dans un seul fichier, ce qui a permis la construction d'un arbre de recherche binaire. Les bases de données sonores sur disque ou sur le Web sont organisées hiérarchiquement, suivant une arborescence de dossiers et de fichiers. A condition de construire un arbre de recherche non plus binaire mais respectant cette arborescence de fichiers, l'algorithme est toujours applicable.

Généralisation

Une des forces de l'algorithme hiérarchique présenté ici est de pouvoir s'adapter à tout autre descripteur résumable par histogramme, par exemple aux histogrammes de fréquence fondamentale utilisés dans l'algorithme décrit section 3.3.

Conclusion

Au cours de ce stage, j'ai étudié différents systèmes d'indexation et de recherche d'information musicale. Le fil rouge de cette étude a été la scalabilité, définie comme propriété des structures de métadonnées, choisie comme contrainte et dont j'ai exploré les possibilités pour la visualisation et la recherche de données sonores dans de grandes bases.

La scalabilité est un pari : compte-tenu de la croissance exponentielle de la quantité de données sonores disponibles, on mise sur le fait que seules les métadonnées scalables seront viables à grande échelle, les autres devenant inutilisables à cause de la même tendance exponentielle que les données qu'elles décrivent. La scalabilité est également une contrainte : elle impose une structure de données, la définition d'opérations de mise à l'échelle et le transport de données supplémentaires destinées à préserver l'autonomie de cette structure.

Le premier enjeu fut donc de s'assurer que les structures de données scalables pouvaient être le support d'applications de visualisation, ou d'algorithmes de recherche, au même titre que les métadonnées conçues sans préoccupation de scalabilité.

Lors de cette expérience, il s'est avéré non seulement que les métadonnées scalables supportent ces opérations, mais surtout qu'elles permettent une amélioration des performances. L'algorithme de recherche hiérarchique que j'ai mis au point est un exemple d'une telle amélioration. Il utilise un arbre binaire d'histogrammes de spectres quantifiés, chaque profondeur dans l'arbre correspondant à une résolution temporelle différente. Cet algorithme accélère la recherche d'un extrait dans une base jusqu'à un facteur 30 par rapport à un algorithme représentatif de l'état de l'art. Ce facteur d'accélération croît avec la taille de la base de recherche. C'est le genre de propriétés attendues des futurs systèmes de recherche qui devront opérer sur des bases de données encore beaucoup plus importantes. En cela, les résultats obtenus sont très encourageants.

Un article sur le sujet a été soumis à ISMIR'2005 (International Conference on Music Information Retrieval).

Bibliographie

[1] N. Adams. Time series representations for music information retrieval. Technical report, University of Michigan, 2004.

[2] S. Bermejo Sánchez. *Learning with nearest neighbour classifiers. Chap. 10 : Oriented Principal Component Analysis for Feature Extraction.* PhD thesis, 2002.

[3] S. G. Blackburn. *Content Based Retrieval and Navigation of Music Using Melodic Pitch Contours.* PhD thesis, University of Southampton, september 2000.

[4] C. Burges, J. Platt, and S. Jana. Distortion discriminant analysis for audio fingerprinting. In *IEEE Trans. Speech and Audio Processing*, volume 11, pages 165–174, March 2003.

[5] P. Cano, E. Batlle, T. Kalker, and J. Haitsma. A review of algorithms for audio fingerprinting. In *Proceedings of 2002 International Workshop on Multimedia Signal Processing*, St. Thomas, Virgin Islands, 2002.

[6] C. Chen, G. Gagaudakis, and P. Rosin. Similarity-based image browsing. In *Proceedings of the 16th IFIP World Computer Congress. International Conference on Intelligent Information Processing*, pages 206–213, Beijing, China, August 2000.

[7] T. Cribbin and C. Chen. Visual-spatial exploration of thematic spaces : A comparative study of three visualisation models. In *Proc. Electronic Imaging 2001 : Visual Data Exploration and Analysis VIII*, 2001.

[8] A. de Cheveigné and H. Kawahara. Yin, a fundamental frequency estimator for speech and music. *J. Acoust. Soc. Am.*, 111 :1917–1930, 2002.

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

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

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

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

[13] J. S. Downie. Music information retrieval. *Annual Review of Information Science and Technology*, 37 :295–340, 2003.

[14] M. Goto, Hashiguchi H., Nishimura 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.

[15] M. Goto, Hashiguchi H., Nishimura 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.

[16] Y. Guiard, M. Beaudoin-Lafon, and D. Mottet. Navigation as multiscale pointing : extending fitt's model to very high precision tasks. In *Proc. Computer-Human Interface (CHI)*, pages 450–456, 1999.

[17] W.M. Hartmann. *Signals, Sound and Sensation.* AIP Press, 1997.

[18] I. Herman, G. Melanon, and M. S. Marshall. Graph visualization and navigation in information visualization : a survey. *IEEE Trans. on visualization and computer graphics*, 6 :24–44, 2000.

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

[20] 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.

[21] 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.

[22] F. Kurth and M. Clausen. Full-text indexing of very large audio data bases. In *Audio Engineering Society 110th Convention, Convention Paper 5347*, 2001.

[23] Marc Leman. *Dealing with the data flood. Mining data, text and multimedia*, chapter 5.5.2. Musical Audio Mining. Meij, J. (Ed.) STT/Beweton, The Hague, The Netherlands, 2002.

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

[25] M.R. Luettgen and A.S. Willsky. Likelihood calculation for a class of multiscale stochastic models, with application to texture discrimination. *IEEE Transactions on Image Processing*, 4 :194–207, 1995.

[26] M. F. McKinney and J. Breebaart. Features for audio and music classification. In *Proceedings of ISMIR (International Conference on Music Information Retrieval)*, 2003.

[27] T. Narita and M. Sugiyama. Fast music retrieval using spectrum and power information. In *CRAC Workshop*, 2000.

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

[29] F. Pachet. *Encyclopedia of Knowledge Management*, chapter Knowledge Management and Musical Metadata. D. Schwartz, D. Ed. Idea Group, 2005.

[30] G. Peeters and X. Rodet. Automatically selecting signal descriptors for sound classification. In *Proceedings of the 2002 ICMC*, Goteborg (Sweden), 2002.

[31] E. Scheirer. Tempo and beat analysis of acoustic musical signals. *J. Acoust. Soc. Am. (JASA)*, 103 :588–601, 1998.

[32] G. Smith, K. Kashino, and H. Murase. Quick audio retrieval using active search. In *Proc. of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 6, pages 3777–3780, May 1998.

[33] G. Tzanetakis. *Manipulation, analysis and retrieval systems for audio signals.* PhD thesis, Department of Computer Science, Princeton University, 2002.

[34] G. Tzanetakis. Pitch histograms in audio and symbolic music information retrieval. In *Proc. 3rd ISMIR*, pages 31–38, 2002.

[35] E. Wold, T. Blum, D. Keislar, and J. Wheaton. Content-based classification, search and retrieval of audio. In *IEEE Multimedia*, volume 3(3) :27-36, 1996.

[36] O. Wüst. *Database-Driven Systems and Applications for Music Information Management.* PhD thesis, UPF Barcelona, 2003.