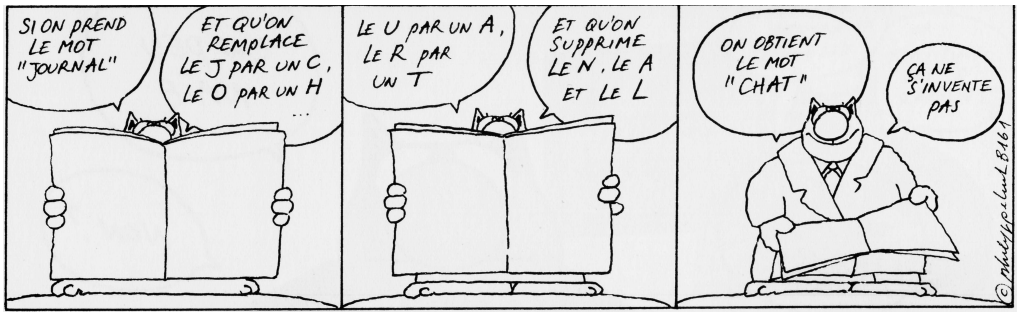
Recherche de Régularités dans les Séquences Musicales Applications à l'Analyse Automatique de Solos de Jazz

Grégoire Carpentier
Mémoire de stage de DEA ATIAM année 2003-2004
Université Paris VI

Laboratoire d'Informatique de Paris VI (Lip6) - Pôle IA
Sous la responsabilité de M. Jean-Gabriel Ganascia
Mars - Septembre 2004


SI ON PREND LE MOT "JOURNAL"
ET QU'ON REMPLACE LE J PAR UN C, LE O PAR UN H ...
LE U PAR UN A, LE R PAR UN T
ET QU'ON SUPPRIME LE N, LE A ET LE L
ON OBTIENT LE MOT "CHAT"
ÇA NE S'INVENTE PAS

Remerciements

Je tiens tout d'abord à remercier les membres du pôle Intelligence Artificielle du LIP6 pour m'avoir accueilli au sein de leur laboratoire. Merci à Nicole Nardy pour sa disponibilité, à Matthieu Delabarre et à Christophe Bouder pour leur patience.

Je remercie évidemment Jean-Gabriel Ganascia, mon responsable de stage, pour son accueil, pour sa volonté d'ouvrir le champ de l'Intelligence Artificielle à des problématiques cognitives et créatrices, et pour m'avoir accordé une grande liberté dans la conduite de mes recherches. Merci également à Pierre-Yves Rolland pour avoir mis à ma disposition son corpus de solos de Charlie Parker.

Merci à l'équipe enseignante du DEA ATIAM pour cette année riche et intense, merci à Cyrille Defaye pour son attention et sa bienveillance, à Philippe Depalle pour avoir accepté et rempli sa mystérieuse mission. Merci également à Denis Matignon, Joël Bensoam et Michel Fano - dont je ne connais hélas, pour les deux derniers, que le son de leurs voix - pour leurs conseils précieux au moment d'intégrer le DEA.

Parmi les membres du corps professoral, je tiens à remercier particulièrement Emmanuel Saint-James pour avoir su, à deux reprises, contenir ma dispersion. J'adresse également ma profonde reconnaissance à Gérard Assayag pour son écoute et pour son soutien tout au long de cette année.

Merci enfin aux étudiants du DEA, pour leurs pluralités, leurs ouvertures, leurs générosités, leurs stoïcismes, leurs humours, leurs désarois, leurs ancrages et leurs évanescences, et merci aux autres, à ceux dont le nom n'a pas sa place ici, mais qui continuent de passer, en silence, sous les mots.

« *-Des champs remplis d'urnes, Bruno. Des tas d'urnes invisibles, enterrées dans un champ immense. Je marchais à travers ce champ et de temps en temps je butais sur quelque chose. Tu vas me dire que j'ai rêvé, naturellement. Tiens, voilà comment ça se passait : de temps en temps, je butais contre une urne, puis, peu à peu, je me suis aperçu que le champ était rempli d'urnes, des milliers et des milliers d'urnes et dans chacune il y avait les cendres d'un mort.* »

Julio Cortazar

Table des matières

Chapitre 1

Enjeux

« Je voudrais un disque de jazz qui ne fasse pas trop *jazz, c'est pour quelqu'un qui* n'aime pas *le jazz. »* - Francis Marmande

Le travail présenté dans ce rapport s'inscrit dans une vaste problématique contemporaine que l'on pourrait ranger sous l'étiquette « Fouille de données » (*data mining*). Cette thématique s'est peu à peu vue occuper une place déterminante dans la recherche informatique qui, face à la déconcertante prolifération de données produites par les machines - au sens Deleuzien du terme - scientifiques, industrielles et commerciales, a trouvé un nouveau champ d'investigation dans la brêche par elle-même entrouverte.

Quels que soient leurs formes ou leurs supports, la quantité de données actuelles, dont le volume double environ tous les vingt mois, ne permet plus une analyse sinon exhaustive, du moins pertinente, entièrement conduite par l'homme seul. Les recherches menées dans le domaine de l'extraction automatique de connaissances dans les bases de données se proposent de tirer profit des capacités calculatoires de l'ordinateur pour tenter de regrouper des données brutes, d'un contenu informatif faible, en structures plus abstraites et plus organisées, en formes - au sens d'une Gestalt -, véhiculant une information de plus haut niveau.

Au delà de la grande diversité de leurs domaines d'application, de la recherche génétique au vaste champ couvert par l'informatique désionnelle en stratégie d'entreprise, ces méthodes possèdent toutes un dénominateur commun, le regroupement de données en catégories. Ces catégories (*clusters*) encapsulent un dégré d'information plus élevé que les entités qui les constituent dans la mesure où celles-ci possèdent toutes, au sein d'un même cluster, un certain degré de ressemblance. C'est ici qu'intervient la notion *d'analyse orientée pattern*, ou *analyse motivique*. Chaque élément d'une catégorie peut être considéré comme la répétition plus ou moins approximative d'un même motif (ou *pattern*) représentant le cluster, encore appelé *prototype*. Il reste toutefois à définir précisément ce que l'on entend par le terme de *pattern*.

L'analyse motivique s'applique en général à des données représentées sous forme séquentielle, comme par exemple les chaînes de caractères. C'est en général le cas de toutes les données à contenu symbolique (par opposition à un contenu visuel ou sonore). On définit un pattern comme un segment, un tronçon d'une chaîne symbolique, c'est-à-dire une succession (au sens d'une relation d'ordre, par exemple temporel) de symboles se répétant, à l'identique ou non, au sein de la chaîne. L'extraction de connaissances à partir de données brutes est donc aussi bien fondée sur la notion de *répétition* que sur celle de *similarité entre les répétitions* dans une séquences de données.

1.1 Musique et Intelligence Artificielle

Parmi les concepts évoqués ci-dessus, certains termes communs à la terminologie musicale devraient nous mettre sur la voie d'une intersection possible entre musique et analyse automatique de données. La notion de *répétition* par exemple est au centre même du processus musical, elle en constitue, sinon le fondement, du moins l'un des caractères majeurs. La musique fait en permanence référence à des structures préétablies, soit dans le passé de l'oeuvre (phrases, thèmes, développements, variations, reprises), soit dans le savoir musical inhérent à une culture (cadences, alternances de tensions et de détentes, citations), et est en général construite, à l'exception peut-être du sérialisme, sur une conception du temps réversible, basée sur le souvenir et l'anticipation. Selon Adorno, *« La première mesure d'un mouvement de symphonie de type classique est perçue seulement quand on a entendu aussi la dernière, qui justifie la première mesure. »*[1] C'est de l'idée même de répétition, par référence interne ou externe à l'oeuvre, que peut émerger le concept plus abstrait de *forme*, c'est-à-dire, d'après Jacques Siron, « [l']*ensemble des processus qui sous-tendent le développement musical.* [...] *Les formes incluent le découpage en plusieurs parties, le montage de différents éléments (répétition, alternances, contraste), l'organisation des phrases et des tonalités, etc.* »[40]

La musique possèdant en outre son propre système de notation, son propre langage, c'est-à-dire un ensemble de signes différentiels pouvant être lus, écrits et interprétés, elle offre la possibilité de rendre compte, sous forme d'une abstraction symbolique, des formes qu'elle produit par delà les répétitions qu'elle génère. La partition n'étant pas une représentation pictographique de la musique, mais une *écriture*, une *séquence* de signes, elle peut donc être encodée sous forme logique et traitée, au même titre qu'une chaîne de caractères, par une machine symbolique telle que l'ordinateur.

Répétitions, formes, similarités, notation séquentielle : Autant par la représentation à laquelle elle se prête que par les structures qu'elle véhicule et les problématiques qu'elle soulève, la musique offre un vaste champ d'expérimentation pour l'Intelligence Artificielle. Resterait à délimiter ou s'arrête la première et ou commence la seconde. En vérité, la grande diversité des travaux conjuguant ces deux savoirs ne permet pas de définir clairement une telle séparation. Il n'est même pas certain que nous en ayons réellement besoin. Les frontières

de l'Intelligence Artificielle sont floues, et celle de la musique ne le sont pas moins. Nous adoptons donc le point de vue de Gérard Assayag et de François Pachet[2] selon lequel « *le simple fait de modéliser des tâches relevant des domaines de compétences des musiciens ou des musicologues fait entrer ce type d'activité dans le domaine de l'Intelligence Artificielle.* »

L'analyse motivique dans un contexte musical se donne donc pour but la localisation et/ou l'extraction de sous-séquences d'abord caractérisées par une certaine fréquence d'occurences. Leur regroupement au sein de classes d'un niveau d'abstraction supérieur sous le concept de *patterns* offre l'avantage de contourner les structures musicales habituellement identifiées par l'analyse (thèmes, variations, phrases, motifs, etc...) et de se concentrer sur le seul phénomène de répétition, rejoignant ainsi le concept d'*analyse paradigmatique* de Nattiez[26]. Cette approche fondée sur une notion intuitive de la similarité et des répétitions ne requiert dans un premier temps aucune forme d'explicitation des entités qu'elle met en évidence. Ces dernières, les *patterns*, peuvent être simplement définies comme des *répétitions perceptibles*[23] au sein d'une pièce musicale[1].

1.2 Improvisation, musicologie, jazz

En général les musiques improvisées, que ce soit dans le jazz ou dans un autre contexte musical, occupent une place relativement mince dans la musicologie. Sans doute parce que l'analyse se propose avant tout de mettre en lumière, par l'étude de la partition, les intentions (ou *intensions*[2]) du compositeur. En se dégageant du temps musical propre à l'oeuvre, elle permettrait de rejoindre le compositeur dans l'*hors-temps*, pour reprendre la terminologie de Xenakis, de l'acte compositionnel. C'est oublier que l'improvisation n'est pas une situation musicale exclusivement contemporaine, et qu'elle a longtemps joué un rôle majeur dans la culture occidentale. Qu'elle était par exemple une pratique courante dans la musique du Moyen-Age, et que l'écriture baroque lui réservait une place majeure avec le système de chiffrage des basses continues. Ce n'est qu'avec le développement progressif de l'écriture et de la théorie qu'elle est devenue, entre le XIX^{e} et le XX^{e} siècle, une activité totalement indépendante de la composition dans la culture occidentale. Mais si l'improvisation est une pratique *en-temps* qui toutefois ne relève pas de l'interprétation, pourquoi dès lors refuser comme espace d'analyse un *hors-temps* qui ne serait pas celui de la composition ? Notre point de vue est que l'improvisation peut occuper une place tout à fait légitime dans la musicologie, sitôt qu'on est en mesure d'en fixer une trace symbolique, donc une écriture, au sens de la notation musicale, traditionnelle ou non, pourvu qu'on fournisse un ensemble de signes reflétant l'ordenancement séquentiel des événements musicaux, ou plutôt, leur *rapport syntagmatique*[19]. Il reste toute-

[1] Il va de soi que nous nous limitons aux répétitions identifiables à partir de la seule partition, ou plutôt de sa représentation logique. Nous excluons ici toute similarité relevant par exemple du timbre, de l'orchestration ou de l'interprétation.

[2] cf. Nicolas, F., *Théorie de l'Ecoute Musicale*, Séminaire de l'*Ecole Normale Supérieure*, inédit, Paris, 2003/2004. Texte disponible en ligne : `http://www.entretemps.asso.fr/Ulm`

fois à définir quels pourraient être les enjeux d'une telle analyse.

Le phénomène de l'improvisation est délicat à cercner. Il s'agit d'un domaine musical très vaste qui ne se limite pas, comme on a trop souvent tendance à le croire, au jazz voire aux musiques improvisées actuelles. Selon Jacques Siron[41] l'improvisation est probablement la manière la plus répandue de faire de la musique dans le monde, et constitue une pratique dominante dans les cultures de tradition orale, particulièrement en Inde, en Afrique et au Moyen-Orient, ainsi que dans certaines traditions populaires comme le Flammenco, et dans les musiques afro-américaines. Outre la pluralité des cultures et des époques dans lesquelles on la rencontre, l'improvisation se distingue par le large éventail de ses modalités : Improvisation individuelle ou collective, libre ou contrainte, dans un stype précis ou selon un scénario déterminé, en relation avec d'autres supports, extra-musicaux (peinture, cinéma, vidéo, textes), ou encore cadrée dans un contexte musical particulier (improvisation mélodique, rythmique, motivique, timbrale, etc...). Dans notre contexte d'étude, le jazz sous sa forme classique[3], l'exécution se déroule toujours, à quelques rares exceptions près[4], selon le même schéma : Un premier *exposé* du *thème*, puis un ou plusieurs *solos* (ou *choruses*) se succédant, chaque musicien improvisant tour à tour sur une trame harmonique préétablie (la *grille* d'accords, en général reproduisant l'harmonie du thème), et enfin une *réexposition finale* du thème. Il est à noter que le soliste n'est pas le seul à improviser, l'improvisation concerne aussi l'accompagnement, ou *rythmique*, dont la qualité est souvent jugée sur son degré d'interaction avec le soliste. Dans notre étude nous ne nous attachons toutefois qu'aux improvisations de ce dernier. Nous prenons certes en compte l'information harmonique, mais dans un cadre contextuel uniquement, et ne tenons pas compte de sa réalisation effective.

Le jazz classique est avant tout une musique tonale. On y retrouve de façon très fréquente (cf. Baudoin[4]) des progressions harmoniques archétypiques de la musique classique (cadences parfaites, imparfaites, plagales, rompues, marches harmoniques, balancements modaux, etc...). A tel point qu'un des axes d'étude indispensable pour le travail de l'improvisation consiste à se constituer un dictionnaire de motifs susceptibles d'être joués sur différentes séquences d'accords. Tous les musiciens de jazz passent par là. Ces motifs pouvant évidemment subir des variations de plus ou moins grande amplitude (en fonction du tempo, de la tonalité, ou tout simplement de l'humeur), ils entrent de plein pied dans notre définition des patterns.

Lorsqu'on étudie le jazz il devient rapidement évident qu'un musicien confirmé se distingue aussi bien par sa technique et son phrasé que par ses choix motiviques lors de ses improvisations. Dans certains cas ces choix constituent véritablement la signature (au sens de Cope, cf. **Repères**, infra) d'un improvisateur. C'est dans cette perspective que nous posons la question, ouverte, de savoir s'il est possible par une analyse orientée patterns de caractériser le style d'un

[3] Des débuts du *Swing*, dans les années trente, jusqu'à la fin du *Hard Bop*, dans les années soixante, où apparaît le *Free Jazz*.

[4] Dans le cas des *Big Bands* notamment, la taille de l'orchestre (une vingtaine de musiciens en général) impose une distribution différente des parties écrites et improvisées.

musicien de jazz. Nous pensons que c'est sous cet angle que les rapports entre musicologie et musiques improvisées peuvent être envisagés, et que la découverte automatique de patterns peut s'avérer d'un secours précieux dans cette tâche.

Dans ce rapport nous explorons successivement deux axes indépendants de recherche, celui des outils pour représenter les connaissances nécessaires à l'analyse et celui des méthodes de localisation de patterns. La notion d'*induction*[5] de patterns n'y est pas abordée. Nous nous contentons de repérer les occurences possibles d'un pattern donné au départ.

Notre base de données est constituée d'une centaine de solos de Charlie Parker (1920-1954) retranscrits et encodés au format MIDI[6]. Ce corpus relativement volumineux nous a été grâcieusement fourni par Pierre-Yves Rolland dont les travaux - dans la continuité desquels s'inscrit ce présent rapport - au cours de sa thèse de doctorat[38] portaient sur Parker. Les raisons du choix de Charlie Parker comme cas d'étude sont multiples.

- Il faut tout d'abord reconnaître que Parker (plus couramment appelé *Bird*) constitue une figure essentielle dans l'histoire du jazz. Fondateur, avec entre autres Dizzy Gillespie et Thelonious Sphere Monk, du mouvement *bebop* (ou encore *bop*), il s'est rapidement distingué comme un musicien d'une grande virtuosité et d'une impressionnante technique, et reste aujourd'hui une étape incontournable dans le parcours des jeunes musiciens de jazz[7].
- Le jazz-bop est une musique difficile à jouer, qui demande une grande maîtrise technique, avant tout en raison des tempi rapides voir extrêmement rapides qui la caractérisent. En général la frontière entre le fast-swing et le bop se situe autour de 200 pulsations à la noire, et les morceaux les plus rapides comme *Bird Gets The Worm* atteignent 340 pulsations à la noire ! A une telle vitesse, aucun improvisateur, aussi doué soit-il, ne peut se permettre de chercher ses chemins « sur le tas », et le recours à un jeu fondé sur des patterns est quasi systématique. La qualité de l'improvisation réside alors dans la diversité des patterns employés et dans le choix de leur ordonnancement. Dans un tel contexte, où Bird excélait, l'analyse motivique a tout son sens.
- Enfin, Charlie Parker est un des rares musiciens de jazz à avoir fait l'objet de travaux musicologiques. Dans sa thèse de doctorat[27], le musicologue américain Tom Owens a recensé un lexique d'environ 200 motifs extraits de 250 transcriptions du jeu improvisé de Charlie Parker. Ces motifs sont organisés dans une structure hiérarchique permettant de les regrouper en classes plus abstraites. Chaque motif comporte en outre le nombre et l'emplacement de ses occurences. Le travail d'Owens est d'une grande utilité

[5] On entend par *induction* l'extraction de patterns *sans connaissances préalables* sur la forme des entités recherchées. En général l'induction résulte d'une phase de regroupement de segments similaires en *clusters*, au sein desquels on identifie ensuite le meilleur représentant, ou *prototype* (cf. par exemple Campouropoulos et al.[8] et Rolland[38]).

[6] Musical Instrument Digital Interface. Pour plus de détails, cf. **Chiffrages**, infra.

[7] La soixantaine de solos de Bird retranscrits dans le célèbre Omnibook[15] constituent en effet, pour les musiciens attirés par le bop, une source très riche pour le travail non seulement de l'improvisation, mais aussi de la technique et du phrasé.

pour notre recherche, il permet de valider les résultats de nos analyses et éventuellement de calibrer des modèles paramétriques ou statistiques à l'aide de techniques d'apprentissage supervisé.

1.3 Plan général du rapport

Repères Nous dressons dans un premier temps une vue d'ensemble des approches passées et actuelles. Loin de prétendre à une énumration exhaustive des technologies et méthodes de découverte de similarité, nous entendons simplement délimiter un cadre de recherche entre musique et informatique vis-à-vis duquel nous tentons de situer notre propre problématique. Nous y justifions par ailleurs nos choix et nos partis-pris quant à l'analyse motivique dans un contexte de jazz bebop.

Chiffrages Nous présentons dans ce chapitre un système de représentations de connaissances musicales nécessaires à la recherche de patterns dans le cadre que nous nous sommes fixé. Le lecteur y trouvera une représentation objet des structures musicales essentielles à l'analyse orientée jazz (notes, accords, séquences d'accords), une brève description de la norme MIDI et de notre corpus de travail (une centaine de solos de Charlie Parker retranscrits en fichiers MIDI). Nous y adjoinions un ensemble de méthodes et d'algorithmes naïfs visant à enrichir le corpus à l'aide de descripteurs de plus haut niveau que le simple stade du « note à note ». Ces méthodes correspondent aux tâches les plus rudimentaires généralement accomplies par un analyste humain lors de l'étude d'un solo de jazz. Nous les considérons comme un pré-traitement indispensable à une recherche de patterns « intelligente », c'est-à-dire qui ne se limite pas aux seules informations contenues dans les notes, mais qui tienne également compte de structures plus globales.

Miroirs Nous tentons dans cette section de définir et de mesurer la similarité entre les séquences musicales. Après un bref exposé du formalisme généralement employé dans des problématiques d'analyse syntagmatique, nous présentons deux approches courantes de la similarité, la première consistant à systématiquement comparer des séquences de même longueurs (modèles à structure de correspondance fixe), la seconde autorisant davantage de souplesse dans le choix des entités comparées (modèle à structure de correspondance variable). Nous discutons l'intérêt potentiel de chacune de ces méthodes quant à notre cadre de recherche et à nos objectifs. Le lecteur trouvera en outre une description détaillée de plusieurs fonctions de similartité implémentées au sein de notre système.

Machineries Nous présentons enfin les deux algorithmes de localisation de patterns mélodiques implémentés dans notre système. Le premier algorithme (δ-*approximate pattern matching*) se restreint à la découverte d'occurences de même longueur que la séquence source. Le second, basé un modèle d'édition,

permet de détecter des occurences de taille quelconque. Ces algorithmes correspondent aux deux paradigmes de mesure de la similarité évoqués au chapitre 3, fondés sur différents types de correspondance entre les segments comparés. Nous présentons quelques résultats de localisation de patterns et comparons nos deux algorithmes en fonction de leur performances et leur intérêt en termes d'analyse musicologique. Dans le cas du modèle d'édition, nous posons également la question de son paramétrage et de sa robustesse.

Trajectoires En conclusion nous tentons de cerner dans quelle mesure le travail réalisé au cours de ce stage peut s'inscrire dans un processus de rechcerche aux perspectives plus larges. Nous envisageons quelques directions futures de l'analyse motivique, nous posons en outre la question de sa portée musicologique et pédagogique dans le contexte du jazz.

Chapitre 2

Repères

« Tout document de culture est aussi un témoignage de barbarie. » - Walter Benjamin

Nous proposons ici une vue d'ensemble des méthodes utilsées pour l'extraction de connaissances à partir d'un contenu musical. Bien que nous nous limitions dans ce rapport à une analyse à partir de données symboliques, nous mentionnons également dans ce chapitre quelques approches orientées signal, utilisant un support audio.

Esquisser un état de l'art en informatique musicale n'est pas chose aisée. La diverité des méthodes, des enjeux et des supports complique sévèrement toute entreprise de classification. En regard des problématiques cognitivistes qui sous-tendent nos recherches (caractérisation de style, modélisation de la créativité), nous adoptons un découpage reflétant ce que nous pensons constiuer les étapes successives d'un possible cheminement :

- Systèmes de reconnaissance : définir la notion de similarité et localiser les occurences d'une forme donnée (aspects *paradigmatiques*[26]).
- Systèmes de découverte : construire des groupements de formes similaires et en extraire des abstractions (aspects *analytiques*).
- Systèmes de production : déterminer un ensemble de règles pour l'effectuation temporelle de ces abstractions (aspects *génératifs*).

2.1 Systèmes de reconnaissance

2.1.1 Identification de concept abstraits

Ces systèmes de reconnaissance s'appuient sur une modélisation de schémas musicaux sous forme d'abstractions et cherchent à en repérer des instances dans la surface musicale.

Représentation explicite Dans certains cas les connaissances musicales sont directement représentées, de manière explicite, au sein du modèle d'analyse. Le

système « sait » ce qu'est une cadence, une marche, un contrepoint de deuxième espèce. L'environnement MusES[1] de François Pachet[29] propose par exemple une représentation objet des connaissances fondamentales en musique tonale (enharmonie, gammes, intervalles, accords, règles d'harmonie). MusES se présente comme une bibilothèque de classes à partir de laquelle il est possible de construire des outils plus élaborés. MusES inclut par exemple un système d'analyse de grilles d'accords de jazz, basé sur des règles de grammaire et un moteur d'inférence d'ordre 1 par chaînage avant. Les règles peuvent également servir à des fins génératives (cf. infra).

Représentation implicite Une autre forme de modélisation, à l'aide d'un modèle connexioniste cette fois, de l'harmonie traditionnelle a été proposée par Bharucha avec le système MUSACT[2][5]. La représentation des connaissances y est implicite dans le sens où elle n'est pas encodée directement dans le système, mais déduite d'une phase d'apprentissage. Pour décrire l'organisation des hauteurs dans le système tonal, MUSACT utilise un réseau de neurones multicouches, où chaque couche représente un niveau hiérarchique de hauteurs (notes, accords, tonalités). L'entraînement du modèle permet la spécialisation des deuxième et troisième couches dans l'identification respective des accords et des tonalités locales.

2.1.2 Identification par « Matching »

Ce type de système cherche à identifer dans un corpus de données les occurences (exactes ou approximatives) d'un pattern donné, sans avoir recours à un quelconque savoir musical. L'accent est ici porté sur la notion de similarité, qui peut être définie selon une grande variété de critères : intuitifs, percetifs, cognitifs, temporels, spectraux, musicologiques. Ce dernier cas ne tombe pas dans la catégorie précédente, la théorie musicale n'étant pas décrite dans le système, seulement mais utilisée pour définir la similarité. Il s'agit uniquement de repérer des segments semblables, et non d'expliciter cette ressemblance à l'aide de classes plus abstraites.

Approches symboliques Dans ce type d'approche l'objet d'étude est une « trace » symbolique de la musique (on parle souvent de *surface musicale*). La plupart des systèmes adoptent en général une représentation numérique (le plus souvent au format MIDI) des hauteurs et des durées. Les méthodes de localisation de patterns peuvent grossièrement être scindées en deux catégories (cf. Cambouropoulos et al.[7]) :

- Recherche d'occurences approximatives, entreprise sur une surface brute et non-structurée. Les occurences sont caractérisées par une certaine proprotion d'éléments communs au pattern recherché. Des algorithmes adaptés à ce type de recherche ont notamment été développés par Cope[11] d'une

[1] Musical Expert System.
[2] MUSic and ACTivation.

part, Stammen et Pennycook[43] d'autre part, dont l'algorithme DTW[3] permet la comparaison de séquences de taille différentes.
– Recherche d'occurence exactes, entreprise sur des projections de la surface sur des composantes structurelles proéminantes. Les identités trouvées peuvent être strictement exactes, relatives (équivalence à une transposition près par exemple) ou partielles (par exemple équivalence sur le plan rythmique uniquement).

Ces deux catégories, si elles permettent une classification des méthodes, ne s'excluent pas mutuellement. Il existe nombreuses approches hybrides. Mongeau et Sankoff[25] ont proposé un algorithme de comparaison des séquences musicales fondé sur un modèle d'édition multi-critères, autorisant à la fois la localisation d'occurences approximatives et la comparaison sur différents niveaux d'abstraction de la surface musicale. Rolland[38] propose une version similaire de cet algorithme, basée sur un modèle d'édition multivalué (cf. **Miroirs** et **Machineries**, infra.).

Approches signal La détection de similarités à partir de données audio a son intérêt dans des problématique de type recherche par le contenu. L'approche la plus courante consiste à générer, à partir du signal audio, des descripteurs de haut-niveau (contour mélodique, informations spectrales, énergie du signal), dont les valeurs peuvent être comparées par un algorithme de matching. Ces descripteurs ne prétendent pas remplacer la notation musicale traditionnelle, mais seulement fournir, sous forme symbolique, une carctérisation du signal. Le lecteur se rapportera notamment aux travaux de Geoffroy Peeters[34] pour le projet CUIDADO[4] et pour l'intégration de descripteurs de haut-niveau dans MPEG-7[5].

2.2 Systèmes de découverte

Les systèmes de découverte sont en général dits *agnostiques*. S'ils possèdent parfois un savoir musical de base (par exemple une représentation explicite de l'harmonie), ils n'ont en revanche aucune connaissance préétablie sur la nature des entitées qu'ils sont sensés extraire. Dans certains cas toutefois, les aspects temporels (c'est-à-dire les résultats générés dans le passé de l'analyse) ainsi que les attentes de l'auditeur peuvent être pris en comptes. Le système Kanthus[19] développé à l'IRCAM par Olivier Lartillot fonctionne sur ce principe, s'auto-alimentant de ses propres analyses[6].

[3] Dynamic Time-Warp Algorithm.

[4] Interfaces de recherche par le contenu et descripteurs pour l'Audio et la Musique accessibles en ligne. `http://www.ircam.fr/produits/technologies/multimedia/cuidado.html`

[5] Standard ISO/IEC développé par MPEG (Moving Picture Experts Group) pour la description de données Multimedia. Une vue d'ensemble de MPEG-7 est disponible à `http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm`.

[6] L'inconvénient de ce type d'approche est qu'elle ne permet pas l'induction de « longs » patterns, en raison d'algorithmes trop couteux.

Les systèmes de découverte obéissent presque toujours à un schéma commun : Regrouper les segments similaires en clusters afin d'en extraire un prototype reflétant les caractéristiques saillantes de sa classe. Idéalement, la construction des clusters nécessite au préalable la comparaison exhausive de tous les patterns potentiels d'une séquence donnée. Sur des corpus volumineux, une telle opération serait évidemment trop coûteuse. Pour palier à ce problème, deux techniques sont couramment employées : la segmentation et la sélection en amont.

2.2.1 Segmentation

La segmentation est une étape de pré-traitement au cours de laquelle la séquence analysée est découpée en patterns potentiels avant la phase de catégorisation (ou clustering), permettant ainsi de réduire considérablement le nombre de comparaisons à effectuer.

Segmentation non-agnostique Cette méthode consiste à diviser la séquence selon des critères relatifs à une théorie musicale. En général le découpage s'opère à des points marquants de la strucutre métrique. Dans la théorie de Lerdhal et Jackendoff[20] par exemple, ces points sont déterminés à partir de règles logiques de décomposition hierarchique de pulsations, fondées sur des critères spécifiques à la tradition classique occidentale.

Segmentation agnostique Cette seconde approche préfère une segmentation fondée sur des critères perceptifs (par exemple l'algorithme de *Beat Tracking* implémenté dans le système MusicMap de Benoît Meudic[22]) ou cognitiviste (cf. l'algorithme LBDM[7] proposé par Cambouropoulos[10]). Ces deux algorithmes utilisent un système de poids pour marquer dans la séquence les points possibles de segmentation.

2.2.2 Sélection en amont

Les « adversaires » de la segmentation lui opposent souvent d'éliminer de manière arbitraire certains patterns. L'alternative souvent proposée consiste à ne comparer que les patterns vérifiant certains critères. Dans son système *Improlog*y, Rolland[38] détermine des longueurs de séquence minimale et maximale en-deçà ou au-delà desquelles les patterns sont ignorés. Sont également laissés de côté les patterns présentant un trop fort degré de recouvrement.

2.2.3 Clustering

L'étape essentielle de l'induction de patterns réside dans le regroupent des entités similaires en clusters. Les clusters peuvent être vus comme un ensemble d'occurences (ou d'effectuations) d'une même entité. Nous citons ici trois algorithmes de catégorisation, prenant chacun en entrée l'ensemble des valeurs

[7] Local Boundary Detection Model.

issues de la phase de comparaison, sous forme d'une matrice ou d'un graphe de similarités.

MusicMap (Meudic[24]) L'induction de patterns est faite par concaténation et filtrage des éléments diagonaux de la matrice de similarité. Les zones de forte similarité sont regroupées en clusters, puis une approche bottom-up est utilisée pour extraire les patterns significatifs. Des patterns de taille supérieure aux séquences produites par la phase de segmentation peuvent ainsi être découverts.

Unscramble (Cambouropoulos et al.[8]) Il s'agit d'un algorithme itératitif capable de raffiner à chaque cycle ses niveaux de seuillage pour optimiser la qualité descriptive des clusters. Le seuillage porte sur la mesure de la similarité entre deux patterns. Au sein d'un même cluster, la similarité de tout couple de patterns est supérieure au seuil.

FlExPat (Rolland[38]) A partir d'un graphe de similarités, un algorithme de type « centre d'étoile » détermine des clusters sous forme de sous-graphes. Cette approche possède en outre l'intérêt de déterminer, parmi les éléments d'un clusters, le pattern le plus représentant (prototype) de la classe considérée. Ce prototype peut alors être considéré comme une *abstraction* musicale, une classe d'objets dont les autres occurences représentent des *applications*.

2.2.4 Approches Audio

La catégorisation concerne également les donnés non-symboliques. L'approche courante (cf. Peeters[35]) est de considérer le signal comme une succession d'« états » (à différentes échelles), formant la structure musicale même. Une analyse de la similarité de ses états et leur regroupement en clusters permet de mettre en évidence des frontières dans le contenu, et de délimiter des segments correspondant à des structures musicologiques (formes, mouvements, thèmes, variations, etc, cf. Dannenberg et Hu[14]). Cette méthode est particulièrement adaptée aux pièces comportant des variations facilement repérables au niveau audio (changements brusques de dynamique engendrant de grandes variation d'énergie, modifications de timbre).

2.2.5 Approche hybride : l'effet « AHA »

L'effet « AHA », introduit par François Pachet dans le projet CUIDADO, est un concept original résultant d'un questionnement sur l'utilité d'une mesure telle que la similarité. Selon Pachet et Aucouturier[33], l'extraction de connaissances présente un intérêt limité si elle ne produit aucune information nouvelle (justifiant le recours à la machine) quant à son objet d'analyse. Par exemple, une similarité élevée entre deux lieder de Schubert n'apporte rien que l'on ne sache déjà.

L'idée de Pachet est de confronter différentes mesures de la similarité entre deux pièces de musique, une première calculée sur des descripteurs de timbre,

et une seconde sur des métadonnées (genre musical, nom de l'artiste, époque, etc). La qualité de l'information véhiculée par la similarité est basée sur le degré de surprise résultant de cette confrontation. Dans l'idée d'orienter l'auditeur vers des choix musicaux qu'il n'aurait pas fait spontanément, AHA privilégie les pièces présentant une forte similarité audio et une faible similarité textuelle (comparaison de métadonnées). Un tel critère permet par exemple de relier le *Kreisleriana Op. 16-5* de Schumann avec *I love you Porgy* interprêté par Bill Evans.

2.3 Systemes de génération

Ces systèmes visent à produire automatiquement un contenu musical symbolique inédit, pouvant être rendu sous forme sonore via une interface MIDI. Leurs applications sont variées : Composition automatique, simulation d'improvisation, accompagnement automatique, jeu interactif avec des musiciens humains. Ils entrent dans des problématiques d'apprentissage de style auxquelles l'Intelligence Artificielle offre deux types d'approches, souvent considérés comme antagonistes par leurs détracteurs respectifs. Pour chacune d'entre elles, nous citons quelques exemple de systèmes ayant fait leurs preuves.

2.3.1 L'école formelle

EMI (*Experiments in Musical Intelligence*) est un système de composition automatique imaginé par David Cope[12] en 1981, utilisant des règles de grammaire pour générer de la musique dans le style d'un compositeur particulier. L'idée de Cope est que toute pièce musicale contient de façon intrinsèque un jeu d'instructions pour créer d'autres pièces, différentes mais fortement corrélées à la pièce originale. Ce processus passe par une « déconstruction » de l'oeuvre en vue d'en extraire une caractérisation du style d'un compositeur via la recherche de similarités globales (communes à plusieurs pièces) que Cope désigne par le terme de *signatures*. Cope est ainsi l'auteur de plusieurs oeuvres dans le style de Bach, Mozart, Malher, Beethoven, Chopin, Joplin, et... Cope !

ChordSequenceMaker est un outil développé par François Pachet[30, 31] pour créer, analyser et transformer des séquences d'accords dans le contexte du jazz en utilsant des règles de substitution. Le système est basé sur une représentation explicite (de type MusES[29]) des accords et utilise un formalisme inspiré des grammaires de Steedman[44].

ImPact (Ramalho[36]) est un système de simulation du jeu d'un bassiste accompagnant un musicien de jazz en solo à partir d'une grille d'accords prédéterminée. La mémoire musicale y est représentée sous forme de fragments, combinés en temps-réel selon un raisonnement à partir de cas et des mécanismes d'inférence à partir de règles de production. Les actions musicales y sont décrites de manière explicite grâce à la notion de PACTs (Potential ACTions[28]). En prenant en compte le non-déterminisme et

l'absence de buts prédéfinis ce modèle rentre dans le cadre de travaux sur la modélisation de la créativité[37].

2.3.2 L'école probabiliste

Dictionary-Based Prediction Cette approche développée par Assayag et al.[3] propose un algorithme d'apprentissage basé sur un arbre de suffixes prédictif, déterminant dans le corpus d'entrainement des motifs de fréquence élevée. Cette méthode s'applique parfaitement à la synthèse automatique de nouvelles séquences musicales au sein desquels les motifs découverts interviennent en fonction de leur probabilité d'occurence.

The Continuator est un système interactif d'apprentissage de style et de génération d'improvisations en temps-réel[32]. Le système d'apprentissage est basé sur un modèle de chaînes de Markov cachées (HMM) capable de gérer des problématiques musicales telles que le rythme, la pulsation, l'harmonie et même l'imprécision rythmique. Le Continator est capable d'apprendre et de générer des improvisations sans contraintes particulières de style. Il constitue une innovation majeure par son haut degré d'interaction et sa capacité à prendre en compte des styles musicaux variés.

2.4 Positions

En regard de l'état de l'art qui précède, nous exposons dans cette section un certain nombre de choix relatifs à l'implémentation de notre système d'analyse.

Comme la plupart des systèmes présentés ci-dessus, nous utilisons comme support une représentation symbolique des événements musicaux, construite à partir de fichiers MIDI monophoniques. Le contexte musical dans lequel se situe notre recherche étant de la forme « mélodie/accompagnement », nous ne pensons pas subir une perte conséquente d'information en ne tenant pas compte de la polyphonie, d'autant plus que nous travaillons sur les improvisations d'un musicien pratiquant un instrument monodique (le saxophone alto). Nous retenons toutefois le contexte harmonique dans lequel se déroulent ces improvisations, mais sous la forme d'une abstraction, la grille d'accords. Cette démarche nous semble cohérente avec l'étude de l'improvisation en jazz, telle qu'elle est généralement pratiquée.

Prétendant à terme parvenir à une caractérisation du style par le biais d'une analyse motivique que nous souhaiterions basée sur une procédure agnostique de découverte de patterns, nous limitons la représentation des connaissances musicales au strict minimum : notes, accords, et séquences d'accords. La modélisation explicite de ces séquences nous paraît justifiée dans la mesure où les progressions harmoniques les plus courantes (cf. Baudoin[4]) font partie intégrante du savoir musical partagé par les musiciens de jazz (cf. Pachet[28]).

Nous insistons enfin sur le fait que nous proposons ici un outil d'analyse orientée dans un style musical très particulier, le jazz tonal bebop. Nous ne

prétendons pas fournir un modèle général. Nous pensons que ce genre de limitations, dans un premier temps, est nécessaire à l'obtention de résultats d'une portée musicologique significative.

Chapitre 3

Chiffrages

« Pour incroyable que cela paraisse, je crois qu'il y a, (ou qu'il y eut) un autre Aleph, je pense que l'Aleph de la rue Garay était un faux Aleph. » - Jorge Luis Borges

Nous présentons dans ce chapitre une architecture pour l'analyse motivique de solos de jazz. A l'image de MusES[29], notre système, implémenté en Python[1] est consitué de méthodes d'analyse greffées sur un noyau de modélisation des connaissances musicales. Nous donnons en outre une description des données d'entrée de notre système, un corpus constitué d'une centaine de fichiers MIDI contenant une partie des solos de Charlie Parker retranscrits par Owens. Nous présentons enfin quelques algorithmes de pré-traitement destinés à enrichir les données d'entrée en vue d'une analyse multi-critères.

3.1 Représentation des connaissances musicales

Nous adoptons une approche orientée objet pour représenter des structures hiérarchiques de hauteur : notes, accords, séquences d'accords (progressions harmoniques). Les aspects temporels sont pris en compte au sein de chaque note, les aspects rythmiques, en particulier métriques, sont pris en compte au sein d'une classe plus globale (cf. objet MonoMidiFile, infra.).

Les classes relatives aux structures musicales sus-cités sont implémentées sans lien de hiérarchie objet. Il n'existe pas de relation d'héritage entre ces objets. Nous pensons que les notes, les accords et les séquences d'accords, bien qu'a priori liées par des relations d'inclusion, constitue des structures totalement indépendantes les unes des autres. Ceci est particulièrement vrai en jazz, où les accords, par exemple, ne sont jamais notés explicitement sous forme d'un empilement de notes, mais représentés par un chiffrage dont la fonction se limite à la « suggestion » d'une « intention harmonique ». Rien n'oblige un pianiste à jouer un Sol lorsqu'il lit un accord de Do majeur. Libre a lui également d'inclure

[1] Langage de programmation orienté objet. Python est gratuit, interactif, interprétable et multi-plateformes. `http://www.python.org`.

des notes de la superstructure (i.e. $7^{\text{èmes}}$, $9^{\text{èmes}}$, $11^{\text{èmes}}$, et $13^{\text{èmes}}$), plus où moins altérérées selon le contexte harmonique et le style. Cette remarque s'applique également au cadences. Lors de réalisation d'un $II-V-I$ (la séquence harmonique la plus courante en jazz, cf. Baudoin[4]) en Do majeur, personne n'est forcé de jouer exactement les accrods de Ré mineur, Sol majeur et Do majeur. Certains accords peuvent être omis, d'autres insérés, d'autres encore subsitués (cf. Siron[41] et Steedman[44]).

Nous préférons donc considérer les notes, les accords et les cadences comme autant de niveaux possibles d'analyse, autant de « plateaux » accessibles à la connaissance musicale et entre lequels, selon nous, la pensée tisse des ramifications irréductibles au liens unilatéraux qu'implique la notion d'héritage.

3.1.1 Notes

La classe `note` comporte quatre attributs de base fondamentaux, correspondant à la description employée par la norme MIDI :

`note.pitch` Hauteur exprimée en nombre de demi-tons. Comme dans la norme MIDI, la valeur 60 correspond au Do central.

`note.duration` Durée exprimée en ticks d'horloge MIDI. La valeur de référence adoptée dans notre système est de 120 divisions par temps. On peut ainsi représenter par des nombres entiers une grande variétés de valeurs rythmiques : noires, croches, doubles- et triples-croches, triolets, quintolets, sextolets, etc...

`note.position` Position exprimée en ticks d'horloge MIDI. La valeur 0 correspond au premier temps de la première mesure.

`note.velocity` Intensité, ou « vélocité » de la note : Valeur comprise entre 0 et 128 permettant une modélisation grossière de la dynamique, très utile pour une pour prendre en compte des informations relatives au phrasé.

D'après Cambouropoulos[6], la représentation des hauteurs et des intervalles la plus couramment utilisée, à savoir basée sur un nombre de demi-tons entre les notes, est insuffisante pour l'analyse de la musique tonale, car elle ne prend pas en compte l'enharmonie. Elle ne suffit pas à différentier un Ré bémol d'un Do dièse. Pour pallier à cette inconvénient, Cambouropoulos propose le GPIR[2], un modèle de représentation des intervalles en fonction d'un jeu de tonalités pertinentes au sein d'un segment musical donné.

Dans notre modèle, le degré diatonique de chaque note dans la tonalité locale est directement encodé dans la classe `note`. D'autres attributs sont également implémentés. Nous les développons toutefois en infra, car ils ne sont renseignés qu'au moment de l'enrichissement des données, lorsque d'autres informations comme tempo, les tonalités locales, ou la signature métrique seront prises en compte.

[2] General Pitch Interval Representation.

3.1.2 Accords

Conformément à ce qui a été dit en supra, nous ne définissons pas les accords comme une listes de notes, mais par la fonction désignée par le chiffrage. Nous nous limitons à quatre types d'accords fondamentaux : Les accords majeurs (chiffrés[3] X, XM, ou $XMaj$), les accords mineurs (chiffrés Xm, $Xmin$, ou $X-$), les accords de dominante (chiffrés $X7$), et les accords demi-diminués (chiffrés $Xm7b5$ ou $X\emptyset$).

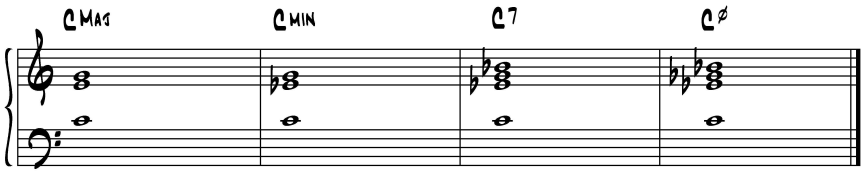


FIG. 3.1 – Types d'accords supportés par le système

Avec ces quatre types d'accords la quasi intégralité des grilles de jazz bop peuvent être représentées (cf. Baudoin[4]). Nous n'avons pas voulu inclure dans notre représentation les accords diminués, car ils sont le plus souvent utilsés comme accords de passage, et peuvent aisément être remplacés par des accords mineurs ou des accords de dominante.

La classe chord comprend six attributs :

`chord.root` Fondamentale de l'accord, exprimée en notation anglo-saxonne.

`chord.type` Type de l'accord : majeur (`M`), mineur (`m`), de dominante (`7`), ou demi-diminué (`o`).

`chord.duration` Durée de l'accord exprimée en nombre de noires.

`chord.position` Position de l'accord exprimée en nombre de noires. Dans le cas de structures cycliques comme un standard de jazz où la grille harmonique est répétée plusieurs fois, la valeur 0 correspond au premier temps de la première mesure du cycle, même en présence d'une levée.

`chord.following` Un pointeur sur l'accord suivant s'il existe.

`chord.preceding` Un pointeur sur l'accord précédent s'il existe. Ces deux derniers champs sont très utiles pour l'identification de cadences lors d'un processus d'analyse harmonique.

```
chord(Am).abstract(C) -> abstractchord(VIm)
  abstractchord(VIm).apply(G) -> chord(Em)
```

FIG. 3.2 – Accords : Absraction/Application

[3] Où X est la tonique de l'accord dans la notation anglo-saxonne.

Accords abstraits Le concept d' »accord abstrait » permet de définir les accords non plus par rapport à leur fondamentale mais par rapport à leur degré diatonique[4] dans la tonalité. Les accords abstraits interviennent notamment dans la définition des cadences (II-V-I, VIb-V-I, I-IV-I, etc...). Nous avons implémenté une classe **abstractchord** différant de la classe **chord** par son premier attribut uniquement (`chord.root` est remplacé par `abstractchord.deg`). Nous définissons en outre des méthodes d'abstraction/application permettant de passer d'un accord à un accord abstrait et vice versa. Par exemple, l'abstraction d'un accord de La mineur par rapport à la tonalité de Do majeur engendre un accord abstrait du VI^{e} dégré mineur. L'application de ce dernier dans la tonalité du Sol majeur engendre un accord de Mi mineur.

3.1.3 Séquences harmoniques

Nous définissons et implémentons une séquence harmonique (classe chordseq) comme une liste d'accords abstraits. Cet objet possède aussi un nom et une tonalité.

chordseq (instance)	
`chordseq.name = [IIo,V7,Im]`	
`chordseq.key = 'G minor'`	

`chordseq.chordlist =`	`abstractchord(IIo),`
	`abstractchord(V7),`
	`abstractchord(Im)`

FIG. 3.3 – Exemple de séquence d'accords : $Am7b5 - D7 - Gm$

Séquences d'accords abstraites Comme pour le cas des accords nous définissons des progressions harmoniques « abstraites ». Elles correspondent à un niveau réel de représentation des connaissances dans la pratique du jazz. Il est en effet coutumier d'entendre les musiciens parler des cadences en utilisant les termes généraux de « turnarounds », « $II - V - I$ », « anatoles », « cadences Christophes », etc[5]. Ces notions sont indépendantes d'une tonalité. Elles constituent une abstraction musicale. Dans notre système les cadences abstraites sont implémentées à l'image des séquences normales, à l'exception de la tonalité. Les cadences abstraites sont nécessaires pour représenter les connaissances harmoniques du système, de la même façon qu'un musicien de jazz connaît un certain nombre de progressions harmoniques indépendamment de la tonalité. Il est par ailleurs remarquable que deux cadences identiques à la tonalité près ne diffèrent, grâce à la notion d'accord abstrait, que par l'attribut `chordseq.key`. L'implémentation de la transposition est ainsi immédiate.

[4] Exprimé ici, selon l'usage, en chiffres romains.
[5] Pour plus de détails voir Siron[41].

Les objets présentés dans cette section ne constituent pas l'intégralité des connaissances musicales prises en compte par le système. Une description exhaustive est fournie en fin chapitre. Il nous faut d'abord dire quelques mots sur l'ensemble des données d'entrée de notre système et sur les méthodes d'enrichissement de ces données par des decripteurs de plus haut niveau que le simple stade du « note à note ».

3.2 Données d'entrée du système d'analyse

Les données d'entrées du système sont de trois types :

1. Les données « numériques », soit l'ensemble des hauteurs et des durées contenues dans le corpus de fichiers MIDI monophoniques.
2. Les données « symboliques », c'est-à-dire les grilles d'accords sur lesquelles sont construites les improvisations. Ces grilles sont stockées dans des fichiers textes indépendants des fichiers MIDI.
3. Les « métadonnées », soit l'ensemble des indications de tonalité globale, de tempo, de signature métrique, et éventuellement, de forme.

Est-ce fournir *trop* d'informations au système que de ne pas se contenter du premier type de données seulement ? Les systèmes agnostiques ne se limitent-ils pas, en général, à la seule partition ? Nous répondrons que c'est exactement ce que fait notre outil d'analyse ; en jazz en effet, toutes les données sus-citées figurent bel et bien sur la partition. A l'exception du blues, la forme, lorqu'elle est standard (AABA, ABAC, AAB, etc) est souvent écrite sur le score, et la tonalité globale (pour le jazz tonal du moins) peut très souvent être déduite de l'armure[6]. Nous n'avons donc pas le sentiment de faciliter la tâche du système, nous lui fournissons la même quantité d'information dont disposerait un analyste humain.

3.2.1 Round about MIDI

MIDI[7] (Musical Instrument Digital Interface) est une spécification matérielle et logicielle conçue pour l'échange d'informations entre instruments de musique, ordinateurs, séquenceurs et autres machines. MIDI décrit un format de données pour le contrôle d'instruments en temps-réel et permet le stockage en mémoire d'une représentation numérique de la notation musicale. MIDI reste aujourd'hui le format d'échange de données musicales symbolique le plus répandu, malgré les tentatives d'élaboration de standards plus génériques comme NIFF, Score ou SMDL[8]. Seul MusicXML[16] est aujourd'hui potentiellement en mesure de tenir tête au format MIDI.

[6] Le lecteur pourra facilement s'en convaincre en consultant les incontournables *RealBooks* : Real, Fake, Colorado, New Real (Sher Music Co.), etc...

[7] Une description complète de la norme MIDI est disponible sur le site de la MMA (MIDI Manufacturer's Association) : `http://www.midi.org`.

[8] Le lecteur intéressé par les standards de notation musicale pourra se reporter à l'ouvrage d'Eleonor Selfridge-Field : *Beyond MIDI : The Handbook of Musical Codes*[39].

Format Les fichiers MIDI (d'abords conçus pour le stream) sont structurés par blocs (ou *chunks*). On distingue les blocs d'en-tête (*header chunks*) et les blocs de pistes (*track chunks*). Un fichier MIDI est toujours constitué d'un en-tête contenant un minimum d'informations portant sur la globalité du fichier, et de une ou plusieurs pistes contenant les événements MIDI proprement dits. La norme MIDI spécifie trois types de formats :

Format 0 Le fichier contient une seule piste (multicanal).

Format 1 Le fichier contient une ou plusieurs pistes (monocanal) simultanées d'une même séquence.

Format 2 Le fichier contient une ou plusieurs pistes (monocanal) séquentiellement indépendantes.

Dans notre système nous utilisons exclusivement des fichiers de format 1, ne contenant qu'une seule piste.

Evénements Les événements MIDI contenus dans chaque piste se présentent sous la forme d'une liste de couples `(horodatage,action)` spécifiant ce que l'utilisateur (application ou instrument) du fichier doit faire et quand il doit le faire. Les événements sont de deux types :

Evénements "musicaux" Ils concernent la surface musicale uniquement. Ils permettent soit de déclencher une note (NoteON), soit de l'arrêter (NoteOFF). Les paramètres de note sont la hauteur, l'intensité, et le canal.

Méta-événements Ils englobent la détermination du tempo, de la tonalité, de la signature métrique, le choix des *patches* (catégories d'instruments), les marqueurs et les données textuelles.

Horodatage Il est exprimé en ticks d'horloge MIDI et peut être absolu ou relatif. Dans ce dernier cas le tempo et le nombre de ticks par valeur rythmique (le plus souvent la noire) doivent êtres spécifiés (en général dans le bloc d'entête). Les fichiers que ne nous utilisons relèvent de cette seconde catégorie.

Parser L'information contenue dans fichiers MIDI est représentée sous forme hexadécimale et n'est pas accessible directement à l'aide d'un éditeur de texte. Pour récupérer cette information, les fichiers doivent êtres *parsés*. La difficulté de cette tâche réside dans le fait que la taille mémoire occupée par les événements MIDI diffère selon leur type. Nous avons implémenté un parser, transformant les données hexadécimales en données textuelles, à partir d'une bibliothèque de fonctions MIDI développée en Python[9], contenant des méthodes appropriées pour la lecture et l'écriture de champs de taille variable.

[9]Disponible en ligne : `ftp://ftp.cwi.nl/pub/jack/python`.

3.2.2 99 solos de Charlie Parker !

Notre corpus d'étude est constitué de 99 fichiers MIDI obtenus avec l'aimable autorisation de Pierre-Yves Rolland. Ces fichiers contiennent les transcriptions par Owens d'improvisations de Charlie Parker[10] (une liste détaillée du corpus est disponible en annexe A). La plupart des fichiers étaient directement exploitables par notre système. Nous avons toutefois dû procéder à un cerain nombre de modifications.

Format Les fichiers avaient initialement été générés par un éditeur de partitions incluant de courts silences après chaque note. Or d'un point de vue symbolique, les silences occuppent le même espace mémoire que les autres notes. Leur élimination a donc permis, sans perte d'information, de réduire de moitié la taille du corpus.

Erreurs Certains fichiers contenaient des notes d'une durée supérieure à l'intervalle de temps les séparant de la note suivante, générant des erreurs dans leur traitement. Nous avons développé des méthodes pour éliminer ces défauts.

Structures La prise en compte de l'harmonie par notre système implique certaines contraintes de structure quant aux imropovisations. La notion de cycle (une grille donée se répétant à l'identique) interdit par exemple la présence d'interludes (petit nombre de mesures jouées entre deux solos), très fréquents en jazz. Les fichiers contenant des interludes été découpés en plusieurs parties.

Le nettoyage et la constitution d'un corpus homogène a été une étape longue et laborieuse de notre travail. Ces difficultés sont assez révélatrices de l'absence de spécification claire pour la représentation des données musicales à l'aide du format MIDI. Les bases de données accessibles sur Internet regorgent de fichiers d'un décourageante hétérogénéité. Souhaitons qu'à l'avenir une spécification (par exemple de de type DTD pour MusicXML[16]) permette d'éviter ce genre de contretemps.

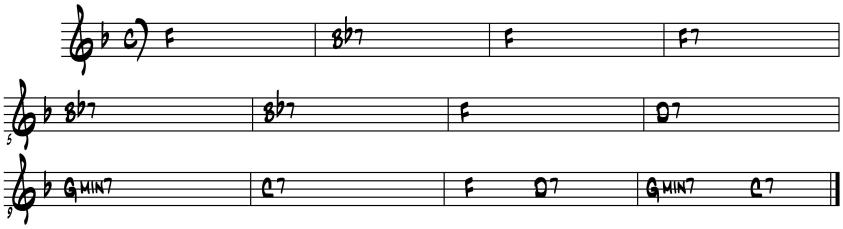
3.2.3 Données supplémentaires

Un fichier MIDI « bien formé » au sens de notre système doit être de format 1 (un seul canal par piste) et strictement monophonique. Il doit en outre comporter des indications de tempo, de métrique et de tonalité. Ces dernières informations peuvent être soit contenues dans l'en-tête du fichier, soit figurer au début de la première piste (la seule, d'ailleurs). Les autres données sont importées sous forme textuelle, à partir d'autres fichiers.

[10] Au total plus de 36300 notes, soit environ 116 minutes de musique.

Grilles d'accords

Chaque fichier MIDI est accompagné de la grille d'accords lui correspondant, à l'image de partition d'un standard de jazz. Les grilles sont stockées sous forme de listes dans un fichier texte. Lors de la constitution du corpus, ces listes sont parcourues par un constructeur qui instancie autant d'objets de la classe chord et construit une représentation de la trame harmonique sous forme d'une liste chaînée, dont le dernier accord pointe sur le premier. Ceci permet de prendre en compte, de façon très simple, la notion de cycle.




```
[['F M',4],['Bb7',4],['F M',4],['F 7',4],['Bb7',8],['F
M',4],['D 7',4],['G m',4],['C 7',4],['F M'],2],['D
7',2],['G m',2],['C 7',2]]
```

FIG. 3.4 – Une grille de Blues en Fa majeur et sa représentation dans un fichier texte

Formes, levée

Pour chaque fichier nous spécifions en outre la forme (Blues, AABA, ABAC, etc) et le nombre de mesures de la levée si elle existe. Nous avons souhaité prendre en compte la forme dans l'intention de décrire, dans des travaux ultérieurs, les intentions de l'improvisateur, correspondant à la notion de PACTs proposée par François Pachet[28, 36] (par exemple : dans une forme AABA, doubler le tempo sur le B et jouer très altéré).

L'indication de levée est nécessaire sitôt que l'on considère le contexte harmonique. Elle permet de situer le début du cycle. Pour déterminer la durée de la levée, nous avons dû, dans certains cas, « accompagner » le fichier MIDI au piano. Ce fut un travail plaisant certes, mais long.

3.3 Enrichissement des données

Afin de procéder à une analyse un peu plus poussée que la seule comparaison des hauteurs et des durées, nous procédons à un pré-traitement des données afin d'intégrer à notre représentation des descripteurs d'un niveau d'abstraction plus élevé que la simple surface musicale.

3.3.1 Segmentation en phrases

Le recours à la segmentation est chose courante dans l'analyse motivique (cf. **Repères**). Dans notre contexte, la détection des phrases dans le discours mélodique permet d'éliminer les patterns à cheval sur sur deux phrases. Analyser un solo au moyen de motifs comportant un long silence en leur milieu semble en effet peu pertinent.

La segmentation permet en général d'améliorer considérablement la performance des algorithmes (cf. Cambouropoulos[8]). Toutefois, dans le cas de la seule localisation de patterns, ce procédé n'a pas beaucoup d'intérêt (cf. **Machineries**). Ce pourquoi dans notre système nous n'opérons pas de sélection en amont au moyen de la segmentation, et nous nous laissons la liberté d'ignorer ce découpage lors de la localisation de patterns. Nous utilisons la segmentation en phrases afin de filtrer (en aval) les résultats de l'algorithme de localisation, et d'éliminer les patterns improbables d'un point de vue à la fois analytique et cognitif.

Heuristiques intuitives

Nous formulons l'hypothèse qu'une fin de phrase mélodique est marquée soit par un silence suffisamment long, soit par une note tenue suffisamment longtemps. Il faut évidemment décider ce qu'on entend par « suffisamment ». Un silence d'un temps n'a bien sûr pas la même valeur perceptive vis-à-vis d'une mélodie en noires ou en double-croches.

Nous procédons donc dans un premier temps à une analyse statistique des durées. Nous retenons comme durée de référence la valeur rythmique la plus fréqente (au sens de la proportion en temps) dans le fichier. Nous appliquons ensuite les règles suivantes :

Algorithm 1 Détection des fins de phrases

```
pour_chaque note dans fichier_MIDI
{
   si note = silence
   et note.duration > a*durée_de_référence
     alors note ← fin_de_phrase
   si note.duration > b*durée_de_référence
     alors note ← fin_de_phrase
}
```

Le choix des valeurs des paramètres a et b est laissé à l'utilisateur. Suite à une série d'expérimentations sur nos données, il est apparu que les valeurs $a = 2$ et $b = 4$ donnaient des résulats très satisfaisants.

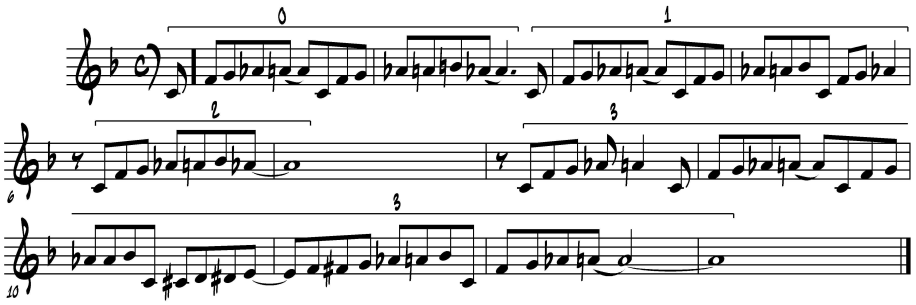


FIG. 3.5 – Segmentation en phrases mélodiques du thème *Straight No Chaser*

Ratios du durée

De même que pour les hauteurs, Cambouropoulos[7] suggère que la représentation usuelle des durées (sous forme de durée absolue) est insuffisante et montre l'importance de travailler avec des ratios de durée. La segmentation en phrases intervient dans le calcul des ratios. Pour les notes en début de phrase, ce ratio sera toujours égal à 1. Ainsi une phrase jouée deux fois plus vite que la présécente aura exactement les mêmes ratios de durée[11].

Approches plus poussées

La méthode employée par notre système pour segmenter les phrases s'avère très performante dans le contexte du jazz bop et particulièrement dans le cas de Charlie Parker, dont le discours mélodique est souvent basé sur de longues phrases entrecoupées de silences significatifs. Il est toutefois peu propable que cette méthode s'étende à d'autres genres musicaux. **EMI**[12], par exemple, utilise un jeu de règles de grammaires pour la segmentation. **MusicMap**[22] a recours à un algorithme de détection de la pulsation (*Beat Tracking*).

Pour l'analyse de la musique interprétée ou non quantifiée comme des solos improvisés, l'implémentation du **LBDM**[10] de Cambouropoulos constituerait une amélioration potentielle de notre système, et rendrait l'extraction de phrases mélodiques indépendante de paramètres arbitraires.

3.3.2 Analyse harmonique

Le prinicpal intérêt de l'analyse harmonique est la mise en évidence des tonalités locales au sein d'une grille de jazz. Il devient alors possible de comparer des motifs par les dégrés diatoniques des notes les constituant. Dans l'*Allegretto en Do mineur D.915* de Schubert par exemple, les mesures 1 et 5 diffèrent du point de vue de la hauteur absolue et des intervalles chromatiques. En revanche elles sont strictement identiques quant aux degrés diatoniques.

[11] Dans le cas d'une note précédée d'un silence son ratio est également fixé à 1.

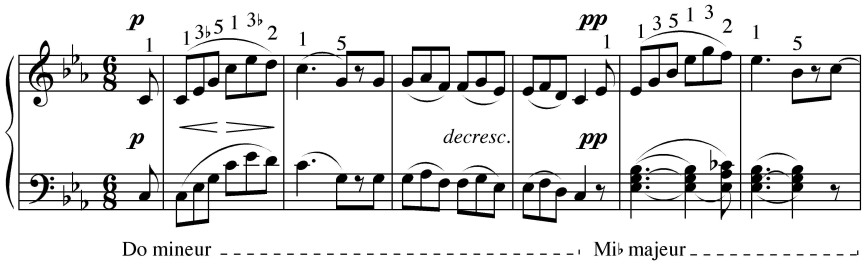


FIG. 3.6 – Analyse en degrés diatoniques des six premières mesures de l'*Allegretto en Do mineur D.915* de Franz Schubert

Progressions reconnues par le système

L'analyse harmonique est basée sur un algorithme de matching qui cherche à repérer des séquences d'accords abstraites dans la grille d'accords. Ces séquences abstraites constituent le « savoir » harmonique du système. De même que pour les grilles d'accords, les séquences reconnues par le système sont représentées dans le code sous forme textuelle (cf. Appendice B), facilitant ainsi l'ajout de nouvelles séquences. Lors de la mise en route du système, un constructeur instantie autant de séquences abstraites et dresse une « carte » des types d'accords. Pour chaque type d'accord (majeur, mineur, septième de dominante, demi-diminué), le système génère une liste des progressions harmoniques dans lesquelles il figure.

Cas du blues Le blues tel qu'il est joué par les musiciens de jazz est fondé sur une ambiguité entre modes majeur et mineur. Dans un blues en Fa majeur par exemple, il est très fréquent de d'« insister » sur les notes Mi bémol et La bémol, qui donnent cette couleur si caractéristique du blues. Sur le plan harmonique, on emploiera très souvent l'accord de Fa septième ($F7$) comme premier degré. Cet accord ne devant pas être considéré comme une dominante (auquel cas la tonalité sous-jacente serait Si bémol), nous introduisons à cette effet des cadences de type « blues », comprenant des accords de type $I7$ (premier degré avec septième mineure). Ces cadences possèdent le préfixe « Blues » au début de leur nom. Elles ont la particularité de ne pouvoir être utilisées par le système que sur des formes de type « Blues ».

Sous-dominante mineure Le passage par la sous-dominante mineure (emprunt du quatrième degré mineur dans un contexte majeur) est très fréquent en jazz tonal. La cadence « Christophe[12] » en consitue l'exemple parfait. Il faut remarquer toutefois que si la progression $EbM - Ebm - Ab7 - BbM$ est une cadence Christophe dans la tonalité de Si bémol majeur, tel ne serait pas le

[12]Ce nom vient d'un standard de jazz, *Christopher Colombus*, dans lequel cette cadence est employée (cf. Siron[41]).

cas dans une autre tonalité. Ce type de progression doit donc faire l'objet d'un test supplémentaire sur la tonalité globale. Ce pourquoi préfixons le nom de cadences du type « Christpohe » ou « SD » (pour Sous-Dominante). Le système ne tentera alors d'appliquer ces cadences que si la tonalité du morceau le permet.

Algorithme de matching

Le principe de l'algorithme est simple. Pour chaque accord de la grille, on cherche à repérer (par un mécanisme d'application) la plus grande progression harmonique correspondant à cet accord et aux accords voisins. L'opération est répétée jusqu'à ce que tous les accords appartiennent au moins à une séquence harmonique.

Algorithm 2 Analyse harmonique des grilles

```
analyse ← {}
tant_que grille est_non_vide
{
  accord ← premier_élément(grille)
  pour_chaque séquence_abstraite contenant accord
  {
    degré ← degré(accord) dans séquence_abstraite
    tonalité ← tonalité_de accord de_degré degré
    dès_que séquence_abstraite.apply(tonalité)
            = segment_de grille
    sortir
  }
  analyse.ajouter(séquence_abstraite.apply(tonalité))
  grille.retirer(accord & suivants_dans segment_de grille)
}
éliminer_séquences_redondantes(analyse)
```

Le lecteur trouvera en annexe les analyses d'un blues (*Straight No Chaser*) et d'un standard de jazz tonal avec passage à la sous-dominante mineure (*Stella By Starlight*).

Extensions

Le lecteur trouvera en appendice une analyse harmonique d'un standard de Parker, *Donna Lee*. Dans cette analyse (le morceau est en La bémol majeur), le système ne sait pas comment considérer les accords $F7$ et $Bb7$. Faute de mieux (le type de cadence $VI7 - II7 - IIm - V7 - I$, soit ici $F7 - Bb7 - Bbm - Eb7 - AbM$, lui est inconnu), il les interprète comme des accords de dominante, suggérant ainsi les tonalités de Si bémol majeur et Mi bémol majeur.

Un raffinement possible de notre système d'analyse serait d'avoir recours à des règles de grammaires lorsqu'il a affaire à des progressions qu'il ne connaît

```
Séquence initiale : V7-I
. Préparation par dominante → II7-V7-I
. Préparation par mineur → II7-IIm-V7-I
. Préparation par dominante → VI7-II7-IIm-V7-I
```

FIG. 3.7 – Exemple d'application de grammaires de Steedman

pas. Les grammaires de Steedman seraient particulièrement adaptées. En effet, pour produire la séquence $VI7 - II7 - IIm - V7 - I$, il suffirait de partir de la cadence $V7 - I$ et d'appliquer 3 règles dites de « préparation ».

3.3.3 Analyse diatonique

Une fois les tonalités locales repérées, il est possible de déduire les dégrés diatoniques de chaque note au sein la(les) tanolité(s) à laquelle(auxquelles) elle appartient. Pour cela il suffit de calculer l'intervalle modulo 12 entre la hauteur de la note considérée et celle de la tonalité locale, puis lire dans la liste suivante l'élément dont l'indice correspond à cet intervalle :

```
['1','2b','2','3b','3','4','5b','5','6b','6','7b','7']
```

Evidemment cette représentation ne tient pas compte de l'enharmonie. En Do majeur par exemple, les notes $F\sharp$ et Gb seront toutes deux considérées comme une quinte diminuée. Mais ce défaut n'est pas si couteux qu'il en a l'air, puisque MIDI n'est pas capable de supporter l'enharmonie de manière rigoureuse (cf. Cambouropoulos[6]).

Gammes

En outre, cette représentation est tout de même appréciable dans une grande majorité de cas. Elle permet par exemple de décrire les gammes sous forme d'une liste de degrés (ou plus exactement une liste d'intervalles chromatiques).

Gamme majeure (mode iononien)	`['1','2','3','4','5','6','7']`
Gamme mineure harmonique	`['1','2','3b','4','5','6b','7']`

FIG. 3.8 – Description des gammes par les degrés diatoniques

Dans l'*Allegretto en Do mineur D.915* de Schubert par exemple (cf. figure 3.6), les notes de la première mesure (en Do mineur) et celle de la cinquième (en Mi bémol majeur) donnent respectivement les analyses diatoniques suivantes :

- Mesure 1 : `['1','1','3b','5','1','3b','2','1','5']`
- Mesure 5 : `['1','1','3','5','1','3','2','1','5']`

Une réduction de ces deux listes par rapport aux gammes mineure et majeure donne le même chiffrage en degrés : `[1,1,3,5,1,3,2,1,5]`. On peut donc considérer ces deux motifs comme identiques sur le plan strictement diatonique.

Anticipations

Une des caractéristiques majeures du discours mélodique dans l'improvisation en jazz (dans le swing du moins) est le recours massif aux anticipations. L'anticipation consiste à jouer un peu en avance sur l'harmonie telle qu'elle est notée dans la grille, et contribue à cet effet de balancement propre au swing. L'anticipation, très souvent employée au endroits de modulation, a donc dans ce cas pour effet, du strict point de vue de la partition, de jouer une note à cheval sur deux tonalités locales. Or, l'analyse d'une note anticipée se fait toujours par rapport à la tonalité à laquelle elle se rapporte.

Nous avons implémenté au sein de notre système un jeu de règles très simples pour tenir compte des anticipations. Pour chaque note à cheval sur plusieurs tonalités locales, nous dirons que cette note est une anticipation si :

1. la durée de la note avant le changement de tonalité est inférieure ou égale à la durée après le changement,
2. (**et**) la durée de la note avant le changement de tonalité est inférieure ou égale à la durée de référence définie lors de l'étape de segmentation (cf.supra).

La figure **3.9** montre une anticipation de la tonalité du Mi bémol. L'anticipation est indiquée par la flèche. Elle est analysée dans la tonalité de Mi bémol majeur.

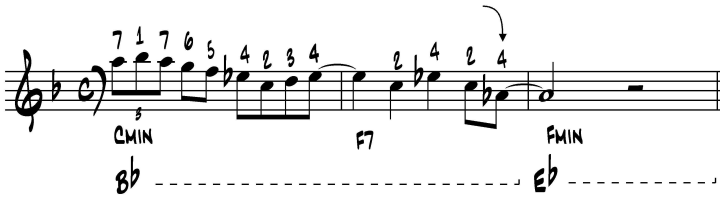


FIG. 3.9 – Anticipations : analyse diatonique par tonalités

Analyse diatonique selon les accords

Selon le même principe que l'analyse diatonique dans les tonalités locales, il est possible d'analyser chaque note par rapport à la(aux) fondamentale(s) de l'(des) accord(s) dans lequel(lesquels) elle se situe. Les mêmes méthodes sont employées, et les anticipations sont également prises en compte (les règles sont les mêmes).

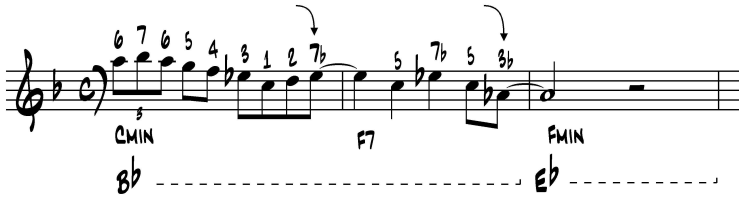


FIG. 3.10 – Anticipations : analyse diatonique par accords

3.3.4 Analyse de contour

La notion de contour interveint non seulement dans la perception de la mélodie mais également dans la détermination du phrasé. Dans sa thèse sur Charlie Parker[27], Tom Owens avait par exemple remarqué que Bird accentuait les maxima locaux de hauteur et attenuait les minima (cf. Rolland[38]). La figure 3.11 montre le pattern **ow5Cau** pourvu de ses extrama locaux de hauteur. Un croix indique un maximum, un losange un minimum.

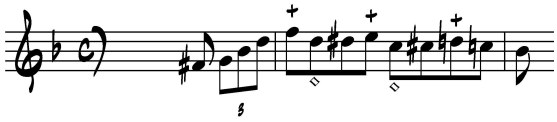


FIG. 3.11 – Extrema de hauteur sur le pattern **ow5Cau**

Notre analyse de contour considère cinq types de notes : débuts de phrases, fins de phrases, maxima de hauteurs, minima de hauteurs, et autre notes. Le principe général de cette analyse consiste à opérer une différentiation d'ordre 1 sur la séries des hauteurs. Chaque note est alors encadrée par deux intervalles, le premier avec la note qui la précède, le second avec celle qui la suit. Les extrema de hauteur sont alors repérés par deux intervalles chromatiques de sens contraires qui les encadrent. Il y a toutefois quelques exceptions parfois délicates à gérer.

Limites de phrases Si l'une des notes à différencier est située respectivement en début ou en fin de phrase, l'intervalle n'est calculé que du côté intérieur de la phrase.

Silences Si l'une des notes à différencier est un silence, l'intervalle est calculé avec la note précédent ou suivant le silence. Si le silence est long, il constitue un frontière entre deux phrases, et nous retombons dans le cas précédent : la note n'est pas comparée avec la note située au delà du silence par rapport à elle.

Petites notes Les appoggiatures et les grupetti sont fréquents en jazz. Ils interviennent en général comme éléments de phrasé et ne doivent pas, à notre sens, être considérés comme des mouvements mélodiques. Par exemple le Si bémol du triolet de la figure 3.9 n'est pas un maximum de hauteur mais un élément stylistque, une broderie typique dans le jazz. Du seul point de vue du contour mélodique ce triolet est équivalent à un La6 durant une noire. Nous avons implémenté un jeu de règles permettant de repérer les formes de broderies et appoggiatures couramment employées en jazz. Sur la figure 3.12, les notes marquées d'une flèche ne peuvent pas être des extrema locaux si les notes adjacentes sont de même hauteur, sauf dans le cas du troisième motif, où la première note n'est jamais un extrema de hauteur.

FIG. 3.12 – Extrema de hauteur : notes interdites

Un fois les exrema de hauteur identifiés, chaque note est pourvue de deux descripteurs supplémentaires, relatifs au contour mélodique :

1. Pente mélodique locale : mélodie ascendante, descendante, ou plate.
2. Distance par rapport au prochain extremum : distance discrète (en nombre de notes) et continue (durée métrique, mesurée en noires).

Le lecteur se reportera en annexe pour une analyse complète de contour mélodique sur le standard *Donna Lee*.

3.4 Données de sortie

Nous pouvons maintenant donner une description exhaustive des représentations musicales assumées par notre système. Chaque fichier MIDI complété de ses grille d'accords, forme et levée, est transformé en un objet de la classe MonoMidiFile, dont la structure est représentée en figure 3.13.

Le champ `MMF.notelist` contient une liste d'objets de la classe `note` telle que décrite en infra et complétée des attributs correspondant aux descripteurs introduits à la section précédente et de quelques autres décrits figure 3.14, où est représentée la quatrième note de la figure 3.9.

MonoMidiFile (object)	
`MMF.name`	Nom du fichier
`MMF.timediv`	Nombre de ticks MIDI par temps
`MMF.tempo`	Tempo (exprimé à la noire)
`MMF.key`	Tonalité
`MMF.signature`	Métrique
`MMF.form`	AABA, blues, etc...
`MMF.structure`	cf. appendice A
`MMF.chords`	Listes des accords
`MMF.phrases`	Coordonnées des phrases mélodiques
`MMF.refvalue`	Référence de durée (cf. segmentation, supra)
`MMF.analyse`	Liste des progressions repérées par l'analyse
`MMF.extrema`	Coordonnées des extrema de hauteur
`MMF.notes`	Liste des notes

FIG. 3.13 – Structure de l'objet MonoMidiFile

note (instance)		
`note.pitch`	Hauteur MIDI	79
`note.velocity`	Intensité MIDI	64
`note.position`	Position en ticks MIDI (120 pour une noire)	120
`note.duration`	Durée en ticks MIDI	60
`note.index`	Index de la note (commence à zéro)	3
`note.phrase`	Dans quelle phrase et à quel index	[0,3]
`note.inbar`	Dans quelle mesure et où	[0,120]
`note.rightint`	Intervalle à droite	-2
`note.leftint`	Intervalle à gauche	-2
`note.isextrem`	Type d'extrema	none
`note.dist2ext`	Distance au prochain extremeum	[3,180]
`note.slope`	Pente mélodique	down
`note.globdeg`	Degré dans la tonalité globale	N.C.
`note.seqdeg`	Tonalité locale et degré dans la tonalité	[BbM,6]
`note.chorddeg`	Accord local et dégré dans l'accord	[Cm,5]
`note.name`	Nom de la note	G5
`note.ratio`	Ratio de durée	1,5

FIG. 3.14 – Classe note enrichie par l'analyse

La méthode retournant le nom de la note essaie de tenir compte de l'enharmonie en situant la note par rapport à la tonalité locale. Si la note appartient à cette tonalité, le nom de la note est celui de la note dans la tonalité. Si elle

n'y appartient pas, la note sera *ou* naturelle *ou* diésée dans le cas d'une tonalité diésée, *ou* naturelle *ou* bémolisée dans le cas d'une tonalité ayant des bémols à la clés. Cette méthode génère tout de même un grand nombre d'erreurs dans le solfiage d'un fichier MIDI. Implémenter un algorithme de solfiage automatique nécessiterait de prendre en compte des formes plus globales comme des approches chromatiques par exemple.

Espace mémoire

L'enrichissement des données, évidemment, augmente considérablement la taille du corpus. Les fichiers MIDI (hexadécimaux) initiaux occupaient environ 400 Ko, contre 17 Mo pour la totalité du corpus enrichi, stocké dans un fichier texte. Cette valeur restant raisonnable au regard des performances d'un ordinateur, nous n'avons pas jugé nécessaire, dans l'immédiat, de réfléchir à une stratégie de réduction des données.

En outre, certains descripteurs (notamment ceux liés au contour) dépendent de structures plus globales que le simple niveau de la note. Il faut au minimum prendre en compte un segment de la séquence contenant un extremum à gauche et un extremum à droite. De plus, leur calcul fait intervenir une combinatoire de cas non négligeable. Générer dynamiquement ces descripteurs en fonction des besoins des algorithmes de localisation dégraderait donc sévèrement leurs performances. Nous préférons dès lors les calculer une bonne fois pour toutes, quitte à occupper un espace mémoire important.

Constitution de sous-corpus

Il peut s'avérer utile de procéder aux analyses sur une portion réduite du corpus total. Le temps de calcul s'en trouve réduit, et cette technique permet par ailleurs de cibler les recherches. Nous pouvons par exemple décider de n'explorer que les morceaux de forme AABA, où que ceux d'un certain groupe de tonalités. L'intérêt musicologique est patent. Il peut être extrêment intéressant d'observer comment évolue l'improvisation lorsqu'on passe d'un tempo médium à un tempo rapide. On pourrait par exemple chercher à savoir si le recours à des patterns quasiment inchangés est plus systématique sur les tempi élevés.

Lorsque nous générons le corpus enrichi le système établit en parallèle une cartographie des objets MonoMidiFile en fonction de leurs attributs de tempo, tonalité, forme et valeur métrique de réfèrence. Cela permet ensuite de constituer facilement des sous-corpus en spécifiant des contraintes sur ces valeurs (cf. Annexe E).

3.5 Patterns

Les patterns que nous cherchons à localiser dans les improvisations de Parker proviennent de trois origines différentes :

1. patterns découverts par Owens dans sa thèse (portant le préfixe « *ow* »),

2. patterns découverts par Pierre-Yves Rolland grâce à son système **Imprology**[38] (portant le préfixe « *pat* »),
3. et enfin patterns suggérés par nos soins (portant le préfixe « *mypat* »), inférés à partir de notre propre connaissance de Parker, acquise par l'étude de ses solos pour le travail de l'improvisation.

Les patterns proviennent de fichiers MIDI, et, comme les fichiers du corpus, sont convertis par le système en objets MonoMidiFile avec le même degré d'information. Une liste détaillée de ces patterns est disponible en annexe.

Chapitre 4

Miroirs

« *Il n'y a rien qui soit un, rien qui soit multiple,*
tout est multiplicités. » - Gilles Deleuze

La recherche de similarités dans la musique représentée sous forme symbolique se rattache à un vaste champ de l'informatique théorique, souvent désigné comme l'*algorithmique des séquences*, ou encore *algorithmique des chaînes*. Les premiers travaux réalisés dans ce domaine concernaient le traitement des chaînes de caractères, et la terminologie usuelle dérive du même champ sémantique. On parle ainsi de mot, d'alphabet, préfixes, suffixes... Dans ce chapitre nous introduisons quelques définitions élémentaires liées à l'algorithmique des chaînes, et nous présentons quelques méthodes pour mesurer la similarité entre deux séquences.

4.1 Généralités

4.1.1 Terminologie et notations

Définition 4.1.1 Le formalisme couramment utilisé désigne chaque élément distinct d'une séquence par le terme de *symbole* appartenant à un *alphabet* donné. Un alphabet, noté Σ peut alors être défini comme un ensemble *fini* de symboles. Une *séquence*, ou *mot*, sur Σ, est alors une suite *finie* de symboles de Σ en relation syntagmatique[1]. On note Σ^* l'ensemble des séquences sur Σ. Par exemple, pour un alphabet $\Sigma = \{a, b, c, d, e, f, g, h, i, j\}$, la séquence $X = abdegh$ est un mot sur Σ. On notera $|X|$ la longueur de cette séquence.

Les séquences étant en général indexées, il est fréquent de repérer chacun de ces éléments par sa coordonnée. Par exemple dans la séquence $s = abdegh$, on aura $X[0] = a,\ X[1] = b$, etc.

[1] Ou encore relation de succession indicielle stricte (cf. Lartillot[19], p.117)

Dans le cas de l'analyse musicale (du moins l'analyse mélodique), les symboles correspondent évidemment aux notes elles-mêmes. Il est alors pertinent de généraliser la description précédente en utilisant un alphabet de symboles *multi-attributs*. Les éléments de la séquence deviennent alors des vecteurs d'un espace n-dimensionnel, et la comparaison entre séquences se fait alors attribut par attribut :

$$X1\,[i]\,.attributN \leftrightarrow X2\,[j]\,.attributN$$

Pour les notes, les attributs de base sont la hauteur, la durée, l'intensité, et la position. Les attributs supplémentaires, correspondant à des descripteurs de plus haut niveau, ont été présentés en 3.4.

Définition 4.1.2 Etant donnés un alphabet Σ et deux séquences X et F sur Σ, on dira que F est un *facteur* de X si et seulement si il existe deux séquences Y et Z sur Σ telles que :

$$X = YFZ$$

Un facteur F d'une séquence X est répérable par un couple d'entiers (i, l) où i est le rang du premier élément de F et l sa longueur. On a alors $l = m - i + 1$, où m est le rang ternminal de F. Par exemple, pour $X = abdegh$ et $F = bde$, on aura $l = 3$, $i = 1$ et $m = 4$. On notera par ailleurs :

$$F = bde = X\,[[1, 3]] = X\,[1..4]$$

Définition 4.1.3 Etant donnés un alphabet Σ et une séquence X et F sur Σ, on dira que F est un *préfixe* de X si et seulement si il existe une séquence Y sur Σ telle que :

$$X = FY$$

De façon évidente, on dira que F est un *suffixe* de X si :

$$X = YF$$

4.1.2 Patterns, recherche d'occurences

Nous nons reportons ici à la définition que donne Rolland[38] d'un *pattern*. Etant donné un alphabet Σ, un *pattern* est mot sur Σ représentant de manière *intentionnelle* un ensemble de séquences de Σ^*. Réciproquement, cet ensemble peut être vu comme la représentation *extentionnelle* (ou *extention*) du pattern considéré.

Définition 4.1.4 Etant donné un alphabet Σ, on appelle *fonction d'équipollence* toute fonction booléenne sur $\Sigma^* \times \Sigma^*$. Deux séquences X et X' seront alors dites *équipollentes* si et seulement si $Eq(X, X') = VRAI$, où Eq est une fonction d'équipollence. En pratique, une fontion d'équipollence est sensée traduire la « ressemblance » entre deux séquences selon des critères donnés. Soit par exemple $\Sigma = \{a, b, c, d, ..., z\}$ et la fonction d'équipolence suivante :

$$Eq(X, X') = \ VRAI \Leftrightarrow \begin{cases} |X| = |Y| = l \\ \forall i \in \{1..l\}, \exists j \in \{1..l\}, X[i] = Y[j] \end{cases}$$

Alors deux mots seront dits équipollents si et seulement si ils sont anagrammes l'un de l'autre.

Définition 4.1.5 Etant donnés un alphabet Σ, un pattern P qui est un mot sur Σ, et une séquence X de Σ^* et F un facteur de X, on dira que F est une occurence de P dans X si et seulement si $Eq(F, P) = VRAI$. Par extention, rechercher toutes les occurences de P dans X, c'est chercher tous les facteurs F de X tels que $Eq(F, P) = VRAI$.

4.1.3 Extraction de patterns

Définition 4.1.6 Etant donné un alphabet Σ, on appelle bloc (ou encore catégorie, ou cluster), tout ensemble B de séquences sur Σ, tels que :

$$\forall (X, Y) \in B^2, Eq(X, Y) = VRAI$$

Il est d'usage, au sein d'un même bloc, d'identifier une séquence comme *représentant* (ou *prototype*) de l'ensemble. Lorsqu'on dispose d'une mesure non binaire de la similarité, il est par exemple courant de choisir comme prototype du cluster la séquence qui minimise la somme des distances avec les autres éléments de l'ensemble, ou encore celle qui possède le plus grand nombre de voisins dans un entourage fixé. Le prototype est alors un pattern au sens d'une représentation intentionnelle du cluster, lequel constitue sa représentation extentionnelle, soit l'ensemble des occurences du pattern dans une séquence donnée.

De façon très schématique l'extraction de patterns consiste alors à rechercher, dans une séquence ou dans un ensemble de séquences (*corpus*) donné, un ou plusieurs patterns satisfaisant des critères fixés à l'avance. Ces critères peuvent porter soit sur les patterns eux-mêmes (par exemple longueur minimale et maximale, cf. Rolland[38]), soit sur la qualité des clusters qu'ils représentent (taille et pertinence du cluster, qualité des occurences, ratio de recouvrement, cf. Rolland[38] et Cambouropoulos et al.[8]).

L'extraction de patterns n'étant pas abordée dans ce mémoire, nous ne détaillons pas davantage ces méthodes ni les algorithmes qu'elles sous-tendent. Le lecteur avide pourra toutefois se reporter aux références cités dans cette section ainsi qu'au chapitre 2.

4.2 Introduction à la similarité

Nous introduisons ici quelques paradigmes de comparaison entre séquences. Dans tout ce chapitre, nous parlerons de comparaison *globale* (par opposition à une comparaison *locale*, telle que définie au chapitre 5). Nous considérons pour cela un alphabet Σ et deux séquences X et Y sur Σ. La comparaison est dite globale dans la mesure ou nous cherchons à mettre en correspondance l'intégralité de la séquence X avec l'intégralité de la séquence Y. Par opposition, une comparaison locale s'attache elle à comparer deux à deux des facteurs des deux séquences, ces facteurs étant alors comparés dans leur globalité.

Si la notion de *fonction d'équipollence* introduite dans la section précédente est utile pour expliciter les mécanismes généraux de comparaison, de localisation et d'extraction de patterns, elle s'avère en revanche trop abstraite pour être utilisée en tant que telle. Nous lui substituons donc des *fonctions de comparaison*, que nous détaillons ici.

4.2.1 Fonctions binaires

Dans un certain sens, ces fonctions ne font que déplacer la notion d'équipollence : Ou bien les deux séquences comparées sont similaires, ou bien elles ne le sont pas. Leur intérêt réside donc uniquement dans l'explicitation des critères qui fondent cette similarité.

Définition 4.2.1 (Similarité α) Etant données deux séquences X et Y de même longueur l, on définit l'ensemble des positions d'identités $\cap_{XY}$ par :

$$\cap_{XY} = \{i \in \{1..l\}, X[i] = Y[i]\}$$

On dit alors des séquences X et Y qu'elles sont α-similaires si et seulement si :

$$|\cap_{XY}| \geq \alpha$$

La similarité α recouvre la notion d'identité partielle. Deux séquences de même longueur l seront α-similaires si et seulement si on peut passer de l'une à l'autre par l'intermédiaire d'un nombre de substitutions inférieur à $l - \alpha$.

Définition 4.2.2 (Similarité δ) Deux séquences X et Y de même longueur l sont dites δ-similaires si et seulement si :

$$\forall i \in \{1..l\}, d(X[i], Y[i]) \leq \delta$$

où d est une distance élémentaire entre deux symboles de Σ (par exemple, dans le cas d'une utilisation musicale, le nombre de demi-tons). Une telle définition de la similarité se rapporte à la notion de norme infinie, l'« écart » entre deux séquences ne devant jamais dépasser un certain seuil δ. Elle présente l'inconvénient de ne pas rendre compte de la fréquence de ces écarts. La figure 4.1 montre trois segments δ-similaires avec $\delta = 1$ et d l'écart de hauteur mesuré en

demi-tons. Pourtant, la fréquence des écarts est bien plus élevée entre les motifs A et C qu'entre A et B, et perceptiblement, C est « plus loin » de A que ne l'est B. La similarité γ permet de remédier à cet inconvénient. Cette fonction de comparaison a été implémentée dans notre système. Nous en discuterons, au chapitre suivant, les avantages et les inconvénients, ainsi que la pertinence dans le cadre de nos recherches.

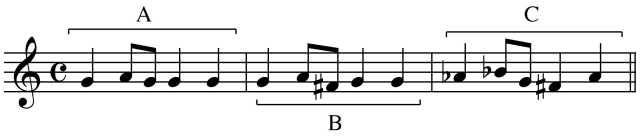


FIG. 4.1 – Exemples de motifs δ-similaires

Définition 4.2.3 (Similarité γ) Deux séquences X et Y de même longueur l sont dites δ-similaires si et seulement si :

$$\sum_{i=1}^{l} d\left(X\left[i\right] - Y\left[i\right]\right) \leq \gamma$$

Cette notion de la similarité se rapporte cette fois-ci à la norme 1. Sur la figure 4.1, les motifs A et B sont γ-similaires avec $\gamma = 1$, tandis que A et C ne le sont qu'avec $\gamma = 4$!

Limitations

L'inconvénient majeur de la comparaison entre séquences à l'aide de fonctions binaires est qu'on ne peut pas distinguer deux occurences d'un même pattern selon leur qualité. De même qu'une fonction d'équipollence, ces fonctions ne permettent pas de quantifier la similarité. Nous devons pour cela avoir recours à des distances.

Nous proposons dans un premier temps de mesurer la dissemblance entre deux séquences à l'aide de fonction dites *fonctions de dissimilitude*. De telles fonctions s'apparentent à des distances, mais ne vérifient pas nécessairement toutes les propriétés métriques qui les caractérisent. Dans le cas de l'analyse musicale par exemple, l'inégalité triangulaire est, d'après Mongeau et Sankoff[25], rarement vérifiée, ce qui peut se traduire par l'absence de transitivité dans la notion de ressemblance entre segments mélodiques.

Les fonctions de dissimilitude peuvent être scindées en deux catégories :

1. Les *modèles à structure de correspondance fixe* (ou *modèles de Hamming*), où les séquences comparées ont même longueur. Les plus utilisés sont la distance euclidienne, la distance de Minkowski, et la distance de Hamming (cf. Rolland[38], p.65), dont Cope utilise une variante dans son système

EMI[12]. N'ayant pas fait recours à ce genre de modèles, nous ne les détaillons pas davantage ici.

2. Les *modèles à structure de correspondance variable* (ou *modèles d'édition*) où les séquences comparées peuvent être de tailles différentes. Nous explicitons dans la section suivante les prinicpes généraux de ces méthodes de comparaison.

4.2.2 Distances d'éditions

Le calcul d'une distance d'édition entre deux séquences repose sur la détermination du nombre minimal d'opérations élémentaires nécessaires à la transformation de la première séquence en la seconde. En règle générale, ces transformations sont l'insertion ou la délétion d'un symbole, ainsi que la substitution d'un symbole à un autre. D'autres types de transformations seront toutefois enviagés en infra.

Définition 4.2.4 (appariement) Lors de la comparaison de deux séquences, nous appelons *appariement* toute mise en correspondance, à travers une opération élémentaire, d'un facteur d'une des séquences (éventuellement vide) avec un facteur de la seconde (éventuellement vide).

Définition 4.2.5 (alignement) Nous désignons par *alignement* entre deux séquences toute collection ordonnée d'appariements telle que les concaténations des facteurs impliqués dans chaque appariement soient égales aux séquences comparées. Nous cherchons par exemple à comparer les mots *endogene* et *exogenes*. Un modèle à structure de correspondance fixe calculerait obligatoirement le coût associé l'alignement suivant :

$$\begin{bmatrix} e & n & d & o & g & e & n & e \\ | & | & | & | & | & | & | & | \\ e & x & o & g & e & n & e & s \end{bmatrix}$$

obtenu par concaténation des appariements suivants :

$$\begin{bmatrix} e \\ | \\ e \end{bmatrix}, \begin{bmatrix} n \\ | \\ x \end{bmatrix}, \begin{bmatrix} d \\ | \\ o \end{bmatrix}, \begin{bmatrix} o \\ | \\ g \end{bmatrix}, \begin{bmatrix} g \\ | \\ e \end{bmatrix}, \begin{bmatrix} e \\ | \\ n \end{bmatrix}, \begin{bmatrix} n \\ | \\ e \end{bmatrix}, \begin{bmatrix} e \\ | \\ s \end{bmatrix}$$

Tandis qu'un modèle d'édition permettrait l'obtention d'un alignement plus conforme à la notion intuitive de la similarité entre ces deux mots :

$$\begin{bmatrix} e & n & d & o & g & e & n & e & - \\ | & | & | & | & | & | & | & | & | \\ e & x & - & o & g & e & n & e & s \end{bmatrix}$$

formé des appariements suivants :

$$\begin{bmatrix} e \\ | \\ e \end{bmatrix}, \begin{bmatrix} n \\ | \\ x \end{bmatrix}, \begin{bmatrix} d \\ | \\ - \end{bmatrix}, \begin{bmatrix} o \\ | \\ o \end{bmatrix}, \begin{bmatrix} g \\ | \\ g \end{bmatrix}, \begin{bmatrix} e \\ | \\ e \end{bmatrix}, \begin{bmatrix} n \\ | \\ n \end{bmatrix}, \begin{bmatrix} e \\ | \\ e \end{bmatrix}, \begin{bmatrix} - \\ | \\ s \end{bmatrix}$$

soit 1 substitution, 1 délétion, 1 insertion, et 6 identités (forme dégénérée de la subsitution).

Pour pouvoir dire que le second alignement est meilleur que le premier, il nous faut un moyen de les comparer, c'est-à-dire, calculer leurs « valeurs ». La méthode la plus évidente consiste naturellement à définir la valeur d'un alignement comme la somme des valeurs des appariements le constituant. Mais qu'entend-on exactement par *valeur d'un appariement* ?

On trouve dans la littérature deux grandes catégories de modèles : les *modèles de similitude* et les *modèles de dissimilitude*. Pour les premiers, la valeur d'un appariement est d'autant plus élevée que leur ressemblance, dont les critères sont à préciser, est grande. On associe alors à chaque appariement une mesure appelée *fonction de contribution*. Dans le cas d'un modèle de dissimilitude, la valeur d'un appariement est d'autant plus élevée que leur ressemblance est faible. On parle alors de *fonction de coût*.

Il existe une dualité évidente entre fonctions similitude et fonctions de dissimilitude. Si par exemple f est une fonction de dissimilitude sur $\Sigma^* \times \Sigma^*$, alors les fonctions :

$$g\,(X,Y) = k_1 - k_2 \cdot f(X,Y)$$

et :

$$h(X,Y) = \frac{1}{1 + k \cdot f\,(X,Y)^{\alpha}}$$

sont des fonctions de similitude et vice versa. Toutefois, Rolland[38] fait remarquer que si tout problème solvable avec des fonctions de dissimilitude peut aussi être résolu avec des fonctions de similitude, l'inverse n'est pas vrai pour autant (cf. Watterman[45], chap. 9). En outre, Mongeau et Sankoff[25] suggèrent que l'utilisation de distances dans la localisation de patterns conduit le plus souvent à des alignements optimaux peu pertinents, les distances minimales étant atteintes pour des alignements de longueur 1, appariant deux symboles identiques. Dans la suite nous nous limiterons donc à des fonctions de similitudes. Nous terminons cette section par une présentation hiérarchique des modèles d'édition les plus couramment utilisés, du plus simple au plus complexe.

Modèle non valué

Le modèle d'édition le plus simple utilisant des fonctions de similitude a été introduit en 1965 par Levenshtein[21]. La simimitude entre deux séquences y est définie comme le nombre maximum d'appariements à l'identique entre les séquences. Pour notre exemple précédent, la similitude entre les mots *endogene* et *exogenes* aurait donc pour valeur 6. Un tel modèle est dit *non-valué* dans la mesure ou la valeur d'une transformation est basée sur un comptage des appariements, lesquels ne possèdent pas de valeur propre.

Modèle valué à fonctions de contribution fixes

Dans ce type de modèle on associe une valeur fixe à chaque type d'appariement (inserstion, délétion, substitution). La valeur $v(A)$ d'un alignement $A = \{a_1 a_2 ... a_n\}$ est alors la somme des coûts de contribution associés aux types d'appariements $(a_i)_{1 \leq i \leq n}$ qui le constituent :

$$v(A) = \sum_{i=1}^{n} c(t(a_i))$$

où $c(t(a_i))$ est la contribution du i$^{\text{ème}}$ appariement, ne dépendant que du type d'appariement $t(a_i)$. Pour ce type de modèle Rolland[38] suggère les contributions suivantes :

- $+4$ pour toute identité,
- -3 pour toute substitution,
- -2 pour toute insertion ou délétion.

La valeur de l'alignement représenté figure 4.2 est alors 17.

$$\begin{bmatrix} e & n & d & o & g & e & n & e & - \\ | & | & | & | & | & | & | & | & | \\ e & x & - & o & g & e & n & e & s \\ +4 & -3 & -2 & +4 & +4 & +4 & +4 & +4 & -2 \end{bmatrix}$$

FIG. 4.2 – Alignement optimal avec un modèle valué à coûts de contribution fixes

Modèle valué à fonctions de contribution variables

Dans ce type de modèle les contributions associées à chaque appariement dépendent non seulement du type d'appariement $(a_i)_{1 \leq i \leq n}$ mais aussi du couple $\left(F_i^1, F_i^2\right)$de facteurs impliqués dans chaque appariement. On aura alors :

$$v(A) = \sum_{i=1}^{n} c\left(t(a_i), F_i^1, F_i^2\right)$$

Reprenons notre exemple précédents avec les fonction de contributions suivantes :

- $+4$ pour toute identité,
- $+4 - d\left(F_i^1, F_i^2\right)$ pour chaque substitution, où d est le nombre de lettres de l'alphabet séparant F_i^1 et F_i^2,
- -2 pour toute insertion ou délétion.

L'alignement de la figure 4.2 possède selon ces critères une valeur égale à 14, et n'est plus optimal. L'alignement optimal (de valeur 16) est représenté figure 4.3.

Ainsi que le remarque Rolland[38], l'alignement optimal entre deux séquences ne dépend pas uniquement des types d'appariement utilisés (dans les exemples

$$\begin{bmatrix} e & n & d & - & o & g & e & n & e & - \\ | & | & | & | & | & | & | & | & | & | \\ e & - & - & x & o & g & e & n & e & s \\ +4 & -2 & -2 & -2 & +4 & +4 & +4 & +4 & +4 & -2 \end{bmatrix}$$

FIG. 4.3 – Alignement optimal avec un modèle valué à coûts de contribution variables

précédents : insertions, délétions et substitutions), mais est fortement déterminé par les fonctions de contributions qui leur sont associées.

Modèle valué multi-descriptions

Ce modèle, que nous utilisons dans notre système, est une simple généralisation du modèle précédent au cas où les symboles appareillés appartiennent à un espace multi-dimensionnel, comme il en va par exemple des notes d'une séquence mélodique, telles que définies et représentées au sein de notre système (cf. figure 3.14).

Dans le cas de la substitution d'une note à une autre par exemple, la contribution de l'opération (*sub*) s'écrit :

$$c\left(sub\left(x,y\right)\right) = \sum_{i=1}^{p} c_i\left(sub, x.attribut\left[i\right], y.attribut\left[i\right]\right)$$

où p est le nombre d'attributs de la note et c_i la contribution d'une substitution, relativement au i$^{\text{ème}}$ attribut. Une description plus détaillée des fonctions de contribution utilisées par notre système fera l'objet de la section suivante.

4.2.3 Algorithme de comparaison

Nous présentons ici un algorithme pour la détermination de l'aligment optimal entre deux séquences. Nous nons plaçons dans le cadre d'un modèle de similitude et nous limitons dans un premier temps aux trois types d'édition les plus courants (insertions, délétions, sustitutions). Cette limitation nous assure de toujours appareiller au plus un symbole d'une séquence avec au plus un symbole de l'autre. Nous supposons en outre que nous disposons d'une fonction de contribution c prenant en entrée un ou deux arguments (le facteur vide sera noté λ) et retournant la valeur de l'appariement réunissant ces deux symboles (cas d'une substitutions) ou d'un symbole avec le facteur vide (cas d'un insertion ou délétion) :

$$\begin{gathered} c\left(x,y\right) \rightarrow \mathbb{R} \\ c\left(x,\lambda\right) \rightarrow \mathbb{R} \\ c\left(\lambda,y\right) \rightarrow \mathbb{R} \end{gathered}$$

Comparaison :
Globale, locale ou « loco-globale » ?

Trois types de comparaison entre séquences sont en général distingués dans le littérature :

1. Dans la comparison *globale*, on cherche le meilleur alignement (celui qui maximise la fonction de simimlitude) entre deux séquences X et Y. L'alignement optimal fera toujours apparaître l'intégralité des séquences X et Y.
2. Dans la comparaison *locale*, on cherche le meilleur alignement entre un facteur de X et un facteur de Y. Cette méthode s'attache avant tout aux similarités locales entre séquences. L'alignement optimal ne fera apparaître qu'une partie des séquences X et Y.
3. Dans la comparaison « *loco-globale* », on cherche le meilleur alignement entre une séquence X et un facteur de Y. L'alignement optimal fera apparaître l'intégralité de la séquence X et une partie de la séquence Y.

Soient par exemple à comparer les mots *maladroit* et *maltais*. Nous nous plaçons dans le cadre du modèle valué à fonctions de contribution fixes présenté dans la section précédente ($+4$ pour toute identité, -3 pour toute substitution, -2 pour toute insertion ou remplacement). Les alignements optimaux et leurs valeurs respectives selon les trois méthodes de comparaison sont présentés figure 4.4.

$$\begin{bmatrix} m & a & l & - & a & d & r & o & i & t \\ | & | & | & | & | & | & | & | & | & | \\ m & a & l & t & a & - & - & - & i & s \end{bmatrix}$$

Comparaison globale : $Valeur = 9$

$$\begin{bmatrix} m & a & l & - & a \\ | & | & | & | & | \\ m & a & l & t & a \end{bmatrix}$$

Comparaison locale : $Valeur = 14$

$$\begin{bmatrix} m & a & l & - & a & d & r \\ | & | & | & | & | & | & | \\ m & a & l & t & a & i & s \end{bmatrix}$$

Comparaison loco-globale : $Valeur = 8$

FIG. 4.4 – Alignement optimaux en fonction du type de comparaison : globale, locale, loco-globale.

La comparaison globale nécessite le calcul de tous les alignements possibles entres les deux séquences. Soient m et n les longueurs respectives des séquences

X et Y à comparer. La complexité de l'algorithme peut être ramenée à $\mathcal{O}(m \cdot n)$ grâce à des méthodes de programmation dynamique (cf. infra).

Pour la comparaison loco-globale, un approche naïve consisterait à énumérer tous les facteurs de Y et à les comparer globalement avec la séquence X. On peut montrer que la complexité d'un tel aglorithme naïf est en $\mathcal{O}\left(m \cdot n^3\right)$. Une approche alternative, basée sur la programmation dynamique, permet de ramener la complexité en temps à $\mathcal{O}(m \cdot n)$ (cf. Crochemore et Ritter[13], Watterman[45], et Rolland[38]).

En ce qui concerne enfin la comparaison locale, une approche naïve serait d'énumérer toutes les paires possibles de facteurs de X et de Y, puis à comparer ces facteurs globalement. La complexité de l'algorithme serait alors en $\mathcal{O}\left(m^3 \cdot n^3\right)$. Là encore, des méthodes basées sur la programmation dynamique permettent de ramener la complexité en temps à $\mathcal{O}(m \cdot n)$ (cf. Smith et Waterman[42], Kruskal et Sankov[18], et Rolland[38]).

N.B. Ces valeurs de complexité en temps ne sont valables que si on se limite aux trois types d'éditions fondamentaux : inserstions, délétions et substitutions. Nous envisagerons en infra d'autres types de transformations et discuterons alors des changements éventuels de complexité qui en résultent.

Les trois méthodes de comparaison sont en fait très similaires dès qu'on fait appel à une approche basée sur la programmation dynamique : Il s'agit de toujours de construire récursivement une matrice de similarité. Au cours de ce calcul les éléments de la matrice déjà calculés interviennent dans le calcul des suivants. Les processus de calcul sont les mêmes pour les trois types de comparaison, seules changent les conditions initiales et la signification des éléments de la matrice. En vue de la localisation de patterns nous avons opté pour une comparaison locale, nous recherchons donc les meilleurs alignements entre un facteur d'un pattern donné et un facteur d'un corpus de séquences. Ce choix peut surprendre : il serait plus intuitif d'avoir recours à une comparaison loco-globale entre un pattern donné et un corpus de séquences. Toutefois, dans le cas de patterns relativement longs, il est possible que seule une partie significative du pattern suffise, de manière perceptible, à en identifier une occurence. Dans ce cas, la comparaison loco-globale qui, en cherchant à « imposer » l'intégralité d'un pattern dans une séquence, peut nettement dégrader la valeur d'une occurence (cf. figure 4.4), n'est peut-être pas le meilleur choix.

Matrice de Similarité

Définition 4.2.6 Soient Σ un alphabet et X et Y deux séquences sur Σ, de tailles respectives m et n, à comparer *localement*. La matrice S de dimension $(m+1) \times (n+1)$ et de terme général :

$$S_{i,j} = \max \left\{ \begin{array}{c} 0 \\ \max \left\{ SG\left(X\left[k..i\right], X\left[l..j\right]\right), 1 \leq k \leq i, 1 \leq l \leq j \right\} \end{array} \right.$$

où SG désigne la similarité *globale* entre deux séquences, est appelée *matrice de similarité locale* entre les séquences X et Y. Chaque élément $S_{i,j}$ de la matrice est alors la valeur maximale de tous les alignements entre un suffixe de $X[1..i]$ et un suffixe de $Y[1..j]$, et la valeur de la similarité locale entre entre X et Y est **le plus grand élément de $\mathbf{S_{i,j}}$**.

Algorithme

La matrice de similarité entre les séquences X et Y est construite de la façon suivante :

$$\forall i \in \{0..m\}, S_{i,0} = 0$$
$$\forall j \in \{0..n\}, S_{0,j} = 0$$

$$\forall (i,j) \in \{1..m\} \times \{1..n\},$$
$$S_{i,j} = \max \begin{cases} 0 & \\ c(X[i], Y[j]) + S_{i-1,j-1} & (substitution) \\ c(X[i], \lambda) + S_{i-1,j} & (deletion) \\ c(\lambda, Y[j]) + S_{i,j-1} & (insertion) \end{cases}$$

Si le plus grand élément de la matrice nous donne la valeur de la similarité locale entre X et Y, il ne dit rien en revanche de l'alignement optimal entre les séquences. Celui-ci peut toutefois être obtenu en mémorisant, à chaque calcul d'un élément de la matrice, le type d'édition (substitution, délétion ou insertion) qui réalise la maximisation. En partant du plus grand élément de S, il suffit alors de « remonter » dans la matrice, en fonction des types d'éditions rencontrés, et de s'arrêter dès qu'on atteint une valeur nulle.

A titre d'exemple, l'alignement local de la figure 4.4 à été obtenu avec la matrice de similarité de la figure 4.5 (λ désigne le facteur vide). Pour chaque élément, l'opération (lorqu'elle existe) qui réalise la maximisation lors de son calcul figure également dans la matrice. Les éléments en gras représentent le « trajet » à parcourir dans la matrice pour en déduire l'alignement optimal.

	λ	**m**	**a**	**l**	**t**	**a**	**i**	**s**
λ	0	0	0	0	0	0	0	0
m	0	**Sub 4**	*Ins* 2	0	0	0	0	0
a	0	*Del* 2	**Sub 8**	*Ins* 6	*Ins* 4	*Sub* 4	*Ins* 2	0
l	0	0	*Del* 6	**Sub 12**	**Ins 10**	*Ins* 8	*Ins* 6	*Ins* 4
a	0	0	*Sub* 4	*Del* 10	*Sub* 9	**Sub 14**	*Ins* 12	*Ins* 10
d	0	0	*Del* 2	*Del* 8	*Sub* 7	*Del* 12	*Sub* 11	*Sub* 9
r	0	0	0	*Del* 6	*Sub* 5	*Del* 10	*Sub* 9	*Sub* 8
o	0	0	0	*Del* 4	*Sub* 3	*Del* 8	*Sub* 7	*Sub* 6
i	0	0	0	*Del* 2	*Sub* 1	*Del* 6	*Sub* 12	*Ins* 10
t	0	0	0	0	*Sub* 6	*Del* 4	*Del* 10	*Sub* 9

FIG. 4.5 – Matrice de Similarité entre les mots *maladroit* et *maltais*

Le plus grand élément de la matrice est 14. Il correspond à l'appariement du second *a* de *maladroit* avec le second *a* de *maltais*. L'opération est une subtitution[2] : on remonte donc à la case située immédiatement en haut à gauche. Elle correspond à une insertion (le *t* de *maltais*). On passe donc alors à la case de gauche, correspondant de nouveau à une substitution (le premier *a* de *maladroit* avec le premier *a* de *maltais*), etc... L'algorithme 3 résume la méthode générale de calcul de similarité locale.

Au final, on aura donc eu 4 subsitutions à l'identique, et 1 insertion pour passer de *maladroit* à *maltais*. L'alignement optimal est celui de la figure 4.4.

Algorithm 3 Calcul de similarite locale entre deux séquences

```
initialiser matrice
élément_en_cours ← matrice[1,1]
faire
{
  calculer élément_en_cours
  mémoriser type_d'édition
  incrémenter élément_en_cours
} jusqu'à élément_en_cours = dernier_élément
alignement ← {}
valeur_en_cours ← max(matrice)
tant_que valeur_en_cours ≠ 0
{
  alignement.ajouter(type_d'édition_en_cours)
  remonter_dans matrice
          selon type_d'édition_en_cours
        jusqu'à nouvelle_case
  valeur_en_cours ← valeur_de(nouvelle_case)
}
retourner max(matrice)
retourner alignement
```

Fragmentation, consolidation

Ainsi que suggéré par Mongeau et Sankoff[25], il apparaît nécessaire, dans la comparaison de séquences musicales, d'autoriser deux types d'édition supplémentaires, similaires aux *compressions* et *expansions* utilisées pour le traitement dans la parole dans le cadre du formalisme connu sous le nom de *Dynamic Time-Warping* (cf. Kruskal et Liberman[17]). Ces transformations, appelées respectivement *consolidations* et *fragmentations*, permettent respectivement le remplacement de plusieurs notes par une seule et d'une note par plusieurs. Mon-

[2]Nous appelons, dans cet exemple et dans la suite de ce rapport, *susbtitution* tout appariement de deux symboles non vides, qu'ils soient identiques ou non. Les fonctions de contribution se chargent d'attribuer des valeurs différentes à l'appariement, selon l'une ou l'autre de ces deux dernières modalités.

geau et Sankoff font remarquer que le remplacement, par exemple, d'une ronde par quatre noires de même hauteur devrait être d'un coût inférieur à celui d'une subsitution et de trois insertions. La figure 4.6, qui donne un exmple de fragmentation dans le domaine musical, met clairement en évidence le besoin de ces deux nouvelles transformations.

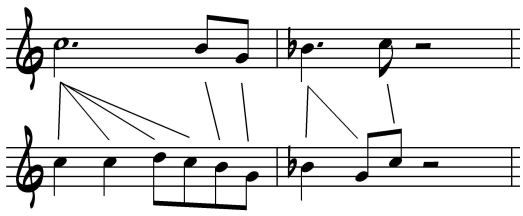


FIG. 4.6 – Exemple de fragmentation dans le domaine musical

Pour autoriser les consolidations et fragmentations au sein de notre algorithme de comparaison, il nous faut tout d'abord étendre notre fonction de contribution c à une fonction acceptant un, deux, ou plus de deux arguments :

$$\begin{array}{cc} c(x,y) \to \mathbb{R} & (substitution) \\ c(x,\lambda) \to \mathbb{R} & (deletion) \\ c(\lambda,y) \to \mathbb{R} & (insertion) \\ c(x,y_1..y_n) \to \mathbb{R} & (fragmentation) \\ c(x_1..x_p,y) \to \mathbb{R} & (consolidation) \end{array}$$

La matrice de similarité entre les séquences X et Y est alors construite de la façon suivante :

$$\forall i \in \{0..m\}, S_{i,0} = 0$$
$$\forall j \in \{0..n\}, S_{0,j} = 0$$

$$\forall (i,j) \in \{1..m\} \times \{1..n\},$$

$$S_{i,j} = \max \begin{cases} 0 & \\ c(X[i], Y[j]) + S_{i-1,j-1} & (sub) \\ c(X[i], \lambda) + S_{i-1,j} & (del) \\ c(\lambda, Y[j]) + S_{i,j-1} & (ins) \\ \{c(X[i], Y[j-k+1]..Y[j]) + S_{i-1,j-k}\}, 2 \leq k \leq j & (frag) \\ \{c(X[i-l+1]..X[i], Y[j]) + S_{i-l,j-1}\}, 2 \leq l \leq i & (cons) \end{cases}$$

Evidemment, ce raffinement augmente la complexité de l'algorithme, puisqu'on ne sait pas à l'avance combien de symboles seront consolidés en un seul, ou en combien de symboles en sera fragmenté un. A chaque étape du calcul la maximisation ne se fait plus entre trois possibilités comme avant, mais entre $i + j + 1$ possibilités. La complexité en temps de l'algorithme passe donc de $\mathcal{O}(m \cdot n)$ à $\mathcal{O}(m \cdot n \cdot (m+n))$.

Pour résoudre ce problème, Mogeau et Sankoff proposent de limiter de nombre se symboles intervenenant dans une fragmentation ou dans une consolidation.

Nous introduisons alors deux constantes C et F, et nous proposons le schéma suivant, lequel permet de ramener la complexité en temps de l'algorithme à $\mathcal{O}(m \cdot n)$.

$$\forall i \in \{0..m\}, S_{i,0} = 0$$
$$\forall j \in \{0..n\}, S_{0,j} = 0$$

$$\forall (i,j) \in \{1..m\} \times \{1..n\},$$
$$S_{i,j} = \max \begin{cases} 0 & \\ c(X[i], Y[j]) + S_{i-1,j-1} & (sub) \\ c(X[i], \lambda) + S_{i-1,j} & (del) \\ c(\lambda, Y[j]) + S_{i,j-1} & (ins) \\ \{c(X[i], Y[j-k+1]..Y[j]) + S_{i-1,j-k}\}, & \\ 2 \leq k \leq \min(j, F) & (frag) \\ \{c(X[i-l+1]..X[i], Y[j]) + S_{i-l,j-1}\}, & \\ 2 \leq l \leq \min(i, C) & (cons) \end{cases}$$

Le calcul explicite des constantes C et F, dans le cas de l'analyse musicale, est détaillé dans la section suivante.

4.3 δ-approximate matching : Mesures de similarité

Nous présentons dans cette section quatre mesues de similarité implémentées dans notre système pour la localisation d'occurences δ-simalaires à un pattern donné. La similarité δ, introduite à la section 4.2, ne peut comparer que des séquences de taille identique, et nessécite une fonction de distance élémentaire entre symboles à valeurs dans $\mathbb{N}$. On rappelle que deux séquences X et Y de taille l, sont dites δ-similaires si et seulement si :

$$\forall i \in \{1..l\}, d(X[i], Y[i]) \leq \delta$$

où d est une distance élémentaire entre deux symboles de Σ.

4.3.1 Hauteur absolue

La première distance implémentée dans notre système mesure l'écart entre deux notes, exprimé en demi-tons :

$$d_{abs}(x, y) = |x.pitch - y.picth|$$

L'utilisation de cette mesure permet par exemple de reprérer (en choisissant $\delta = 0$) toutes les occurences exactes d'un pattern donné. A noter toutefois que les identitées ainsi repérées ne sont que *partielles*, puisque leur identification n'est fondée que sur le seul paramètre de hauteur.

4.3.2 Intervalles chromatiques

L'inconvénient d'une identification basée sur la hauteur absolue est qu'elle laisse totalement échapper les occurences d'un pattern transposées dans un autre ton. Pour palier à se problème, nous avons implémentés une distance fondée sur les intervalles chromatiques (mesurés en demi-tons) entre une note et la note suivante. Cet intervalle existe *toujours* (sauf pour la dernière note d'une séquence) et a déjà été calculé par le système au cours de la phase d'enrichissement de données (cf. section 3.3.4)

$$d_{int}(x, y) = |x.rightint - y.rightint|$$

A noter par ailleurs que lors de la comparaison de deux séquences X et Y de taille l, on ne compare en réalité que des séquences de taille $l-1$, dans la mesure où on s'intéresse aux *intervalles* entre les notes. Ainsi, deux séquences X et Y de taille l, seront dites δ-similaires au sens de la distance d_{int} si et seulement si :

$$\forall i \in \{1..l-1\}, |X[i].rightint - Y[i].rightint| \leq \delta$$

4.3.3 Degrés diatoniques

Cette troisième distance permet d'intentifier deux séquences transposées diatoniquement, comme par exemple les mesures 1 et 5 de l'*Allegretto en Do mineur D.915 de Schubert* (cf. figure 3.6). Elle est fondée sur les descripteurs de degrés diatoniques d'une note au sein de la(des) tonalité(s) locale(s) ou au sein de l'(des) accord(s) local(aux). Ces descripteurs existent *toujours* et sont accessibles par les attribus `note.seqdeg` et `note.chorddeg` (cf. section 3.3.3).

Degrés dans les tonalités locales

Pour déterminer la distance diatonique entre deux notes, relativement à leurs tonalités locales, le système s'appuie sur une description explicite des degrés diatoniques inhérents aux gammes majeure, mineure, et chromatique :

Majeur	`['1','2','3','4','5','6','7']`
Mineur	`['1','2','3b','4','5','6b','7']`
Chromatique	`['1','2b','2','3b','3','4','5b', '5','6b','6','7b','7']`

Pour comparer diatoniquement deux notes, le système envisage l'alternative suivante :

1. Si chacune des notes, en regard de la représentation évoquée ci-dessus, appartient à sa tonalité locale, la distance d_{deg} retourne l'écart diatonique entre leur degrés respectifs. Par exemple, la distance diatonique entre un Sol en Do majeur (quinte) et un La en Fa majeur (tierce) sera égale à 2.

2. Si l'une des notes, en regard de la représentation évoquée ci-dessus, n'appartient pas à sa tonalité locale, la distance d_{deg} retourne l'écart diatonique entre leur degrés respectifs dans la gamme chromatique. Par exemple, la distance diatonique entre un Sol en Do majeur (quinte) et un Mi bémol en Fa majeur (septième mineure) sera égale à 3.

Notons par ailleurs que dans notre représentation explicite des gammes les listes sont circulaires. Ainsi, la distance entre un Do et un Si en Do majeur sera égale à 1. En outre, dans tous les cas, la distance diatonique sera inférieure ou égale à 6.

Grâce à cette distance, les première et cinquième mesures de l'*Allegretto en Do mineur D.915* (cf. figure 3.6) sont donc δ-similaires avec $\delta = 0$.

Degrés dans les accords locaux

De même que pour les tonalités locales, le système dispose d'une représentation explicite des degrés diatoniques possibles au sein de chaque accord :

Majeur	`['1','2','3','4','5','6','7']`
Mineur	`['1','2','3b','4','5','6b','7b']`
Dominant	`['1','2','3','4','5','6','7b']`
Demi-diminué	`['1','2b','3b','4','5b','6b','7b']`
Chromatique	`['1','2b','2','3b','3','4','5b', '5','6b','6','7b','7']`

Le système procède exactement de la même manière que dans le cas précédent pour calculer la distance diatonique relativement aux accrods locaux. Ainsi la distance entre un Fa dans un accord $G\emptyset$ et un La dans un accord $C7$ sera égale à 1. La distance entre un Sol dans un accord Am et un Fa dans un accord $D7$ sera égale à 5.

N.B. La représentation adoptée des dégrés diatoniques susceptibles d'être rencontrés dans les tonalités ou accords locaux est bien évidemment discutable. Le système ne connait en effet qu'une seule gamme mineure (la gamme mineure harmonique) et considèrera certaines notes des gammes mineures mélodique ou naturelle comme étrangères à la tonalité. De même il est fréquent, en jazz moderne, de jouer une seconde majeure sur les accords demi-diminués. Cela permet de faire ressortir la couleur de la gamme mineure mélodique, très employée en jazz, et devenue caractéristique de certains musiciens (Bill Evans, par exemple). La encore, le système considèrera cette note comme étrangère. Nous sommes conscients de ces limitations mais les tenons pour secondaires quant aux buts que nous sommes fixés. Nous ne saurions prétendre fournir une architecture capable de représenter de manière exhaustive toutes les connaissances musicales nécessaires à l'analyse du jazz tonal. Dans le cadre d'un stage de DEA, nous

entendons seulement aborder quelques facettes de l'analyse motivique, et nous pensons que notre système, tel qu'il a été implémenté, nous le permet.

4.3.4 Ratios de durées

Cette dernière distance s'avère très utile pour appareiller deux séquences équivalemente sur le plan rythmique, pouvant éventuellement être notées différemment (par exemple, un motif en croches et l'autre en double-croches). Nous définissons simplement la distance d_{ratio} par :

$$d_{ratio}(x, y) = k \times |x.ratio - y.ratio|$$

où k est une constante à déterminer. En pratique le choix de k importe peu, car nous utilisons en général cette distance pour repérer des occurences 0-similaires (cf. Chapitre 5).

4.4 Modèle d'édition : Fonctions de contribution

Nous présentons enfin dans cette section un ensemble de fonctions de contribution utilisées par le modèle d'édition de notre système. La première se base sur une représentation diatonique des hauteurs et sur la durée des notes pour en effectuer la comparaison. Les seconde et troisième font appel aux descripteurs de contour mélodique pour essayer d'appareiller des séquences mélodiques de contours semblables.

4.4.1 Fonction de Mongeau & Sankoff

La première fonction de contribution implémentée dans notre système est inspirée de celle proposée par Mongeau et Sankoff[25]. Cette fonction renvoie une mesure de la « ressemblance » entre deux notes, fondée sur l'écart entre leur degrés diatoniques respectifs et sur l'écart entre leurs durées. Elle se présente sous la forme :

$$c_{MS}(x, y) = k_2 \cdot (x.duration + y.duration) - k_1 \cdot d_{dur}(x, y) - d_{deg}(x, y)$$

où k_1 et k_2 sont deux constantes à définir et où d_{dur} et d_{deg} sont des distances mesurant respectivement l'écart en durée et l'écart en degrés diatoniques entre les deux notes x et y. Le paramètre k_1 permet de régler la part relative de chacune de ces deux distances. Le paramètre k_2 est utilisé pour passer d'une fonction de dissimilitude à une fonction de similitude.

Poids asscoié à la différence de durées

Dans le modèle de Mongeau et Sankoff, la durée des notes est mesurée en nombre de double-croches et la distance d_{dur} est simplement la différence de

durée entre les notes, exprimée également en nombre de double-croches. Cette représentation présente deux inconvénients majeurs :

1. Elle ne permet pas de prendre en compte des rythmes ternaires.
2. Elle accorde une similarité trop faible à deux séquences identiques notées avec des valeurs métriques différentes.

Nous lui substituons donc une distance renvoyant la différence de ratios de durée (cf. figure 3.14) entre les notes comparées, ces durées étant exprimées en nombre de ticks MIDI par noire, soit 120 dans la représentation adoptée pour notre système. Cette distance est par ailleurs pondérée par la somme des durées des deux notes, afin d'accorder plus de poids à une différence de ratio concernant des notes longues (un changement de durée entre deux notes courtes ayant un poids perceptif plus faible).

$$d_{dur}(x, y) = |x.ratio - y.ratio| \times (x.duration + y.duration)$$

Poids associé à la différence de hauteurs

De même que notre distance d_{deg} décrite dans la section 4.3.3, Mongeau et Sankoff proposent une distance mesurant l'écart entre les degrés diatoniques de chacune des notes dans la tonalité *globale*. Nous reprenons ce modèle en comparant toutefois les notes dans leur tonalité *locale*, afin d'autoriser d'éventuelles transpositions diatoniques d'un motif au sein d'une même séquence. Deux cas sont encore ici à distinguer :

1. Les deux notes à comparer appartiennent toutes deux à leurs tonalités locales respectives. Alors le poids associé à la différence de hauteur est $d_{deg} = deg(n)$, où n est l'écart entre degrés *diatoniques* et deg une fonction tenant compte des hiérarchies tonales entre les degrés.
2. Au moins une des deux notes n'appartient pas à sa tonalité locale. Dans ce cas le poids associé à la différence de hauteur est $d_{deg} = ton(n)$, où n est l'écart entre degrés *chromatiques* et ton une fonction tenant compte des hiérarchies tonales entre les degrés.

n	0	1	2	3	4	5	6
deg(n)	0.0	0.9	0.2	0.5	0.1	0.35	0.8

n	0	1	2	3	4	5	6	7	8	9	10	11
ton(n)	0.6	2.6	2.3	1.0	1.0	1.6	1.8	0.8	1.3	1.3	2.2	2.5

FIG. 4.7 – Poids associés aux écarts en degrés dans le modèle de Mongeau et Sankoff

A noter qu'ici encore, d_{deg} est multiplié par la somme des durées des notes comparées pour les mêmes raisons que précédemment.

Cas des silences

Un cas particulier mérite d'être traité. Quelle valeur attribuer à d_{deg} lorsqu'une des notes appareillées est un silence ? Si les deux notes sont des silences, on a éviemment $d_{deg} = 0$. Dans le cas contraire, Mongeau et Sankoff suggèrent la valeur constante $d_{deg} = 0.1$. Cette valeur peut paraître satisfaisante dans le mesure où elle est supérieure à $deg\,(0)$. Au cours de nos recherches il est toutefois apparu que cette valeur était trop faible, l'algorithme de comparaison préférant appreiller une note étrangère à la tonalité avec un silence ($d_{deg} = 0.1$) plutôt qu'avec une note étrangère de même degré diatonique ($d_{deg} = 0.6$). Nous avons donc réhaussé cette valeur à 1.0. On a alors :

$$\begin{gathered} c_{MS}\,(x, rest) = k_2 \cdot (x.duration + rest.duration) \\ -k_1 \cdot d_{dur}\,(x, rest) - d_{deg}\,(x, rest) \end{gathered}$$

Cas des insertions et délétions

Nous n'avons traîté jusqu'à présent que le cas des substitutions. En ce qui concerne les insertions ou les délétions, les poids respectifs de durées et d'intervalles sont, de manière très intuitive :

$$\begin{gathered} d_{dur}\,(x, \lambda) = x.ratio \times x.duration \\ d_{deg}\,(x, \lambda) = 0 \end{gathered}$$

On a alors :

$$c_{MS}\,(x, \lambda) = k_2 \cdot x.duration - k_1 \cdot d_{dur}\,(x, \lambda) - d_{deg}\,(x, \lambda)$$

Cas des fragmentations et consolidations

Lorsque plusieurs notes sont appareillées à une note, le calcul de d_{dur} et d_{deg} s'obtient par sommation des distances élémentaires :

$$\begin{aligned} d_{dur}\,(x, y_1..y_n) &= \sum_{i=1}^{n} d_{dur}\,(x, y_i) \\ d_{deg}\,(x, y_1..y_n) &= \sum_{i=1}^{n} d_{deg}\,(x, y_i) \end{aligned}$$

On a alors :

$$\begin{gathered} c_{MS}\,(x, y_1..y_n) = k_2 \cdot \left(x.duration + \sum_{i=1}^{n} y_i.duration \right) \\ -k_1 \cdot d_{dur}\,(x, y_1..y_n) - d_{deg}\,(x, y_1..y_n) \end{gathered}$$

Il nous faut également préciser de quelle manière sont déterminées les constantes F et C évoquées à la section 4.2. Avant de comparer deux séquences X et Y, l'algorithme effectue un parcours préalable de ces séquences et en extrait dans

chacune les notes les plus longues et les plus brèves, dans le but d'évaluer les plus grandes consolidation et fragmentation possible. On a alors :

$$F = \frac{\max\{X[i].duration, 1 \leq i \leq m\}}{\min\{Y[j].duration, 1 \leq j \leq n\}}$$

et :

$$C = \frac{\min\{X[i].duration, 1 \leq i \leq m\}}{\max\{Y[j].duration, 1 \leq j \leq n\}}$$

Paramétrage du modèle

Si les valeurs prises par les fonctions $deg(n)$ et $ton(n)$ peuvent être déterminées de manière « intuitive », à partir de considérations sur la hiérarchie des hauteurs en musique tonale, en revanche les constantes k_1 et k_2 posent davantage de problèmes. Dans leur article Mongeau et Sankoff suggèrent d'utiliser $k_1 = 0.348$ et $k_2 = 0.05$, et Rolland, dans sa thèse, conserve également ces valeurs. Elles sont déterminées de manière à réaliser des alignements « sensés », conformes à l'intuition, entre les variations du thème *« Ah ! vous dirais-je, maman » K.300* de Mozart.

Nous avons dans un premier temps expérimenté cette fonction de similitude en utilisant les valeurs de paramètres suggérées par Mongeau et Sankoff, bien que nous ayons adopté une représentation différente des durées. Les résultats étaient satisfaisants pour les variations de Mozart. En revanche, la comparaison de séquences présentant un nombre significatif de notes extérieures à la tonalité posait plus de difficultés. Soient par exemple à comparer les quatre motifs de la figure 4.8.

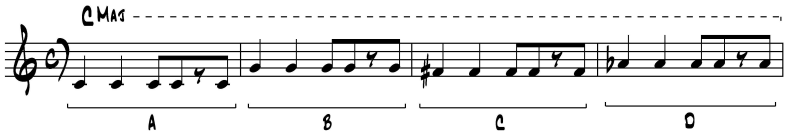


FIG. 4.8 – Cas de test pour le paramétrage du modèle

Avec les paramètres de Mongeau et Sankoff, les motifs A et B sont systématiquement alignés et la comparaison ne met en oeuvre que des substitutions. Toutefois, dès qu'une comparaison implique les motifs C ou D, l'algorithme trouve une transformation optimale composée de six délétions et six insertions. Le poids accordé aux valeurs rythmiques était donc trop faible. Nous avons essayé d'« optimiser » empiriquement les paramètres k_1 et k_2, en rejetant systématiquement tout couple de valeurs alignant deux motifs de la figure 4.8 avec d'autres types d'édition que des substitutions. Nous sommes parvenus aux valeurs $k_1 = 3.0$ et $k_2 = 2.4$, qui semble donner des résultats satisfaisants. Une valeur élevée de k_2 est nécessaire pour ne pas tomber trop rapidement dans des mesures négatives de la similarité.

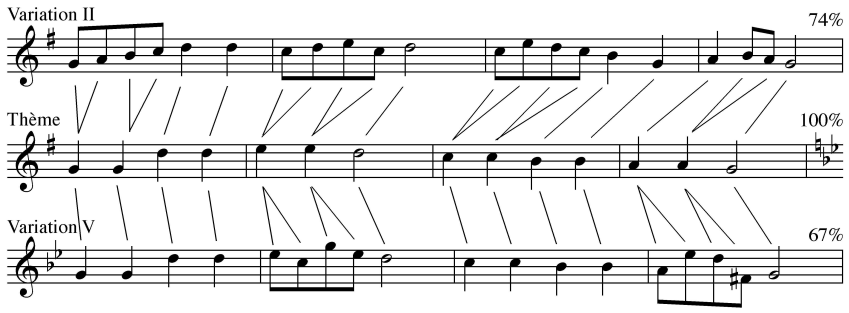


FIG. 4.9 – Similarités entre les variations de l'*« Ah ! vous dirais-je, maman » K.300* de Mozart

La figure 4.9 montre deux exemples d'alignement. Le thème de l'*« Ah ! vous dirais-je, maman » K.300* de Mozart est au centre. Les pourcentages traduisent le taux de similarité entre les séquences et sont calculés en référence à l'*auto-similarité* du thème (cf. chapitre suivant[3]).

A noter que la similarité entre la cinquième variation et le thème est plus faible qu'entre la seconde variation et le thème. Ces valeurs correspondent bien au jugement perceptif de la similarité entre ces séquences.

Nous sommes conscients que le paramétrage du modèle reste une question ouverte. Nous ne pouvons pas prétendre obtenir des paramètres robustes en se basant sur les seuls motifs de la figure 4.8 comme cas de test. En guise de travaux futurs, la recherche d'un calibrage efficace à l'aide d'une base d'exemples et de techniques d'apprentissage pourrait constituer une piste de travail.

Limitations

Si la fonction de contribution de Mongeau et Sankoff génère des alignements sensés entre séquences issues du répertoire classique ou de mélodies populaires (son efficacité est en général optimale entre variations proches d'une même motif), elle s'avère par contre limitée pour l'analyse du jazz, et trouve souvent des alignement surprenants.

[3] En particulier la notion de *ratio-identité*.

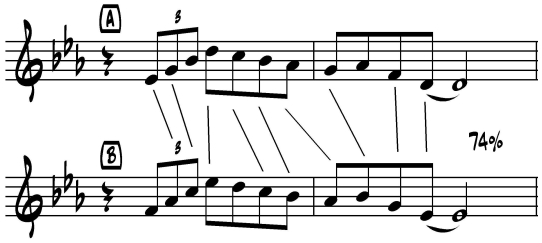


FIG. 4.10 – Limitations de la fonction de Mongeau et Sankoff

La figure 4.10 montre l'alignement de deux motifs déduits l'un de l'autre par transposition diatonique d'un degré, procédé fréquent dans le jazz. Très clairement l'alignement proposé ne correspond pas à la perception de ces deux motifs. L'intuition nous suggère davantage de les identifier sur leur similarités de rythme et de contour mélodique plutôt que de chercher à appareiller les degrés diatoniques les plus proches.

En outre, il est possible de trouver des séquences relativement éloignées sur le plan perceptif, mais présentant, au sens de la fonction de Mongeau et Sankoff, une grande similarité avec les séquences de la figure 4.10. Pour illustrer ce propos nous avons recherché le meilleur alignement entre le premier motif de la figure 4.10 et la séquence *The Bird, section 2* tirée de notre corpus (cf. annexe A). La figure 4.11 met clairement en évidence les limitations de la fonction de Mongeau et Sankoff.

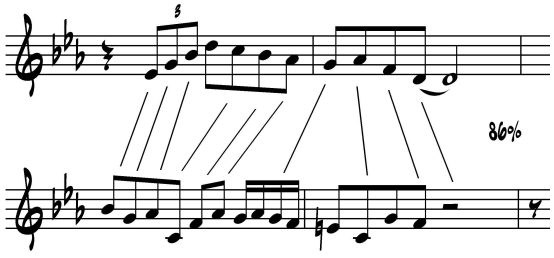


FIG. 4.11 – Un alignement surprenant

Perceptivement, les motifs de la figure 4.11 sont plus éloignés que ceux de la figure 4.10. Les mesures de similarités contredisent pourtant cette observation. Ce pourquoi nous proposons d'introduire deux nouvelles fonctions de contributions, dans le but d'aligner les séquences en fonction de leur contour mélodique.

4.4.2 Fonctions de contour : Intervalles algébriques

Cette fonction de contribution utilise les attributs `note.rightint` et `note.leftint` pour tenter d'appareiller des notes liées aux notes précédente et suivante par des intervalles similaires. Ainsi que suggéré par Rolland[38], nous utilisons une fonction fortement discrète standard, à valeurs finies :

$$c_{Contour} : \Sigma^* \times \Sigma^* \rightarrow \{Min, Bas, Moyen, Haut, Max\}$$

avec $Min = -4$, $Bas = -2$, $Moyen = 0$, $Haut = +2$, et $Max = +4$.

Le calcul de cette fonction fait appel à une arborescence de cas relativement complexe, en raison du fait que les attributs `note.rightint` et `note.leftint` ne sont pas toujours renseignés. C'est le cas des silences, et des notes situées aux extrémités des phrases.

Cas d'une substitution

Soient à comparer deux notes x et y. Nous éliminons dans un premier temps les cas suivants :

- Si x et y sont des silences, alors $c_{Contour}(x, y) = Max$.
- Si x ou (exclusif) y est un silence, alors $c_{Contour}(x, y) = Min$.

Nous supposons maintenant que x et y ne sont pas silences et que leurs attributs `note.rightint` et `note.leftint` sont renseignés (x et y ne sont pas des notes extrêmes de phrases). Nous comparons alors deux à deux : $x.rightint$ avec $y.rightint$, et $x.leftint$ avec $y.leftint$, en ne sous intéressant qu'aux signes de ces valeurs (comparaison fondée sur les intervalles algébriques). Pour cela nous utilisons la terminologie suivante (les flèches $\uparrow$, $\rightarrow$, et $\downarrow$ désignent respectivement la qualité des intervalles à comparer : ascendant, plat, descendant) :

	$\uparrow$	$\rightarrow$	$\downarrow$
$\uparrow$	*identiques*	*semblables*	*contraires*
$\rightarrow$	*semblables*	*identiques*	*semblables*
$\downarrow$	*semblables*	*semblables*	*identiques*

Nous appliquons enfin la dychotomie finale, en se rappelant que nous comparons ici deux couples d'intervalles :

$$\begin{array}{lcl} 2\,identiques & \rightarrow & Max \\ 2\,contraires & \rightarrow & Min \\ \left.\begin{array}{l} 1\,identique \\ 1\,semblable \end{array}\right\} & \rightarrow & Haut \\ \left.\begin{array}{l} 1\,identique \\ 1\,contraire \end{array}\right\} & \rightarrow & Bas \\ \left.\begin{array}{l} 1\,semblable \\ 1\,contraire \end{array}\right\} & \rightarrow & Min \\ 2\,semblables & \rightarrow & Moyen \end{array}$$

Dans le cas où les attributs `note.rightint` et `note.leftint` ne sont pas tous renseignés, on compare, si c'est possible, les deux intervalles à gauche, ou les deux intervalles à droite (si c'est impossible, on a alors $c_{Contour}(x, y) = Min$) :

$$\begin{array}{lcl} identique & \rightarrow & Max \\ semblable & \rightarrow & Moyen \\ contraire & \rightarrow & Min \end{array}$$

Autres cas

Les insertions et les délétions sont sanctionnées de manière identique :

$$c_{Contour}(x, \lambda) = c(\lambda, y) = Bas$$

Quant aux fragmentations et consolidations, nous procédons de façon analogue à la méthode proposée dans la section précédente :

$$c_{Contour}(x, y_1..y_n) = \sum_{i=1}^{n} c(x, y_i)$$

Des preuves !

La figure 4.12 montre l'alignement des deux motifs qui avaient mis en échec la fonction de Mongeau et Sankoff. Cette fois l'alignement semble plus conforme à l'intuition, et la similarité est maximale (100% : les séquences ont exactement le même contour mélodique).

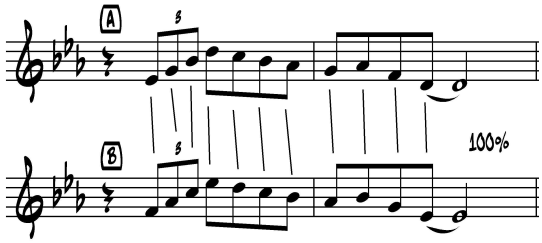


FIG. 4.12 – Alignement selon le contour mélodique

Par ailleurs la similarité entre les motifs de la figure 4.11 n'est plus que de 73%. Cette fois encore, nous avons recherché la meilleure occurence du premier motif de la figure 4.10 dans la séquence *The Bird, section 2* . L'occurence trouvée (cf. figure 4.13), cette fois, est bien plus convaincante (95%).

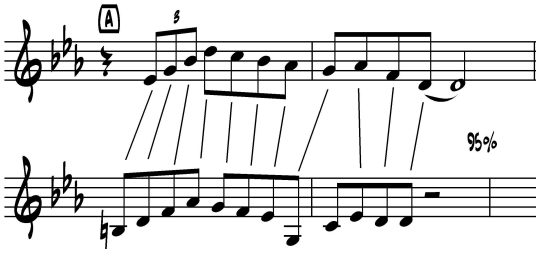


FIG. 4.13 – Une occurence plus convaincante

La comparaison de la fonction de contribution de Mongeau et Sankoff avec notre fonction basée sur le contour mélodique est résumée dans le tableau ci-dessous. Les lettres A et B désignent les séquences de la figure 4.12. Les séquences $Bird_{MS}$ et $Bird_{Contour}$ désignent respectivement les meilleures occurences de la séquence A trouvées dans la séquence *The Bird, section 2* à l'aide de la fonction de contribution de Mongeau et Sankoff et notre fonction de contribution. Les éléments du tableau sont les valeurs de similarité avec la séquence A. La dernière colonne correspond à la similarité calculée avec une somme équipondérée des deux fonctions.

	c_{MS}	$c_{Contour}$	$c_{MS} + c_{Contour}$
B	67%	100%	89%
$Bird_{MS}$	86%	73%	35%
$Bird_{Contour}$	54%	95%	65%

4.4.3 Fonctions de contour : Extrema de hauteur

Nous avons également implémenté une fonction de contribution qui cherche à appareiller les extrema de hauteurs des séquences à comparer. Cette fonction c_{Extr} utilise notamment les attributs `note.isextremum`, `note.dist2ext`, et `note.slope` (cf. figure 3.14), lequel, comme les intervalles algébriques, peut prendre les valueurs *ascendant*, *descendant*, ou *plat*. La encore, nous utilisons une fonction fortement discrète standard, dont le calcul est aussi basé sur une arborescence complexe de possibilités.

Nous ne détaillons ici que le cas de la substitution d'une note x à une note y. Les autres possibilités peuvent se déduire très facilement, à l'aide d'une démarche similaire à celle menée dans la section précédente.

- Si x et y sont des extrema de hauteurs, et s'ils sont de même type, alors $c_{Extr}(x, y) = Max$. S'ils sont de types différents, alors $c_{Extr}(x, y) = Min$.
- Si ni x ni y n'est un extremum de hauteur alors :

$$c_{Extr}(x, y) = \begin{cases} Bas & si\ x.slope \neq y.slope \\ Haut - |x.dist2ext - y.dist2ext| & si\ x.slope = y.slope \end{cases}$$

- Si enfin y seulement est un extremum de hauteur, alors :
  - Si y est un maximum et si $x.slope = \uparrow$, alors :

$$c_{Extr}(x, y) = Haut - x.dist2ext$$

  - Si y est un maximum et si $x.slope = \downarrow$, alors :

$$c_{Extr}(x, y) = Bas$$

  - Si y est un minimum, le raisonnement est symétrique.

Cette fonction de contribution cherche à appareiller non seulement les extrema de hauteur, mais également les notes sitées *avant* chaque extremum, en fonction de leur position par rapport à celui-ci (les notes situées après seront elles appareillées en fonction de l'extremum qu'elles précèdent, s'il existe). Utiliser l'attribut `note.dist2ext` permet ainsi d'accorder une similarité plus élevée à deux notes situées à des distances comparables de leurs extrema respectifs.

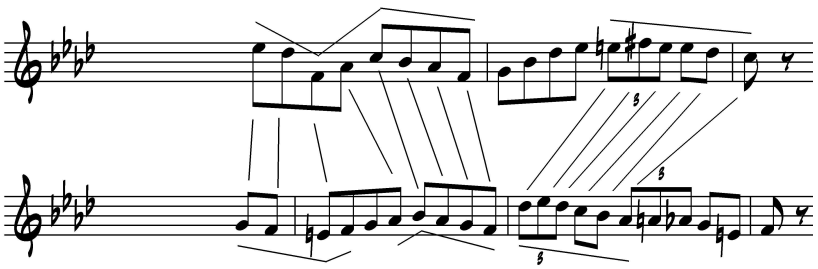


FIG. 4.14 – Alignement selon les extrema de hauteur

La figure montre un alignement entre deux phrases du thème de *Donna Lee*. Les extrema sont appreillés entre eux et les pentes mélodiques autour de ces derniers sont respectées. En général, les alignements générés par les fonctions $c_{Contour}$ et c_{Extr} sont sensiblement équivalents. Nous n'avons par trouvé d'exemple illustrant une différence notable entre ces deux fonctions.

Dans le chapitre suivant, nous nous restreignons donc à l'utilisation de $c_{Contour}$ lorsque nous voulons mettre le contour mélodique à contribution dans la localisation de patterns.

Chapitre 5

Machineries

« Les ailes de la Rolls effleuraient les pylônes
Quand m'étant malgré moi égaré
Nous arrivâmes ma Rolls et moi dans une zone
Dangereuse, un endroit isolé »
- Serge Gainsbourg

Nous présentons dans ce chapitre deux algorithmes de localisation de patterns implémentés dans notre système. Le premier algorithme recherche les occurences δ-similaires à un pattern donné, et utilise les distances élémentaires entre symboles décrites à la section 4.3. Le second recherche des occurences approximatives (au sens d'un modèle d'édition) et utilise les fonctions de contributions présentées à la section 4.4. Nous discutons en outre les performances de ces algorithmes et les possibilités qu'ils offrent en regard de l'analyse orientée patterns, ainsi que la pertinence des outils présentés dans ce rapport.

5.1 δ-Approximate Matching

5.1.1 Algorithme

Etant donnés un alphabet Σ, X une séquence sur Σ de taille n, et P un pattern sur Σ de taille $m \leq n$, nous recherchons tous les indices $k \in \{m..n\}$ tels que :

$$\forall i \in \{1..m\}, d(P[i], X[k+i-1]) \leq \delta$$

où d est une distance élémentaire à valeurs dans $\mathbb{N}$. Autrement dit, k est l'indice initial d'un facteur de X de taille m, δ-similaire à P.

Nous utilisons un algorithme très élégant proposé par Cambouropoulos et al.[9], basé sur une représentation binaire des séquences à comparer. Dans un premier temps, nous construisons une table (appelée « *Delta-Table* », ou DT), associant à chaque symbole de Σ présent dans X une « mesure » de sa δ-similarité avec l'ensemble des symboles de P. Dans le pire des cas, la table DT aura pour taille

$|\Sigma|$. Sa construction est donc en $\mathcal{O}(1)$, elle ne dépend pas da la taille de X. Soit Σ_X le sous-ensemble de Σ ne contenant que les symboles présents dans X. On a alors :

$$\forall x \in \Sigma_X, DT(x) = r$$

où $r = [r_1..r_m]$ est un mot binaire défini par :

$$\forall i \in \{1..m\}, r_i = \begin{cases} 1 & si\ d(x, P[i]) \leq \delta \\ 0 & sinon \end{cases}$$

Nous construisons ensuite récursivement le vecteur DS (*Delta-State*) de la façon suivante :

$$DS[0] = 0$$
$$\forall j \in \{1..n\}, DS[j] = ((DS[j-1] \ll 1) \text{ OR } 1) \text{ AND } DT(X[j])$$

où $\ll 1$ désigne un décalage de 1 bit vers la gauche, OR et AND les opérateurs logiques standards.

Nous avons alors la propriété suivante : Un facteur F de X de rang initial k et de taille m est une δ-occurence de P si et seulement si le $m^{\text{ième}}$ bit de $DS[k+m-1]$ est égal à 1, soit, lorsque le vecteur DS est exprimé en notation décimale, si et seulement si :

$$DS[k+m-1] \geq 2^{m-1}$$

Les occurences δ-approximatives d'un pattern peuvent ainsi être obtenues très rapidement avec un algorithme de complexité en temps en $\mathcal{O}(n)$.

5.1.2 Recherche multi-critères

Nous avons présenté à la section 4.3 quatre distances élémentaires pour la localisation d'occurences δ-approximatives. Ces quatres fonctions (hauteurs absolues, intervales chromatiques, degrés dans la tonalité locale ou dans l'accord local, ratios de durée) peuvent être combinées de façon à imposer des contraintes sur les occurences cherchées. Pour chaque distance élémentaire, nous construisons une table DT et un vecteur DS spécifique. Les occurences retenues sont celles communes aux critères choisis. L'algorithme 4 résume cette procédure.

Algorithm 4 Recherche multi-critères d'occurences δ-approximatives

```
pour_chaque distance_élémentaire_i, 1≤i≤n
{
  construire Delta-Table
  construire Delta-State
  repérer occurences
  occurences[i] ← occurences
}
retourner intersection(occurences[i], 1≤i≤n)
```

5.1.3 Recherche dans un corpus de séquences

Lorsque nous recherchons les occurences d'un pattern dans un corpus de séquences, l'algorithme 4 est utilisé pour chaque séquence du corpus. Les occurences sont alors repérées par le nom des séquences auxquelles elles appartiennent ainsi que leurs coordonnées au sein de la séquences.

Travailler sur un corpus de séquences et non sur une séquence unique, obtenue par concaténation des séquences du corpus (cf. Rolland[38]), ne modifie ni la complexité de l'algorithme ni celle de son implémentation. En outre, cela permet d'éviter un post-traitement sur les résultats pour éliminer les occurences « à cheval » sur deux séquences.

La figure E.3 montre l'interface utilisée par notre système pour la localisation de patterns δ-similaires. Pour chaque occurence l'alignement avec le pattern recherché peut être visualisé, et un fichier MIDI de l'occurence peut être exporté. Le choix est laissé à l'utilisateur d'éliminer les occurences « à cheval » sur deux phrases. Dans ce dernier cas, le système procède simplement à un post-traitement des résultats, en utilisant l'attribut `MMF.phrases` (cf. figure 3.13).

5.2 Modèle d'édition

5.2.1 Fonction d'équipolence

La localistaion de patterns au sens de la δ-similarité présente l'avantage de fournir un critère explicite de détection des occurences. La distance binaire qu'elle implique est déjà une fonction d'équipollence : ou bien les deux séquences à comparer sont similaires, ou bien elles ne le sont pas, et il n'y a pas d'intermédiaire. Dans le cas d'une mesure de la similarité avec un modèle d'édition, nous avons un continuum de valeurs possibles, et il nous faut un critère explicite pour décider si deux facteurs sont équipollents.

Définition 5.2.1 (ratio-identité) Soient Σ un alphabet, X et Y deux séquences sur Σ et s une fonction de similarité sur $\Sigma^* \times \Sigma^*$. On définit le *ratio-identité* de Y relativement à X par :

$$r_X(Y) = \frac{s(X,Y)}{s(X,X)}$$

On a évidemment :

$$\forall Y \in \Sigma^*, 0 \leq r_X(Y) \leq 1$$

$s(X,X)$ est appelé *auto-similarité* de la séquence X. La notion de ratio-identité, introduite par Rolland[38], permet de normaliser la similarité relativement au pattern cherché. En effet, l'auto-similarité d'une séquence étant d'autant plus élevée que la taille de celle-ci est grande, une même valeur de similarité n'a pas la même signification selon la taille des séquences comparées. Substituer le ratio-identité à la similarité permet de palier à ce problème.

Nous pouvons maintenant définir une fonction d'équipollence pour l'identification de patterns à l'aide d'un modèle d'édition. Soient Σ un alphabet, P un pattern sur Σ dont nous cherchons les occurences approximatives dans une séquence X. Alors pour tout facteur F de X, on dira que F est une occurence de P dans X si et seulement si :

$$r_P(F) \geq \alpha$$

où α est un seuil à définir. Dans notre système, celui-ci est laissé au choix de l'utilisateur. La fonction d'équipollence est évidemment :

$$Eq\,(F, P) = \text{VRAI} \Leftrightarrow r_P(F) \geq \alpha$$

5.2.2 Recherche d'occurences

Nous avons présenté, à la section 4.2.2, un algorithme efficace pour calculer la similarité locale, au sens d'un modèle d'édition entre deux séquences. Ce calcul nécessitait la construction d'une matrice de similarité entre les séquences à comparer. Lorsque nous cherchons toutes les occurences d'un pattern donné P dans une séquence X (au sens de la fonction d'équipollence définie ci-dessus), nous construisons également une matrice de similarité S_{PX} dont nous savons que le plus grand élément est la valeur de la similarité entre P et *la* meilleure occurence de P dans X. La question est alors : Comment trouver les autres occurences ?

Une approche naïve serait la suivante. Nous construisons une matrice T_{PX} de même taille que S_{PX} dont chaque élément est une étiquette sur l'élément à la même place dans S_{PX}. Cette étiquette peut prendre deux valeurs : *utilisé (0)* ou *non utilisé (1)*. Initialement, tous les éléments de T_{PX} sont égaux à 1. Nous commençons donc par chercher le couple d'entiers $\{i, j\}$ qui maximise $S_{PX}\,[i, j]$. Si cette valeur est suffisamment élevée pour être une occurence de P, nous construisons l'alignement correspondant (cf. section 4.2.2), nous marquons le couple $\{i, j\}$ comme utilisé, ($T_{PX}\,[i, j] = 0$) et nous continuons la recherche sur la matrice S'_{PX}, avec :

$$\forall\,(i, j) \in \{1..m\} \times \{1..n\}\,, S'_{PX}\,[i, j] = S_{PX}\,[i, j] \times T_{PX}\,[i, j]$$

La recherche s'arrête lorsque :

$$\forall\,(i, j) \in \{1..m\} \times \{1..n\}\,, \frac{S'_{PX}\,[i, j]}{s\,(X, X)} < \alpha$$

Cette méthode présente l'inconvénient d'extraire *toutes* les occurences, au sens de la fonction d'équipollence, d'un pattern donné. Or toutes ces occurences ne sont pas nécessairement bonnes à prendre. Il est par exemple évident que si $F = X\,[i..i + l]$ est une occurence de P dans X, alors il y a toutes les chances

pour que $F' = X\,[i..i+l-1]$ le soit aussi. Mais F' n'apporte pas d'information supplémentaire une fois que F a été découvert.

Pour palier à ce problème, nous étiquetons dans T_{PX}, à chaque localisation d'occurence F, non seulement le couple $\{i, j\}$ qui maximise $S'_{PX}\,[i, j]$, mais également tous les couples de coordonnées qui correspondent à des occurences F' séquentiellement « trop proches » de F, i.e. vérifiant :

$$|F \cap F'| \geq \beta \times |P|$$

où le facteur β est laissé à la discrétion de l'utilisateur. En ce qui nous concerne, nous obtenons de bons résultats avec $\beta = 0.5$.

5.2.3 Recherche multi-critères dans un corpus

Lorsque nous recherchons un pattern dans un corpus de séquences, nous construisons autant de matrices de similarité qu'il y a de séquences dans le corpus considéré. Chaque occurence est identifiée par le nom de la séquence dans laquelle elle se trouve ainsi que ses coordonnées au sein de la séquence. La similarité est calculée à partir des fonction de contribution décrites à la section 4.4. L'utilisateur dispose d'un système de poids pour accorder davantage d'importance à l'une ou l'autre de ces fonctions. Comme dans le cas de la recherche d'occurences δ-similaires, l'utilisateur est libre d'éliminer les occurences « à cheval » entre deux phrases. La figure E.4 montre l'interface de notre système pour la localisation d'occurences approximative à l'aide de notre modèle d'édition.

5.3 En pratique

5.3.1 Pertinence de la δ-similarité

Notre algorithme de δ-matching, s'il offre l'avantage d'être extrêmement peu coûteux en temps de calcul ($\mathcal{O}(n)$), s'avère en revanche d'un intérêt limité pour la localisation d'occurences approximatives. Deux raisons à cela :

1. La localisation de patterns δ-similaires, basée sur une définition binaire de la similarité (ou bien deux séquences sont similaires par rapport à un critère donné, ou bien elles ne le sont pas), ne permet pas de quantifier la « qualité » des occurences extraites. La figure montre deux occurences du pattern *ow*3*ac* (cf. Annexe C) avec $\delta = 1$ (les distances utilisées sont les intervalles chromatiques et les ratios de durée, cf. section 4.3).

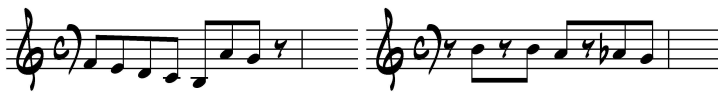


FIG. 5.1 – Deux δ-occurences du pattern *ow*3*ac*

Très clairement, la seconde occurence est bien moins proche du pattern initial que la première, mais la définition de la δ-similarité ne permet pas d'établir cette distinction.

2. La localisation de patterns δ-similaires ne permet que de repeérer des occurences du même nombre de notes (silences inclus) que le pattern initial. Elle élude donc totalement toute notion d'anticipation, retard, répétiton de note, fragmentation... bref, le formalisme DTW[1], qui justifie l'interêt des distances d'édition, y est absent.

Toutefois nous ne devons pas négliger l'intérêt du δ-matching. Tel qu'il a été implémenté dans notre système, cet algorithme permet de repérer très rapidement les occurences « quasi » identiques à un pattern donné, c'est-à-dire équivalentes à une transposition près (comparaison des intervalles chromatiques) ou à une contraction/expansion rythmique près (comparaison des ratios de durées). Comme nous allons le montrer dans la section suivante, cette fonctionnalité s'avère très utile pour juger de la qualité de notre modèle d'édition.

5.3.2 Sélectivité du modèle d'édition

Nous nous proposons ici de localiser toutes les occurences du pattern $ow5Ca$ (cf. Annexe D) dans un corpus. Pour le confort de l'analyse (en termes de temps de calcul) nous effectuons cette recherche dans le sous-corpus $CDbM$ constitué de tous les solos joués dans les tonalités globales de Do et Ré bémol majeur.

Pour mener à bien la localisation il nous faut spécifier plusieurs paramètres au modèle :

- fonctions de contribution utilisées,
- poid relatif de chacune des fonctions de contributions,
- seuil de tolérance sur le ratio identité.

Localisation des occurences « quasi » exactes

Pour que notre modèle soit valide il faut évidemment qu'il soit capable de repérer les occurences exactes, ou « quasi » exactes, d'un pattern donné.

Définition 5.3.1 Etant donnés un pattern P et un facteur F d'une séquence X, on dira que F est une *occurence quasi-exacte* de P dans X si et seulement si F et P sont 0-similaires, au sens de la similarité δ fondée sur les intervalles chromatiques et les ratio de durées (cf. section 4.3). Autrement dit, F et P sont identiques perceptivement, et *sur le plan mélodique uniquement, en dehors de tout contexte harmonique.* Une fois posée cette définition, nous recherchons dans un premier temps les occurences quasi-exactes du pattern $ow5Ca$ dans le sous-corpus $CDbM$. L'algorithme de δ-matching en trouve trois, dont une dans la tonalité d'origine (Fa majeur) et deux dans une tonalité voisine (Do majeur).

[1] Dynamic Time Warping (cf. section 4.2.2).

Profil d'occurence avec la fonction de Mongeau et Sankoff

Nous cherchons d'abord à localiser, à l'aide de notre modèle d'édition et de la seule fonction de contribution de Mongeau et Sankoff, *toutes* (nous n'imposons pas de seuil) les occurences du pattern $ow5Ca$. Le profil d'occurences (triées par ordre décroissante de ratio-identité) est représenté figure 5.2.

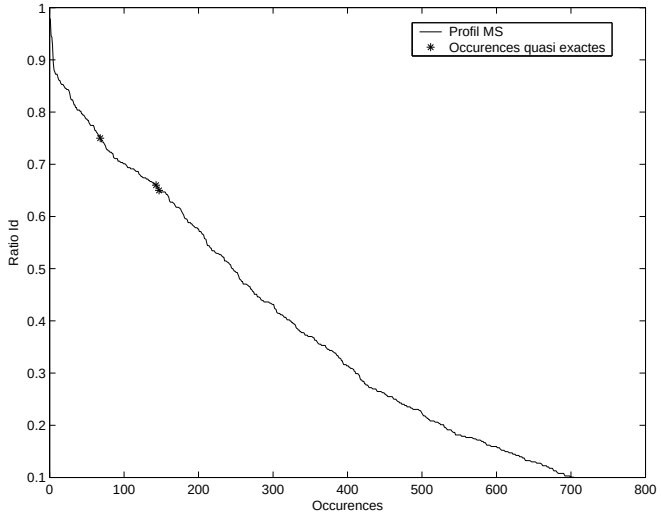


FIG. 5.2 – Profil d'occurences du pattern $ow5Ca$, utilisant la fonction de Mongeau et Sankoff

Si nous voulons conserver la validité du modèle, il nous faut au minimum appliquer sur le ratio-identité un seuil de 0.65 pour conserver les trois occurences quasi-exactes préalablement repérées. Avec une valeur aussi basse, le modèle trouve 147 occurences approximatives du pattern, occurences dont l'écoute est peu convaincante. A titre d'exemple, nous donnons figure 5.3 la meilleure occurence trouvée du pattern $ow5Ca$, avec un ratio-identité de 0.98.

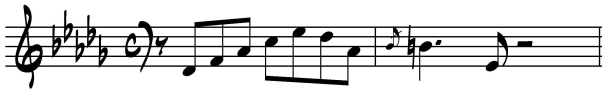


FIG. 5.3 – Une occurence surprenante du pattern $ow5Ca$

De façon très évidente, la seule fonction de contribution de Mongeau et Sankoff ne suffit pas à l'obtention d'un modèle sélectif. Les meilleures occurences ne présentent pas de similarité perceptible avec le pattern recherché, et les occurences

quasi-exactes sont largement sous-estimées.

Profil d'occurences avec notre fonction de contour

La figure 5.4 montre le profil d'occurences du pattern $ow5Ca$, obtenu avec notre fonction de contribution basée sur la contour mélodique.

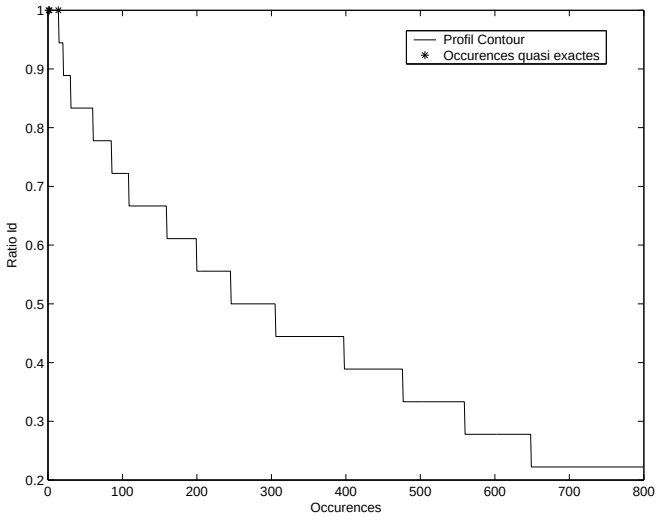


FIG. 5.4 – Profil d'occurences du pattern $ow5Ca$, utilisant notre fonction de contour

Cette fois, les occurences quasi exactes sont récompensées, avec un ratio-identité égal à 1. Toutefois, d'autres occurences perceptiblement plus éloignées du pattern initial présentent également le même ratio-identité. Le modèle est donc sélectif, mais trop peu discriminant.

Profil d'occurences avec une fonction mixte

Nous utilisons enfin une combinaison linéaire des deux fonctions de contribution précédentes, en attribuant un poid égal à chacune. Le profil d'occurences est cette fois très satisfaisant. Les occurences quasi-exactes occupent les valeurs élevées du profil, avec des ratios-identité respectifs de 0.90 et 0.88. En outre, le modèle trouve 17 occurences au dessus de 0.88, toutes perceptiblement très similaires au pattern cherché. L'écoute de toutes les occurences nous suggère une valeur possible de seuil égale à 0.8, au dela de laquelle la similarité avec le pattern $ow5Ca$ se dégrade très nettement.

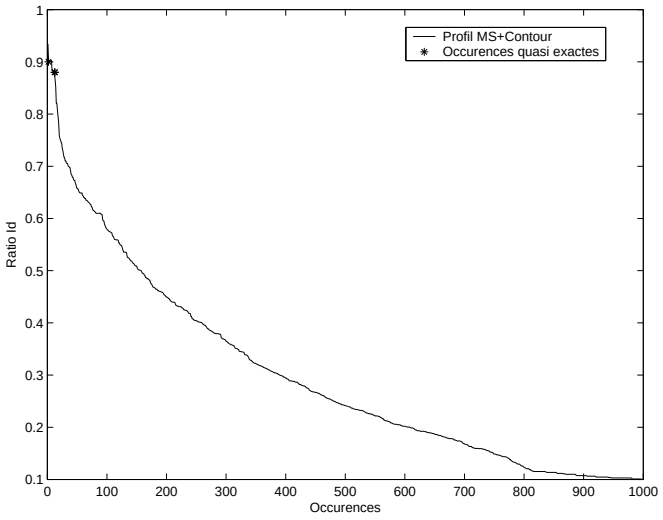


FIG. 5.5 – Profil d'occurences du pattern $ow5Ca$, utilisant une fonction de contribution mixte

5.3.3 Localisation d'un pattern

Nous testons maintenant notre modèle de localisation avec les paramètres déterminés au cours de l'expérimentation précédente, à savoir :

- une fonction de contribution mixte égale à la somme équipondérée de la fonction de Mongeau et Sankoff et de notre fonction de contribution basée sur le contour mélodique,
- une valeur de seuil sur le ratio-identité égale à 0.8.

Nous nous proposons cette fois de repérer le pattern $patA1$ extrait par le système **FlExPat** de Pierre-Yves Rolland[38]. L'expérience est menée sur le sous-corpus $CMblues$, composé de tous les morceaux de forme *blues* dans la tonalité de Do majeur. Le profil d'occurence (cf. figure 5.6) semble là encore très convenanble. Toutes les occurences au dessus du seuil (dont les occurences quasi-exactes) sont à l'écoute très satisfaisantes. Une seule occurence (avec un ratio-identité de 0.75) perceptiblement proche du pattern recherché échappe toutefois à la localisation.

5.3.4 Discussion

Nous sommes conscients que le paramétrage « empirique » du modèle, tel que nous l'avons exposé dans ce chapitre, demande à être mené avec une plus grande rigueur et sur une base d'exemples plus conséquente. Nous ne pouvons prétendre, à partir des différents profils d'occurences relatifs à un unique pattern, avoir obtenu des valeurs robustes pour les paramètres de notre modèle.

Toutefois, cette courte série d'expériences nous aura au moins fourni quelques

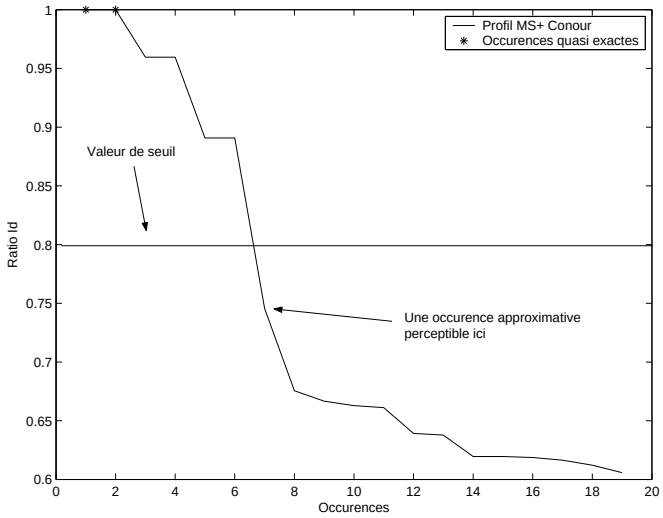


FIG. 5.6 – Profil d'occurences du pattern $patA1$

clés pour d'éventuels travaux futurs plus poussés. Le concept d'occurence quasi-exacte par exemple, justifie la pertinence du δ-matching (permettant d'écarter rapidement des configurations trop lâches du modèle) dans ce type de recherche. Par ailleurs, une localisation efficace de patterns mélodiques à l'aide d'un modèle d'édition nécessite une fonction de contribution cabable d'appareiller des motifs de contours mélodiques semblables. La fonction de Mongeau et Sankoff, dont la pertinence dans l'analyse de la musique tonale est indéniable, puisqu'elle a recours à une modélisation explicite de la hiérachie des hauteurs, ne peut à elle seule suffire à l'obtention d'un modèle suffisamment sélectif.

Chapitre 6

Trajectoires

« *Un jour, peut-être, le siècle sera deleuzien.* » - Michel Foucault

6.1 Conclusions et perspectives

Le bilan de ce stage est difficile à dresser. En regard des objectifs de recherche que nous nous étions fixés, certes inatteignables sur une durée aussi courte, il nous semble n'avoir parcouru que des premiers pas, et sans doute n'avons-nous réussi qu'à délimiter, et encore de manière assez floue, une possible direction de travail. Face à la profusion des approches, face à la multitude des choix qui s'offraient à nous, nous avouons nous être sentis quelque peu désemparés, et sans doute la longueur de ce mémoire témoigne de nos tâtonnements. Comme dit Bachelard, « *Quand il s'agit d'écrire des sottises, il serait vraiment trop facile de faire un gros livre.* »

Sans doute avons-nous commis l'erreur, dès le début de notre travail, de vouloir se doter d'un système relativement complexe pour la représentation des connaissances musicales dans le cadre d'une analyse orientée jazz, et sans doute l'estime personnelle que nous dévouons à cette musique nous a-t-elle imposé un niveau d'exigence que la durée du stage ne nous permettait pas. Toutefois, le développement d'un tel système n'a jamais cessé de nous paraître nécessaire, et nous continuons aujourd'hui de penser que nous n'avons pas perdu notre temps, quand bien même cette phase de conception aura occupé à elle seule plus de la moitié du stage. Cette architectue dont, connaissant les capactités et les carences, nous pensons à de potentielles améliorations, peut déjà servir de plateforme pour une vaste série d'expérimentations.

En ce qui concerne l'analyse motivique proprement dite, nous avons le sentiment d'avoir acquis une vision relativement claire des approches actuelles, savoir en vertu duquel nous ne pensons pas nous être mépris en faisant le choix, pour l'analyse que nous nous proposions, d'algorithmes basés sur des modèles d'édition. Certes nous n'avons qu'à peine eu le temps d'aborder la localisation de patterns, mais nous avons préféré ne pas brûler les étapes. L'implémentation

d'algorithmes d'extraction de motifs, la réalisation d'un système agnostique d'induction de régularités au sein de la surface musicale, n'auraient certes pas posé de difficultés notables en regard du travail déjà accompli, mais il nous paraissait raisonnable de n'aborder ce sujet qu'une fois les processus de localisation maîtrisés et efficacement exploités. Nous ne disposons toujours pas, à l'heure actuelle, d'un modèle robuste de localisation, paramétré à l'aide d'une phase d'apprentissage et validé par un savoir musicologique tel que celui fourni par Tom Owens sur Charlie Parker. Cette étape franchie, nous pourrons alors envisager l'induction automatique de patterns mélodiques et confronter ces résultats au travail des musicologues. Le recours à l'analyse automatique se justifiera par la possibilité de traîter rapidement un corpus conséquent de partitions, au sein duquel la production de connaissances de plus haut niveau et inaccessibles à l'analyse traditionnelle sera sans doute possible. Nous pensons en particulier à déterminer si les patterns découverts couvrent une partie significative du corpus étudié, et si tel est le cas, tenter d'inférer leurs mécanismes de production à l'aide d'outils propres à l'intelligence artificielle, comme les grammaires formelles par exemple. Les travaux de David Cope sur la détermination de signatures des compositeurs classiques laissent en effet à penser qu'il y a là également matière à caractériser, de manière objective, le style d'un musicien de jazz. Si cette hypothèse se vérifie pour Charlie Parker, nous pourons alors envisager l'étude d'autres musicens, et pourquoi pas, dans des styles musicaux autres que le jazz tonal.

6.2 Quelques réflexions sur l'analyse motivique

Outre les enjeux musicologiques qu'elle évoque, nous pensons que l'extraction automatique de patterns dans les séquences musicales ne se borne pas à des seules fins analytiques.

Au simple niveau pédagogique par exemple (la question épineuse de l'enseignement idéal du jazz est loin, tant mieux sans doute, d'être tranchée), elle permettrait de donner au jeunes musiciens quelques clés alternatives pour le travail de l'improvisation. Loin de nier que l'étude exhaustive et approfondie des solos de Parker constitue une étape essentielle à quiconque prétend devenir musicien de bebop, quelques points de repères pour aborder cet édifice décourageant ne seraient pas malvenus. Nous-même, au cours de notre travail, à force d'écouter de longues séries de variations du même motif, avons-nous eu parfois le sentiment d'accéder à une vision différente, intentionnelle, du jeu de Parker. Et plutôt que de s'attaquer au relevé complet d'un solo, nous avons préféré ne travailler que certains segments. Tous les professeurs de jazz seront d'accord sur ce point. *Si vous voulez jouer comme untel, travaillez ses motifs les plus saillants, ceux qui le ou la caractérisent vraiment.* Encore faut-il savoir les repérer.

Au niveau de la génération automatique d'improvisations, l'analyse motivique pourrait peut-être constituer une alternative aux approches traditionnelles basées sur des méthodes statistiques ou le raisonnement à partir de cas. Tou-

tefois, l'intérêt potentiel d'une telle application reste suspendu à la question de son efficacité dans des problématiques temps-réel, ce qui est encore loin d'être assuré.

Au niveau compositionnel enfin, le système EMI de Cope a déja fait ses preuves dans la génération de pièces imitant le style d'un compositeur particulier. Mais la perversion, le détournement de la machine de son but initial, problématique et direction de travail si chères à certains compositeurs contemporains, ne sont sans doute pas encore totalement consommés, et mille et une inimaginables déviances restent encore à imaginer. Générations automatiques de variations plus ou moins fidèles, déconstructions progressives de motifs, autodérisions, autant de plateaux pour la pensée créatrice, peuplés d'autant de machines à se souvenir, machines à oublier, machines à répéter, machines à mimer le même, oui, autant de veaux d'or et de silicium qui n'ont pas fini de hanter les labyrinthes de la pensée scientifique.

> « *Le silence était hostile et presque parfait ; il n'y avait d'autre bruit dans ce réseau de pierre que celui d'un vent souterrain, dont je ne découvris pas l'origine ; sans bruit, se perdaient dans les grottes des filets d'eau ferrugineuse. Avec horreur, je m'accoutumai à ce monde suspect ; il me semblait impossible qu'il pût exister autre chose que ces cryptes à neuf portes et de longs souterrains qui se ramifiaient.* » - Jorge Luis Borges

Annexe A

Corpus

Le corpus est organisé par tonalités. Le champ **Taille** comprend le nombre de notes du fichier MIDI correspondant, silences inclus.

Le champ **Structure** est de la forme $l + n * g + q$, ou l est la durée de l'éventuelle levée, n est le nombre de cycles (ou grilles), g est la longueur de la grille et q la durée de l'éventuelle queue. Toutes ces valeurs sont exprimées en nombre de mesures.

Le champ champ **Sous-Titre** comporte la valeur « Take# » lorsqu'il y a plusieurs versions du même morceau, la valeur « Section# » lorqu'un morceau a été découpé pour des raisons de structure.

Morceaux en Si bémol mineur (2)

Titre	Sous-Titre	Forme	Taille	Tempo	Structure
Diverse	-	AABA	378	250	3+2*32+1
Sement	-	AABA	410	250	3+2*32+1

Morceaux en Fa majeur (27)

Titre	Sous-Titre	Forme	Taille	Tempo	Structure
Barbados	Take1	Blues	454	225	5*12+1
Barbados	Take2	Blues	448	245	1+5*12+1
Billie's Bounce	Take1	Blues	238	170	3*12+1
Billie's Bounce	Take2	Blues	269	170	1+4*12
Billie's Bounce	Take3	Blues	307	170	1+4*12+1
Billie's Bounce	Take4	Blues	240	165	3*12+1
Billie's Bounce	Take5	Blues	319	160	1+4*12+1
Chasin' the Bird	Take1	IGR[1]	188	175	1*32+1
Chasin' the Bird	Take2	IGR	214	175	1+1*32+1

[1] Forme « I Got Rythm », dite aussi « Anatole » (cf. Siron[41]).

Chasin' the Bird	Take3	IGR	219	190	1+1*32+1
Chasin' the Bird	Take4	IGR	471	235	1+2*32+3
Chasin' the Bird	Take5	IGR	432	240	1+2*32+1
Now's the Time	Take1	Blues	271	140	3*12+1
Now's the Time	Take2	Blues	259	135	3*12+1
Au Privave	Take1	Blues	696	200	11*12+1
Au Privave	Take2	Blues	452	210	4*12+1
Salt Peanuts	Take1	IGR	698	250	16*5+32
Salt Peanuts	Take2	IGR	376	250	2+2*32
Salt Peanuts	Take3	IGR	435	250	2*32
Sippin' at Bells	Take1	Blues	120	204	2*12
Sippin' at Bells	Take2	Blues	160	204	2*12+1
Sippin' at Bells	Take3	Blues	161	200	2*12+1
The Squirrel	Section1	Blues	879	220	12*12+1
The Squirrel	Section2	Blues	93	220	2+1*12+1
The Squirrel	Section3	Blues	83	220	1*12+1
The Squirrel	Section4	Blues	86	220	1*12+1
The Squirrel	Section5	Blues	102	220	1+1*12+1

Morceaux en Fa mineur (2)

Titre	**Sous-titre**	**Forme**	**Taille**	**Tempo**	**Structure**
Be Bop	-	AABA	370	250	1+2*32+1
Mango Mange	-	?	684	130	4*32+5

Morceaux en Mi bémol majeur (22)

Titre	**Sous-titre**	**Forme**	**Taille**	**Tempo**	**Structure**
Blue Bird	Take1	Blues	295	125	1+2*12+1
Blue Bird	Take2	Blues	252	125	2*12+1
Carvin' the Bird	Take1	Blues	226	210	3*12+1
Carvin' the Bird	Take2	Blues	214	210	3*12+1
Dark Shadows	Take1	?	193	64	1+2*8
Dark Shadows	Take2	?	200	89	1+2*8+1
Dark Shadows	Take3	?	216	89	1+2*8+1
Dark Shadows	Take4	?	216197	100	1*2*8+1
Dewey Square	Take1	AABA	523	185	3*32+1
Dewey Square	Take2	AABA	295	182	2*32+1
Dewey Square	Take3	AABA	313	130	2*32+1
Groovin' High	Take1	ABAC	465	220	2+2*32+1
Groovin' High	Section2	ABAC	470	230	2+2*32+1
Groovin' High	Section3	ABAC	152	230	1*32
Groovin' High	Take4	ABAC	895	250	2+5*32
Home Cooking III	-	AABA	720	215	4*32
Meandering	-	ABAC	383	60	1+1*32+1

Little Suede Shoes	Take1	AABA	277	156	1+1*32+1
Little Suede Shoes	Section2	AABA	347	150	1*32+1
Little Suede Shoes	Section3	AABA	248	150	1+1*32
Quasimodo	Take1	ABAC	166	140	1*32
Quasimodo	Take2	ABAC	188	145	1*32

Morceaux en Ré bémol majeur (8)

Titre	Sous-titre	Forme	Taille	Tempo	Structure
20th Century Blues	-	Blues	104	89	1+1*12
Groovin'High	Take1	ABAC	779	175	4+3*32+1
Groovin'High	Take2	ABAC	127	185	4+0*32+15
Body and Soul	-	AABA	974	94	21+2*32+1
Relaxing with Lee	Section1	AABA	269	175	1+32+1
Relaxing with Lee	Section2	AABA	267	175	1+1*32
Relaxing with Lee	Take3	AABA	270	180	1+1*32
Blues for Norman	-	Blues	890	250	10*12+1

Morceaux en Mi bémol mineur (1)

Titre	Sous-titre	Forme	Taille	Tempo	Structure
Round Midnight	-	?	523	60	8+1*32

Morceaux en Do mineur (6)

Titre	Sous-tire	Forme	Taille	Tempo	Structure
I Love Paris	Take1	?	229	125	1*32
I Love Paris	Take2	?	245	120	1+1*3201
The Bird	Section1	ABCA	420	220	2*32
The Bird	Section2	ABCA	876	220	1+4*32
My Heart Belongs to Daddy	Section1	AABC	228	165	1+1*32+1
My Heart Belongs to Daddy	Section2	AABC	93	165	0*32+16

Morceaux en Do majeur (8)

Titre	Sous-titre	Forme	Taille	Tempo	Structure
Cool Blues	Take1	Blues	233	220	1+3+12+2
Cool Blues	Take2	Blues	236	220	1+3*12+2
Cool Blues	Section2	Blues	139	165	1+2*12+1
Cool Blues	Section3	Blues	92	165	1+1*12+1
Cool Blues	Section4	Blues	168	180	1+2*12+1

Cool Blues	Section5	Blues	91	180	1+1*12
Cool Blues	Take6	Blues	988	204	1+13*12
Funky Blues	-	Blues	100	50	1+1*12

Morceaux en Ré mineur (7)

Titre	Sous-titre	Forme	Taille	Tempo	Structure
Bernie's Tune	-	AABA	612	190	2*32
A Night in Tunisia	Take1	AABA	173	170	4+1*32
A Night in Tunisia	Take2	AABA	169	175	4+1*32
A Night in Tunisia	Take3	AABA	609	160	4+2*32
A Night in Tunisia	Take4	AABA	185	180	4+1*32
A Night in Tunisia	Take5	AABA	606	175	4+2*32+1
A Night in Tunisia	Take6	AABA	767	195	4+3*32

Morceaux en Sol majeur (16)

Titre	Sous-titre	Forme	Taille	Tempo	Structure
4-F Blues	-	Blues	98	135	2*12+1
G.I.Blues	-	Blues	113	130	2*12
Okidoke	-	AABA	862	210	1+3*32+1
Ornithology	Take1	ABAC	215	200	1*32+1
Ornithology	Take2	ABAC	233	225	1+1*32+1
Ornithology	Take3	ABAC	642	230	2+3*32+1
Ornithology	Take4	ABAC	453	230	2+2*32+1
Ornithology	Take5	ABAC	902	220	2+4*32
Ornithology	Take6	ABAC	879	250	1+4*32+1
Out of Nowhere	Section1	ABAC	240	165	1+1*32+1
Out of Nowhere	Section2	ABAC	311	165	1*32+1
Out of Nowhere	Take3	ABAC	520	150	2+2*32
Out of Nowhere	Take4	ABAC	477	69	1+1*32+1
Out of Nowhere	Take5	ABAC	534	60	1+1*32+1
Out of Nowhere	Take6	ABAC	534	69	1+1*32+1
That's the Blues	-	Blues	92	69	1*12

Annexe B

Cadences

Les séquences d'accords reconnues par le système sont présentées ici telles qu'implémentées. Chaque progression harmonique est modélisée par une liste de trois éléments :

1. Le nom de séquence, tel que généralement employé dans l'analyse des grilles de jazz. le nom peut éventuellement être préfixé (cf. **Chiffrages**). `christ` indique une cadence de type « Christophe », `SD` un passage par la sous-dominante mineure, et `Blues` une séquence particulière au blues.
2. La liste des accords abstraits consituant la séquences.
3. Le type de contexte harmonique sous-tendu par la progression : *majeur*, *mineur*, ou *ambigu*.

N.B. A la différence de accords constituant les grilles, les accords abstraits figurant dans les progressions-type reconnues par le système ne comportent aucune spécification de durée.

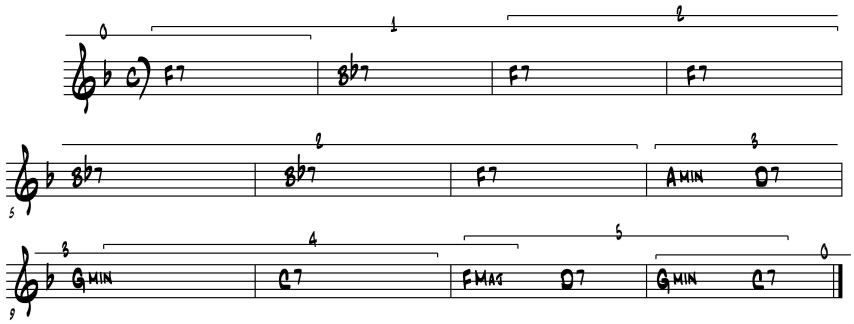
```
['IIIm-VIm-IIm-V7-I',['3 m','6 m','2 m','5 7','1 M'],'Major']
['Christ Vm-I7-IV-VIIb7-I',['5 m','1 7','4 M','7b7','1 M'],'Major']
['Christ Vm-I7-IV-IVm-I',['5 m','1 7','4 M','4 m','1 M'],'Major']
['Christ I-I7-IV-VIIb7',['1 M','1 7','4 M','7b7'],'Major']
['Christ I-I7-IV-IVm',['1 M','1 7','4 M','4 m'],'Major']
['Christ Vm-I7-IV-VIIb7',['5 m','1 7','4 M','7b7'],'Major']
['I-VIm-IIm-V7',['1 M','6 m','2 m','5 7'],'Major']
['I-VI7-IIm-V7',['1 M','6 7','2 m','5 7'],'Major']
['VIm-IIm-V7-I',['6 m','2 m','5 7','1 M'],'Major']
['IIIm-VIm-IIm-V7',['3 m','6 m','2 m','5 7'],'Major']
['IIIm-VI7-IIm-V7',['3 m','6 7','2 m','5 7'],'Major']
['III7-VI7-II7-V7',['3 7','6 7','2 7','5 7'],'Major']
['IIm-V7-I-IV',['2 m','5 7','1 M','4 M'],'Major']
['Im-VIb-IIo-V7',['1 m','6bM','2 o','5 7'],'Minor']
['IIm-V7-I',['2 m','5 7','1 M'],'Major']
```

```
['IIm-IIb7-I',['2 m','2b7','1 M'],'Major']
['SD Min IVm-VIIb7-I',['4 m','7b7','1 M'],'Major']
['SD Min IV-VIIb7-I',['4 M','7b7','1 M'],'Major']
['IIo-V7-Im',['2 o','5 7','1 m'],'Minor']
['IVm-VIIb-I',['4 m','7b7','1 M'],'Major']
['Blues IIm-V7-I7',['2 m','5 7','1 7'],'Major']
['Blues I7-IV7-I7',['1 7','4 7','1 7'],'Major']
['IIm-V7',['2 m','5 7'],'Major']
['IIo-V7',['2 o','5 7'],'Minor']
['V7-I',['5 7','1 M'],'Major']
['IIb7-I',['2b7','1 M'],'Major']
['V7-Im',['5 7','1 m'],'Minor']
['IIb7-Im',['2b7','1 m'],'Minor']
['SD Min VIIb7-I',['7b7','1 M'],'Major']
['I',['1 M'],'Major']
['V7',['5 7'],'Ambiguous']
['IIm',['2 m'],'Ambiguous']
```

Annexe C

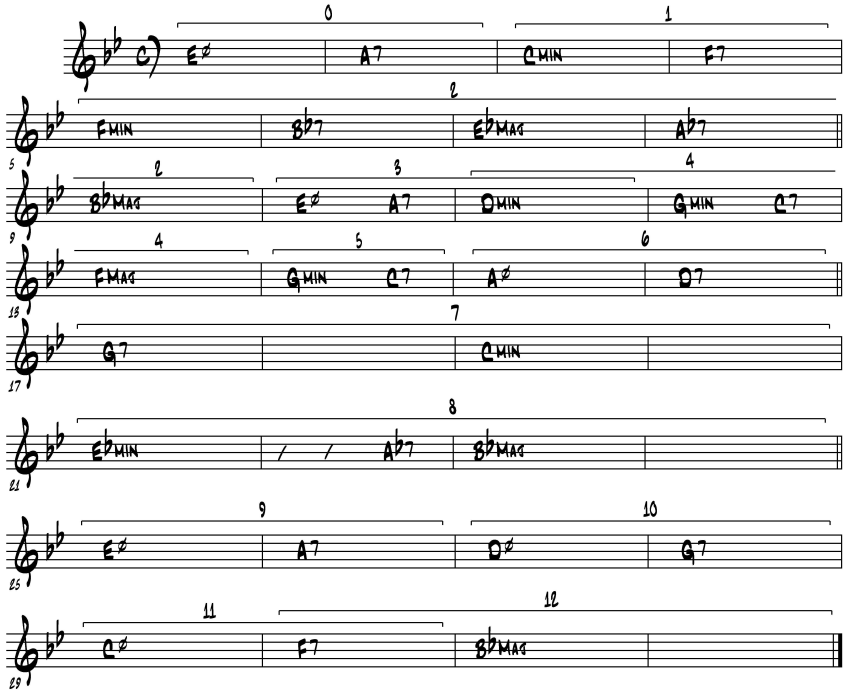
Trois analyses

Le lecteur touvera dans cette annexe deux analyses harmoniques et une analyse de contour. La détection des progressions harmoniques étant un processus de regroupement, non de segmentation, certaines cadences se chevauchent. Le résultat de l'analyse tel qu'affiché par le système (sous forme textuelle) est accompagné d'un partition pour plus de lisibilité. Sur la troisième partition figurent le contour mélodique des phrases, les extrema de hauteurs, et les « notes interdites », marquées par une flèche (cf. **Chiffrages**). La dernière page de cette annexe fait référence au paragraphe **Extentions** de la section 3.3.2.




```
0   Blues IIm-V7-I7    Major in F     2-2-4  From 45 To 4
1   Blues I7-IV7-I7    Major in F     4-4-8  From  1 To 16
2   Blues I7-IV7-I7    Major in F     8-8-4  From  9 To 28
3   IIIm-VI7-IIm-V7    Major in F   2-2-4-4  From 29 To 40
4          IIm-V7-I    Major in F     4-4-2  From 33 To 42
5      I-VI7-IIm-V7    Major in F   2-2-2-2  From 41 To 48
```

Fig. C.1 – Analyse harmonique de *Straight No Chaser*



```
 0             IIo-V7  Minor in D       4-4  From   0 To 8
 1             IIm-V7  Major in Bb      4-4  From   8 To 16
 2    Christ Vm-I7-IV  Major in Bb 4-4-4-4-4  From  16 To 36
               -VIIb7-I
 3          IIo-V7-Im  Minor in D      2-2-4  From  36 To 44
 4       VIm-IIm-V7-I  Major in F    4-2-2-4  From  40 To 52
 5             IIm-V7  Major in F       2-2  From  52 To 56
 6             IIo-V7  Minor in G       4-4  From  56 To 64
 7              V7-Im  Minor in C       8-8  From  64 To 80
 8 SD Min IVm-VIIb7-I  Major in Bb     6-2-8  From  80 To 96
 9             IIo-V7  Minor in D       4-4  From  96 To 104
10             IIo-V7  Minor in C       4-4  From 104 To 112
11             IIo-V7  Minor in Bb      4-4  From 112 To 120
12              V7-I   Major in Bb      4-8  From 116 To 128
```

FIG. C.2 – Analyse harmonique de *Stella By Starlight*

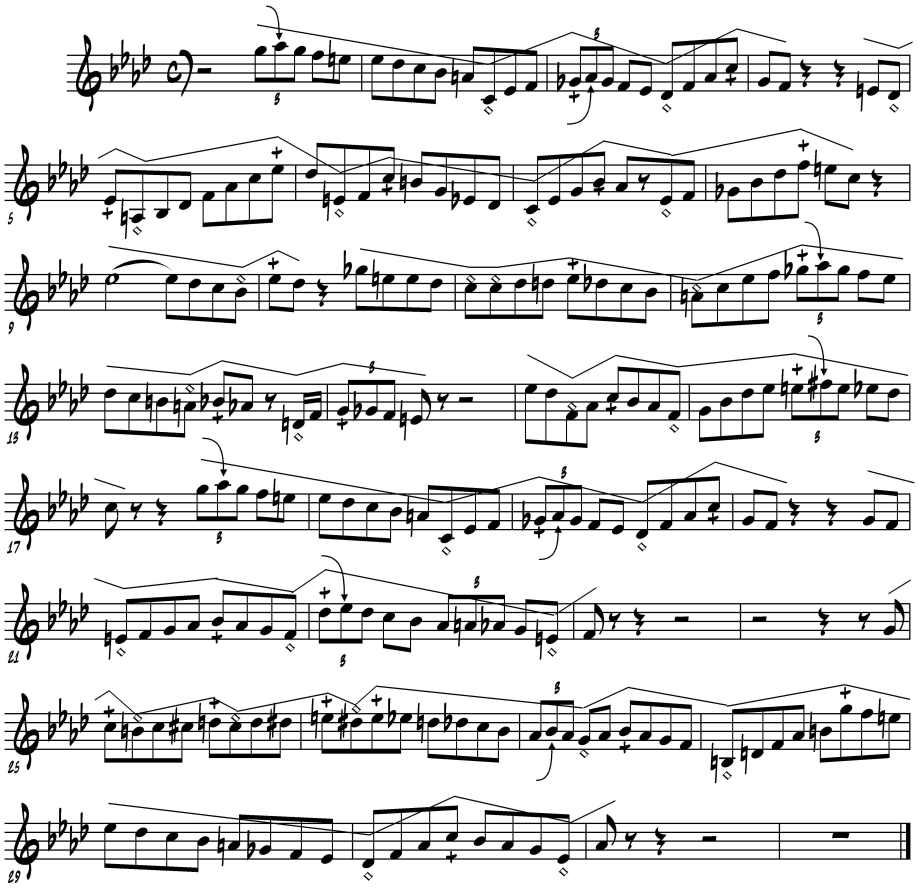


FIG. C.3 – Segmentation en phrases et analyse de contour de *Donna Lee*

AbM	**F7**	**Bb7**	
Bbm	Eb7	AbM	Ebm / Ab7
DbM	Dbm / Gb7	AbM	**F7**
Bb7		Bbm	Eb7
AbM	**F7**	**Bb7**	
C7		Fm	C7
Fm	C7	Fm	**G7**
AbM / F7	Bbm / Eb7	AbM	Bbm / Eb7

```
 0           IIm-V7-I      Major in Ab   2-2-4 From 124 To 4
 1                 V7  Ambiguous in Bb       4 From   4 To 8
 2                 V7  Ambiguous in Eb       8 From   8 To 16
 3           IIm-V7-I      Major in Ab   4-4-4 From  16 To 28
 4           IIm-V7-I      Major in Db   2-2-4 From  28 To 36
 5 SD Min IVm-VIIb7-I      Major in Ab   2-2-4 From  36 To 44
 6                 V7  Ambiguous in Bb       4 From  44 To 48
 7                 V7  Ambiguous in Eb       8 From  48 To 56
 8           IIm-V7-I      Major in Ab   4-4-4 From  56 To 68
 9                 V7  Ambiguous in Bb       4 From  68 To 72
10                 V7  Ambiguous in Eb       8 From  72 To 80
11              V7-Im      Minor in F      8-4 From  80 To 92
12              V7-Im      Minor in F      4-4 From  92 To 100
13              V7-Im      Minor in F      4-4 From 100 To 108
14                 V7  Ambiguous in C        4 From 108 To 112
15       I-VI7-IIm-V7      Major in Ab 2-2-2-2 From 112 To 120
16           IIm-V7-I      Major in Ab   2-2-4 From 116 To 124
```

FIG. C.4 – Analyse harmonique de Donna Lee

Nous avons reproduit sur la figure C.4 la grille d'accords de *Donna Lee* ainsi que l'analyse harmonique produite par le système. La grille est ici notée sous forme traditionnelle, à savoir un tableau dont on parcourt successivement chaque ligne de gauche à droite, chaque case correspondant à une mesure. Les accords en gras montrent les carences de notre méthode d'analyse, évoquées à la section 3.3.2.

Annexe D

Patterns

Le lecteur trouvera dans cette annexe quelques patterns utilisés par le système dans les algorithmes de localisation. Les patterns provenant du lexique d'Owens sont préfixés par « **ow** ». Nous y adjoinions deux patterns extraits par l'algorithme **FlExPat** de Rolland[38] (préfixés « **pat** ») et deux autres patterns issus de notre propre connaissance du jazz et de Charlie parker (préfixés « **mypat** »).

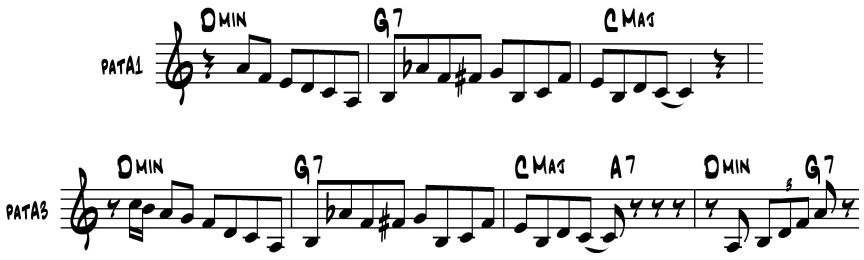


Fig. D.1 – Deux patterns extrait par FlExPat

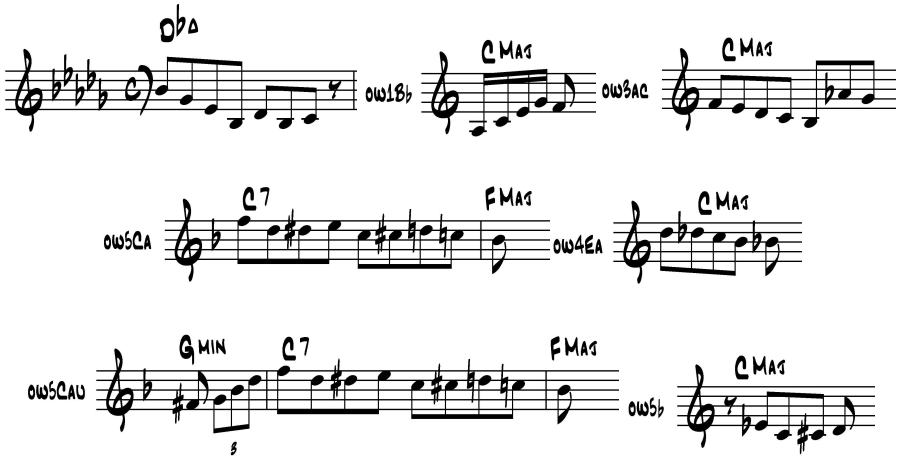


Fig. D.2 – Quelques patterns d'Owens

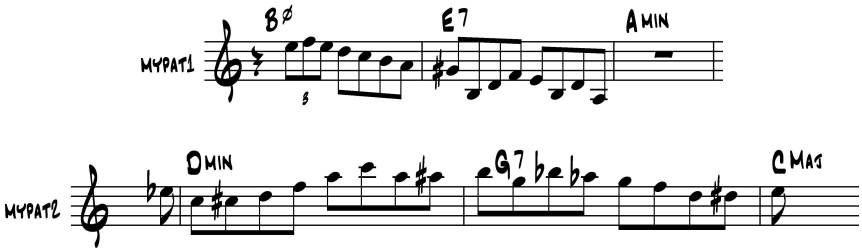


Fig. D.3 – Deux patterns « personnels »

Annexe E

Interfaces

Sont regroupées dans cet appendice l'ensemble des interfaces utilisateur développées pour notre système d'analyse :

- Un outil de création de corpus permettant d'engendrer des sous-corpus par spécification de contraintes (tempo, tonalité, forme, unité rythmique de référence).
- Un outil d'exploration de corpus, permettant l'accès aux fichiers MIDI et au données enrichies générées dans la phase de pré-traitement (cf. section 3.3).
- Deux interfaces pour la localisation de patterns (δ-*approximate matching* et *modèle d'édition*). L'utilisateur est libre se specifier les critères sur lesquels identifier les occurences ainsi que les poids (dans le cas de notre modèle d'édition) attribués aux fonction de contribution.

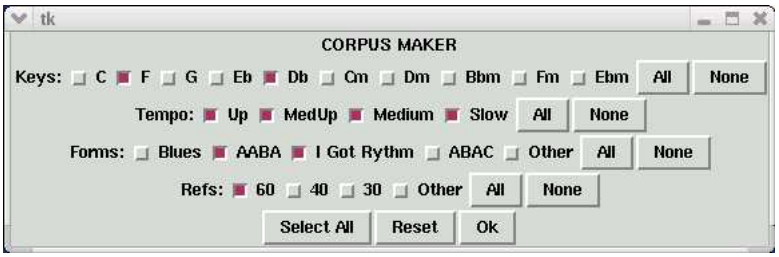


FIG. E.1 – Outils de création de sous-corpus

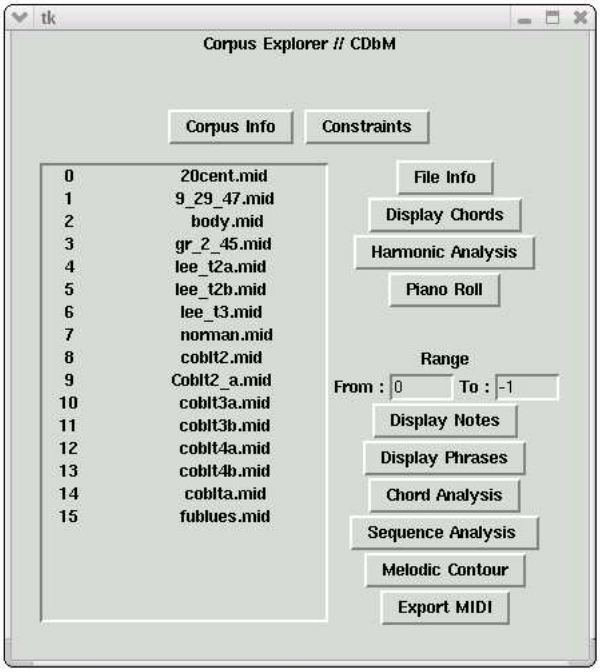


FIG. E.2 – Outil d'exploartion de corpus

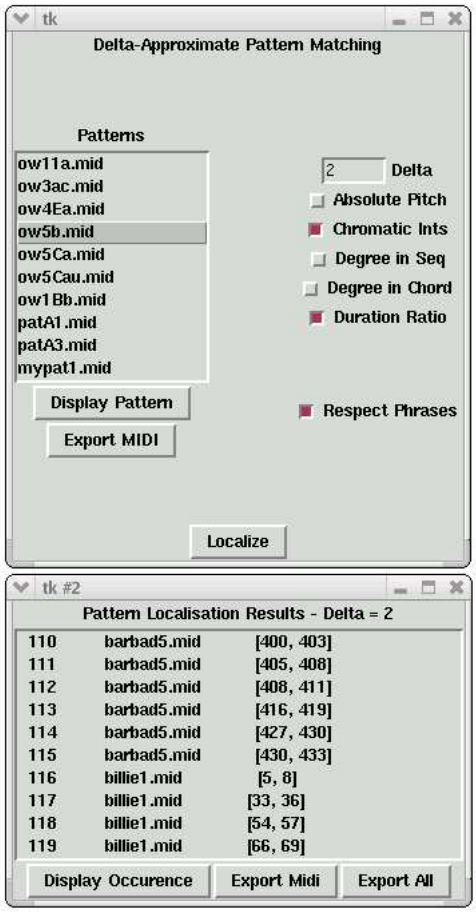


FIG. E.3 – Interface de localisation d'occurences δ-approximatives

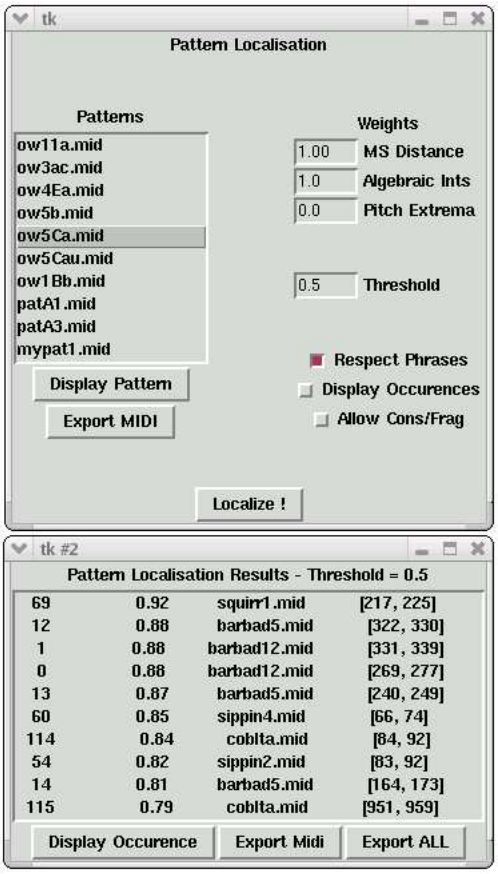


FIG. E.4 – Interface de localisation d'occurences approximatives à l'aide d'un modèle d'édition

Table des figures

Bibliographie

[1] Adorno, T.W., *Théorie de la nouvelle musique*, Coll. *Tel*, Gallimard, Paris, 1979, 222 pages.

[2] Assayag, G. & Pachet F., Dossier « *IA et Musique* », In *Les Dossiers de l'AFIA (Association Française pour l'Intelligence Artificielle)* n°23, octobre 1995.

[3] Assayag, G., Dubnov, L., Lartillot, O., et Bejerano, G., « *Using Machine-Learning Methods, for style Modelling* », In *Computer*, 36(10), p.73-80, IEEE Computer Society, 2003.

[4] Baudoin, P., *Jazz mode d'emploi : petite encyclopédie des données techniques de base* - Volumes 1 & 2, Coll. *Théories*, Ed. Outre-Mesure, Paris, 1990.

[5] Bharucha, J.J., *« MUSACT : a connexionnist model of musical harmony »*, In *Proceedings of the Cognitive Science Society*, Lawrence Erlbaum, 1987.

[6] Cambouropoulos, E., « *A General Pitch Interval Representation : Theory and Applications* », *Journal of New Music Research*, 25(3), p.231-251, 1996.

[7] Cambouropoulos, E., Crawford, T. and Iliopoulos, C.S., « *Pattern Processing in Melodic Sequences : Challenges, Caveats and Prospects* », In *Proceedings of the AISB'99 Convention (Artificial Intelligence and Simulation of Behaviour)*, Edinburgh, U.K., 1999.

[8] Cambouropoulos, E., Smaill, A. & Widmer, G., *« A Clustering Algorithm for Melodic Analysis »*, In *Proceedings of the Diderot'99 Forum on Mathematics and Music*, Vienne, 1999.

[9] Cambouropoulos, E., Crochemore, M., Iliopoulos, C.S., Mouchard, L. and Pinzon, Y.J., « *Algorithms for Computing Approximate Repetitions in Musical Sequences* », In *Proceedings of the AWOCA'99 Workshop (Australasian Workshop on Combinatorial Algorithms)*, Perth, Western Australia, 25-27 July 1999.

[10] Cambouropoulos, E., « *The Local Boundary Detection Model (LBDM) and its Application in the Study of Expressive Timing* », In *Proceedings of the International Computer Music Conference (ICMC'2001)*, La Havane, Cuba, septembre 2001.

[11] Cope, D., « *Pattern-Matching as an Engine for the Comuter Simulation of Musical Style* » In *Proceedings of the International Computer Music conference (ICMC)*, Glasgow, 1990.

[12] Cope, D., *Experiments in Musical Intelligence*, *Computer Music and Digital Audio* Series, vol. 12, A-R Editions Inc., 1996, 263 pages.

[13] Crochemore, M. et Ritter W., *Text Algorithms*, Oxford University Press, 1994.

[14] Dannenberg, B., et Hu, N., « *Pattern Discovery Techiniques for Music Audio* », In *Proceedings of the 3^{rd} International Conference on Music Information Retrieval (ISMIR)*, Paris, 2002.

[15] Goldsen, M.H., *Charle Parker Omnibook,* Atlantic Music Corp., U.S.A., 1978.

[16] Good, M., « *MusicXML for Notation and Analysis* », in *The Virtual Score*, ed. W.B.Hewlett & E.Selfridge-Field, MIT Press, p.113-124.

[17] Kruskal, J.B. et Liberman, M., « *The Symetric Time-Waring Problem : From Contiuous to Discrete* », In *Time Warps, String Edits, and Macromolecules : The Theory and Practice of Sequences Comparison*, Addison-Wesley, 1983.

[18] Kruskal, J.B. et Sankoff, D., « *An Anthology of Algorithms and Concepts for Sequences Comparison* », In *Time Warps, String Edits, and Macromolecules : The Theory and Practice of Sequences Comparison*, Addison-Wesley, 1983.

[19] Lartillot, O., *Fondements d'un Système d'Analyse Computationnelle suivant une Modélisation Cognitiviste de l'Ecoute,* Thèse de Doctorat de l'Université Paris vi, Paris, 2004, p.116.

[20] Lerdahl F., et Jackendoff, R., *A Generative Theory of Tonal Music*, The MIT Press, 1983.

[21] Levenshtein, V.I., « *Binary Codes Capable of Correcting Deletions, Insertions, and Reversals* », In *Cybernetics ans Control Theory* 10(8) : 707-710, 1965.

[22] Meudic, B., « *A Causal Algorithm for Beat Tracking* », 2^{nd} *Conference on Understanding and Creating Music*, Caserta, Italy, 2002.

[23] Meudic, B. « *Musical Pattern Extraction : From Repetition to Musical Structure* », In *Proceedings of CMMR (Computer Music Modeling and Retrieval)*, Montpellier, mai 2003.

[24] Meudic, B., *Détermination automatique de la pulsation, de la métrique et des motifs musicaux dans des interprétations à tempo variable d'oeuvres polyphoniques,* Thèse de doctorat de l'Université Paris vi, Paris, 2004, 237 pages.

[25] Mongeau, M. et Sankoff, D., « *Comparison of Musical Sequences* », In *Computer and the Humanities*, 24 :161-175, 1990.

[26] Nattiez, J.-J., *Fondements d'une sémiologie de la musique*, Union Générale d'Editions, Paris, 1975.

[27] Owens, T., *Charlie Parker : Techniques of Improvisation*, PhD. Thesis, Dept. of Music, Univerity of California Los Angeles, 1974, 2 vols., 900 pages.

[28] Pachet, F., « *Representing Knowledge used by Jazz musicians* », In *Proceedings of the International Computer Music conference (ICMC)*, Montreal, p.285-288., 1991.

[29] Pachet, F., *« An Objet-Enriented Representation of Pitch-Class, Intervals, Sclaes and Chords, The basic MusES », Premières Journées d'Informatique Musicale*, LaBRi, Université de Bordeaux, 19-34, 1993.

[30] Pachet F., « *Sur la structure algébrique des séquences d'accords de Jazz* », *Journées pour l'Informatique Musicale 1998 (JIM'98)*, Agelonde, France, 1998.

[31] Pachet, F., « *Computer Analysis of Jazz Chord Sequences : Is Solar a Blues ?* » In *Readings in Music and Artificial Intelligence, Harwood Academic Publishers*, Miranda, E., 2000.

[32] Pachet, F., « The Continuator : Musical Interaction with Style ». In *Proceeding of the International Computer Music conference (ICMC)*, ICMA, pages 211-218, septembre 2002.

[33] Pachet, F., et Aucuturier J.-J., « *Music Similarity Measures : What's the Use ?* », In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, Paris, 2002.

[34] Peeters, G. et al., « Instrument Sound Description in the Context of MPEG-7 », In *Proceeding of the International Computer Music conference (ICMC)*, 2000.

[35] Peeters, G. et al., « *Towards Automatiquc Music Audio Summary Generation from Signal Analysis* », *In Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, Paris, 2002.

[36] Ramalho, G., *Construction d'un Agent Rationel Jouant du Jazz*, Thèse de doctorat de l'Université Paris vi, Paris, 1997, 306 pages.

[37] Ramalho, G., Ganascia J.-G., « *Simulating Creativity in Jazz Performance* ». In *Proceedings of the National Conference in Artificial Intelligence*, pp. 108-113, AAAI-94, AAAI Press, Seattle, 1994.

[38] Rolland, P.-Y., *Découverte Automatique de Régularités dans les Séquences et Application à l'Analyse Musicale »*, Thèse de Doctorat de l'Université Paris vi, Paris, 1998, 335 pages.

[39] Selfridge-Field, E., *Beyond MIDI : The Handbook of Musical Codes*, MIT Press, Cambridge MA, 1997.

[40] Siron, J., *Des mots aux sons*, Ed. Outre-Mesure, Paris, 2001, 208 pages.

[41] Siron, J., *La partition intérieure : jazz, musiques improvisées*, Coll. *Théories*, Ed. Outre-Mesure, Paris, 2001.

[42] Smith, T.F. et Waterman M.S., « *Identification of Common Molecular Subsequences* », In *Journal of Molecular Biology*, 147 : 195-197, 1981.

[43] Stammen, D.R. and Pennycook, B., « Real-Time Recognition of Melodic Fragments Unsing the Dynamic Timewarp Algorithm », In *Proceedings of the International Computer Music conference (ICMC),* 1993.

[44] Steedman, M.J., *A Generative Grammar for Jazz Chord Sequences*, *Music Perception*, University of California Press. 2(1), 52-77, 1984.

[45] Waterman, M. S., *Introduction to Computational Biology : Maps, Sequences and Genomes*, Chapman & Hall, Londres, 1995.