Improviser des séquences d'accords de jazz avec des grammaires formelles

Marc Chemillier
Université de Caen
marc@info.unicaen.fr

1. Introduction

Cet article décrit un logiciel implémentant un générateur d'accords dans un environnement de type « DJ mix », d'après les principes de l'harmonie jazz. Le programme produit une sortie midi au moyen de 1) un générateur de séquences d'accords de jazz à base de règles de grammaire, et 2) un dictionnaire de mesures précomposées, associant les symboles d'accords à des échantillons au format midi. L'aspect harmonique d'un accompagnement de jazz repose sur une grille harmonique tonale de *n* accords, répétée en boucle avec des variations autant de fois que nécessaire. Le pianiste improvise une réalisation de la séquence d'accords, en variant les renversements, et en introduisant de nouveaux accords par l'application de substitutions harmoniques à la grille d'origine. Le sujet principal de cet article est l'implémentation de cette technique de réalisation et de substitution harmonique.

2. La grammaire de Steedman

En 1984, Steedman a proposé une grammaire dans le but de décrire les substitutions harmoniques utilisées par les musiciens de jazz. Sa grammaire est exprimée sous la forme de six règles de réécriture prenant comme séquence initiale la grille de base du blues et générant une variante du type de celles jouées par les musiciens de jazz moderne [Steedman 1984]. Il a déduit les règles de l'analyse d'un corpus de huit grilles empruntées à un livre de [Coker 1964]. Dans le prolongement de son article, d'autres auteurs ont apporté de nouveaux développements à son intuition originale [Johnson-Laird 1991], [Pachet 1999], [Chemillier 2001].

L'exemple suivant montre deux états d'une grille de blues en *do* (*C* en chiffrage anglo-saxon). Le premier est la grille de base.

C	C	C	C7
F	F	C	C
G7	G7	C	C

C	C	C	Gm7
F	F	C	C
Dm7	G7	C	C

Le second résulte de deux substitutions appliquées respectivement à la quatrième mesure :

C7 Æ Gm7 C7

et aux neuvième et dixième mesures :

G7 G7 Æ Dm7 G7

La grammaire de Steedman consiste en six règles de ce type, chacune exprimée avec une variable *x* qui désigne l'un quelconque des degrés *C, C#, D,* ... Ainsi, chaque règle de Steedman correspond en réalité à douze règles différentes. Les fonctions *Sdx, Dx, Stbx, Stx, Mx, Lx* sont respectivement la sous-dominante, la dominante, le sus-tonique bémole, la sus-tonique, la médiante, et la sus-dominante de *x*. Dans la règle 3, la variable *w* désigne un accord indéterminé, dont le degré n'a pas été modifié par une règle appliquée précédement, et dont le chiffrage n'est pas 7. Les accolades de la règle 6 indiquent un choix.

Règle 1:	*x Æ x x*
	x7 Æ x x7
	xm7 Æ x xm7
Règle 2:	*x Æ x Sdx*
	x7 Æ x7 Sdx
	xm7 Æ xm7 Sdx
Règle 3a:	*w x7 Æ Dx7 x7*
	w x7 Æ Dxm7 x7
Règle 3b:	*w xm7 Æ Dx7 xm7*
Règle 4:	*Dx7 x7 Æ Stbx7 x7*
	Dx7 x Æ Stbx x
	Dx7 xm7 Æ Stbxm7 xm7
Règle 5:	*x x x Æ x Stxm Mxm*
Règle 6:	*x x {Dx, Stxm7, Lxm7} Æ x x#°7 {Dx, Stxm7, Lxm7}*
	xm xm {Dx, Stxm7, Lxm7} Æ xm x#°7 {Dx, Stxm7, Lxm7}

Tous les accords sont à la fois des symboles terminaux et non-terminaux. Une règle supplémentaire permet de réécrire l'axiome de la grammaire en une séquence correspondant à la grille de base du blues.

3. Parsing automatique avec Lex

Plus récemment, Steedman a publié un autre article consacré au même sujet, pour étudier certaines difficultés qui apparaissent dans l'implémentation d'un analyseur construit à partir de sa grammaire, à cause du fait que la grammaire est context-sensitive [Steedman 1996]. En effet, comme on peut le voir ci-dessus, certaines règles ont plus d'un symbole non terminal du côté gauche (c'est le cas pour la règle *G G7 Æ Dm7 G7* dans l'exemple précédent), et par conséquent ne sont pas de type régulier, ni même de type context-free, dans la classique hiérarchie de Chomsky. Il semble impossible d'implémenter cette grammaire sous forme d'un analyseur avec des outils comme Lex ou Yacc.

Nous avons proposé une solution au problème en formalisant certains aspects de la grammaire qui ne sont pas explicite dans les règles décrites ci-dessus, concernant la durée des accords. Dans la seconde règle de l'exemple précédent, les accords des deux côtés de

la règle ont la même durée. Mais ce n'est pas le cas dans la première règle où les accords du côté droit ont une durée moitié plus courte que l'accord du côté gauche. Aussi l'action de règle de ce type doit-elle être limitée, dans la mesure où la durée des accords dans une grille de jazz n'est généralement pas plus courte qu'un quart de mesure. Pour prendre en compte cette restriction, nous avons introduit une modification dans la forme des règles de Steedman, en ajoutant un nouveau symbole à la grammaire pour représenter les barres de mesure (noté /). Par exemple, la règle

x Æ x x

est remplacée par la règle

/ x / Æ / x x /

ce qui signifie que la règle précédente ne peut être appliquée que dans une case de la grille ne comportant qu'un seul accord. Les nouvelles règles obtenues à partir des règles originales sont désormais capable de contrôler que la durée des accords reste supérieure au quart de la mesure.

On peut prouver facilement que la nouvelle forme de cette grammaire est équivalente à un automate fini. La durée d'un accord étant restreinte à des valeurs supérieures au quart de la mesure, et le nombre de mesures étant fixé, il est évident que le nombre de séquences d'accords possibles générés par la grammaire est fini. Aussi peut-il être engendré par un automate fini, et nous avons construit un algorithme calculant l'automate correspondant.

En utilisant Lex, nous avons implémenté cet automate sous forme d'un analyseur, qui prend en entrée une séquence d'accords, et affiche Reconnu si la séquence peut être engendré par la grammaire, et Non reconnu dans le cas contraire. Pour réaliser cette implémentation, Lex utilise une description de l'automate sous forme d'expression régulière, et produit un programme écrit en C qui implémente l'analyseur. Le code décrivant l'analyseur est reproduit dans [Chemillier 2001]. L'analyseur obtenu prend en entrée une séquence d'accords séparés par une barre de mesure (/), comme dans l'exemple ci-après, dans lequel la séquence d'entrée est le célèbre *Blues for Alice* composé par Charlie Parker.

$ blues
> F / Em7 A7 / Dm7 G7 / Cm7 F7 / Bb7 / Bbm7 Eb7 / Am7 / Abm7 Db7 / Gm7 / C7 / F7 / Gm7 C7
Reconnu

4. Langages formels d'accords et renversements échantillonnés

Notre logiciel générant des accords de séquence de jazz est une combinaison de deux modules. Le premier applique aléatoirement des règles de substitution à la séquence d'accords, engendrant ainsi un langage formel d'accords de type régulier. Le second module choisit des renversements pour les accords successifs de la séquence obtenue, en sélectionnant des renversements échantillonnés dans une base de données. L'ensemble du

programme a été écrit en Lisp dans l'environnement OpenMusic, qui est un langage visuel pour la composition musicale développé à l'Ircam. Des exemples de fichiers midi calculés par le programme sont disponibles sur le Web (www.info.unicaen.fr/~marc/publi/grammaires), et le code Lisp du programme est reproduit dans [Chemillier 2001].

Le module de renversement d'accords transforme les accords en données midi afin de produire une séquence midi qui est jouée par un synthétiseur. Il repose sur une table d'association entre accords et données midi préfabriquées appelées *samples midi*. Cette table n'est pas limitée a des entrées constituées d'un seul accord, mais associe des sample midi a des séquences de n accords consécutifs. Cela signifie que le module lit la séquence d'entrée à travers une fenêtre de longueur n. Ainsi, le modèle formel de ce module est-il une transduction à états finis.

Le module de substitution opère en choissant aléatoirement une position dans la séquence d'accords où une règle peut être appliquée. La procédure de base de ce module prend en argument une grille et un entier k, et retourne une nouvelle grille obtenue après k application du processus de substitution à la grille d'entrée. Voici deux exemples, avec respectivement une et dix substitutions appliquées à la grille de base du blues mémorisée dans la variable *grille-base*. Dans ces exemples, les accords engendrés par le processus de substitution sont marqués avec une étoile.

? (reecriture *grille-base* 1)
((c) / (c) / ((g *) 7) / (c 7) / (f) / (f) / (c) / (c) / (g) / (g 7) / (c) / (c))

? (reecriture *grille-base* 10)
((c) / (c) / (c) / ((c# *) 7) (c 7) / (f) / ((f# *) 7) / ((f *) 7) / ((bb *) 7) / ((eb *)) ((eb *) 7) / ((d *) 7) ((c# *)) / (c) / (c))

Dans l'état actuel du logiciel, l'utilisateur peut interagir en temps réel avec lui, réalisant ainsi une véritable improvisation. Cette interaction peut se faire de trois différentes façons.

Tout d'abord on peut modifier la profondeur du processus de substitution, qui correspond au nombre k de règles appliquées dans l'exemple ci-dessus. Pendant qu'une grille est jouée, l'utilisateur a la possibilité de modifier la valeur de k pour le calcul de la grille suivante. Ainsi, la séquence harmonique peut-elle évoluer entre l'état de base de la grille sans substitution, et un état très modifié comportant de nombreuses substitutions.

La seconde possibilité d'interagir avec le système est d'orienter manuellement le choix des accords. Cette possibilité est étroitement liée à la propriété que nous avons mentionnée dans la section précédente concernant le type régulier du langage de séquences d'accords engendré par la grammaire de Steedman (restreintes à des durées supérieures à la pulsation). L'accord $X_i{+}1$ qui peut suivre un accord donné X_i est le résultat de l'activation de n'importe quelle règle de substitution appliquée à la séquence d'accords courante, mais au-delà, cet accord peut résulter également de l'activation de k règles successives appliquées de manière cumulative. Par exemple, si l'on considère la séquence suivante

$XYXXXZ,$

le successeur de Y est X. Si l'on applique la règle de substitution $XZÆZZ,$ on obtient la variante

$XYXXZZ, \quad k = 1,$

ce qui ne change pas les successeurs possibles de Y. Par contre, si on itére le processus en appliquant une deuxième, puis une troisième fois la règle, on obtient les variantes

$XYXZZZ, \quad k = 2,$
$XYZZZZ, \quad k = 3,$

où les successeurs possibles de Y sont désormais X et Z. Bien que k puisse prendre n'importe quelle valeur entière, il est possible de calculer tous les X_{i+1} pouvant suivre un X_i donné, grâce au fait que la grammaire de Steedman est équivalente à un automate fini. D'une manière usuelle, cet automate peut être représenté par un graphe avec un nombre fini d'états, et des flèches représentant les transitions entre états. Durant le calcul, il est possible d'afficher l'état exact du système et de donner à l'utilisateur la possibilité de choisir quelle transition sera activée à partir de l'état courant.

Une troisième manière d'interagir avec le système est de choisir de nouveaux samples midi pour le module de renversement. La base de données contenant tous les samples midi disponibles est organisée selon différentes tables correspondant à des effets variés, des ambiances contrastées, etc. Dans l'état présent, les renversements stockés dans la base de données sont des arrangements dans le style des musiques électroniques actuelles de type house, ambient (comprenant basses, percussions et bruitages divers). Une table très simple d'arrangements pour piano dans le style boogie-woogie est disponible sur le Web à l'adresse indiquée ci-dessus.

4. DJing avec un générateur d'accords

Cette manière d'interagir avec le système en choisissant des samples midi correspondant à différents types de renversements rappelle la technique des DJ consistant à choisir des samples dans des disques et à les mixer les uns avec les autres. La tâche du DJ n'est autre que la sélection d'extraits de musique précomposée (stockée sur des vinyles), et la combinaison de ces extraits à l'intérieur d'une nouvelle composition (réalisée avec deux platines connectées à un mixeur). Il existe de nombreux logiciels simulant la tâche du DJ en permettant le mixage en temps réel de différents types de fichiers audio (wav, aiff, mp3). Notre système partage quelques aspects en commun avec ce type de logiciel. La principale différence réside dans le fait que les samples ne sont pas audio mais midi, et surtout qu'ils ne sont pas choisis manuellement, mais à l'aide d'une procédure automatique implémentant la grammaire de substitution décrite ci-dessus. Aussi notre générateur d'accords peut-il être considéré comme une extension des traditionnels logiciels de DJ.

Dans un développement futur du système, nous envisageons de donner à l'utilisateur la possibilité de mettre à jour la table des renversements en temps réel, en jouant de

nouveaux renversements sur un clavier midi connecté à l'ordinateur. Une extension de ce processus de mise à jour pourrait consister à appliquer le schéma d'apprentissage statistique développé par [Assayag 1999]. Ce shéma comporte un algorithme d'analyse incrémental basé sur la technique de compression de Lempel-Ziv. Pendant que le système fonctionne, l'utilisateur joue des renversements correspondant aux accords joués par le système. Puis les caractéristiques stylistiques (textures, enchaînements, etc) de ces renversements sont mémorisés par le système, et ajoutés dans la base de données de renversements pour une sélection future dans le processus d'improvisation.

5. Bibliographies

Assayag G., Dubnov S., Delerue O., Guessing the Composer's Mind: Applying Universal Prediction to Musical Style, *Proc. of the ICMC 99*, Beijing, China (I.C.M.A., San-Francisco, 1999).

Chemillier M., Générateurs musicaux et singularités, *JIM 99, 6èmes journées d'informatique musicale* (CNET-CEMAMu, Issy-les-Moulineaux, 1999) 167-177.

Chemillier M., Grammaires, automates et musique, F. Pachet (éd.), *Informatique musicale* (Hermès, Paris, à paraître 2001).

Coker J., *Improvising Jazz*, 1964, réédition Fireside, 1987.

Dannenberg R.B., Mont-Reynaud B., Following an Improvisation in Real Time, *Proc. of the ICMC 87*, Urbana-Champaign (I.C.M.A., San-Francisco, 1987) 241-248.

Johnson-Laird P., Jazz Improvisation: A Theory at the Computational Level, in: Cross I., Howell P., West R. (eds.), *Representing Musical Structure* (Academic Press, 1991) 291-326.

Pachet F., Surprising harmonies, *JIM 99, 6èmes journées d'informatique musicale* (CNET-CEMAMu, Issy-les-Moulineaux, 1999) 187-206.

Pachet F., Computer Analysis of Jazz Chord Sequences. Is Solar a Blues ?, in: Miranda E. (ed.), *Readings in Music and Artificial Intelligence* (Harwood Academic Publishers, 2000).

Steedman M.J., A Generative Grammar for Jazz Chord Sequences, *Music Perception* **2** (1) (1984) 52-77.

Steedman M.J., The Blues and the Abstract Truth: Music and Mentals Models, in: A. Garnham, J. Oakhill (eds.), *Mentals Models in Cognitive Science* (Erlbaum, Mahwah, NJ, 1996).

Ulrich J.W., The analysis and synthesis of jazz by computer, *Fifth International Joint Conference o Artificial Intelligence* (1977) 865-872.