ANR-14-CE24-0002-01

Projet DYCI2,
WP2 Apprentissage interactif de structures musicales,
WP2.1 Apprentissage de structures multi-dimensionnelles

Rapport de livrable :

L2.1.1 Apprentissage de structures multi-dimensionnelles, première version algorithme.

Livrable	Date	Contributeurs	Rédacteurs	Contenu
L2.1.1 Version 01	Mars 2016	K. Déguernel, E. Vincent (Inria), G. Assayag (Ircam-STMS)	K. Déguernel	Maquette Logicielle, Rapport scientifique

Résumé

Ce document présente les recherches menées dans la thèse de Ken Déguernel qui ont débouché sur trois contributions : a) une nouvelle méthode d'improvisation qui combine une mémoire musicale probabiliste apprise sur un corpus de morceaux et un oracle des facteurs déterministe qui représente l'improvisation en cours ; b) une méthode d'apprentissage de modèles multidimensionnels (mélodie, accords) ; c) une méthode de communication entre différents systèmes d'improvisation basée sur la théorie du « message passing » probabiliste.

Adresse du livrable logiciel

DYCI2_WP2_L2.1.1.zip
sur
https://forge.ircam.fr/p/Dyci2/

Projet DYCI2 - Livrable 2.1.1

Inria

Un pas vers l'improvisation automatique multidimensionnelle

Ken Déguernel

Directeurs de thèse : Emmanuel Vincent (Inria) et Gérard Assayag (Ircam)

11 mars 2016

Table des matières

1 Introduction

Les systèmes d'improvisation actuels permettent de prendre en compte une information unidimensionnelle (typiquement une mélodie représentée par une succession de hauteurs) fournie en direct par un musicien ou lue dans un corpus afin de générer de nouvelles improvisations par une recombinaison

du matériau musical.
Ces improvisations générées peuvent être, par exemple, dirigées par la connaissance antérieure d'un scénario à suivre [12] ou par une écoute active permettant au système de réagir en direct à son environnement [4].
Cependant, ces systèmes n'ont pas la possibilité de prendre en considération les corrélations existantes entre plusieurs dimensions musicales (hauteurs, harmonies, durées, intensités, timbres...).

2 Modèles probabilistes du langage

Dans [15], Raczyński et al. exploitent des méthodes issues de la modélisation probabiliste du langage pour effectuer des harmonisations de mélodies pour un corpus de musique classique. Dans [7], nous avons adapté ces méthodes pour une tâche de génération de mélodie capable de prendre en considération plusieurs dimensions musicales.
L'objectif est de prédire quelle mélodie jouer à un instant t connaissant l'ensemble de l'information musicale ayant précédé en fonction de plusieurs variables :

$$P(M_t|X_{1:t}) \tag{1}$$

où M_t représente la mélodie jouée à l'instant t, et $X_{1:t}$ correspond à un ensemble de variables musicales sur l'ensemble des instants de 1 à t. Ce modèle permet de prendre en considération plusieurs dimensions musicales car les variables musicales définies dans $X_{1:t}$ peuvent être de natures différentes.
Cependant, une telle prédiction n'est pas possible à calculer en pratique car le modèle global généré est d'une dimension trop importante. Pour réduire ce problème, on effectue alors une interpolation de différents sous-modèles P_i, plus simples, ne prenant en compte qu'une partie des variables musicales $A_{i,t} \subset X_{1:t}$, par exemple un modèle de n-gramme sur une dimension ou des modèles d'interactions directes entre dimensions (par exemple, quelle note jouer à l'instant t connaissant l'harmonie à ce même instant ?).
L'interpolation des sous-modèles peut être linéaire [8] :

$$P(M_t|X_{1:t}) = \sum_{i=1}^{I} \lambda_i P_i(M_t|A_{i,t}) \tag{2}$$

avec λ_i des coefficients d'interpolation tels que

$$\sum_{i=1}^{I} \lambda_i = 1 \ , \ \lambda_i \geq 0 \text{ pour tout } i = 1, ..., I$$

et I le nombre de sous-modèles.
L'interpolation peut également être log-linéaire [9] :

$$P(M_t|X_{1:t}) = Z^{-1} \prod_{i=1}^{I} P_i(M_t|A_{i,t})^{\gamma_i} \tag{3}$$

avec γ_i des coefficients d'interpolation tels que $\gamma_i \geq 0$ pour tout $i = 1, ..., I$ et Z un facteur de normalisation dépendant des $A_{i,t}$ défini par :

$$Z = \sum_{M_t} \prod_{i=1}^{I} P_i(M_t|A_{i,t})^{\gamma_i} \tag{4}$$

Il sera également possible par la suite de considérer une modélisation par réseaux de neurones (permettant une généralisation du principe d'interpolation de sous-modèles).
L'apprentissage ne pouvant être exhaustif de par la taille limitée des corpus d'apprentissage et les possibilités d'expressivité infinies de l'improvisation, on effectue un lissage des sous-modèles [5]. Le but du lissage est de corriger les probabilités estimées et d'éviter notamment d'avoir des probabilités d'événements nulles.

Afin de tester l'interpolation de sous-modèles dans le cadre de la génération improvisée de mélodie, nous avons utilisé un corpus de 50 thèmes et improvisations de Charlie Parker issus de l'Omnibook [14]. Ce corpus est alors divisé en trois sous-corpus :

— 40 thèmes et improvisations servent de corpus d'apprentissage pour l'apprentissage des probabilités des différents sous-modèles,
— 5 thèmes et improvisations servent à régler les coefficients d'interpolation et de lissage par minimisation de l'entropie croisée,
— 5 thèmes et improvisations sont réservés pour les tests.

Afin d'évaluer nos résultats nous avons utilisé la notion d'entropie croisée sur les données de test définie par

$$H(C) = -\frac{1}{T} \sum_{t=1}^{T} \log_2 P(C_t|X_{1:t}) \ . \tag{5}$$

Cette métrique peut être interprétée comme le nombre moyen de bits nécessaire pour encoder un élément (au sens du codage de Shannon), et représente l'incompréhension du système. Une faible valeur d'entropie croisée indique alors un meilleur pouvoir de prédiction.
En utilisant cette métrique, nous avons montré qu'utiliser une interpolation de sous-modèles permet au modèle global d'avoir un meilleur pouvoir de prédiction pour la génération de mélodie [7]. Cependant, cette amélioration semble assez faible. Nous avons donc décidé de combiner ces méthodes avec la structure d'oracle des facteurs.

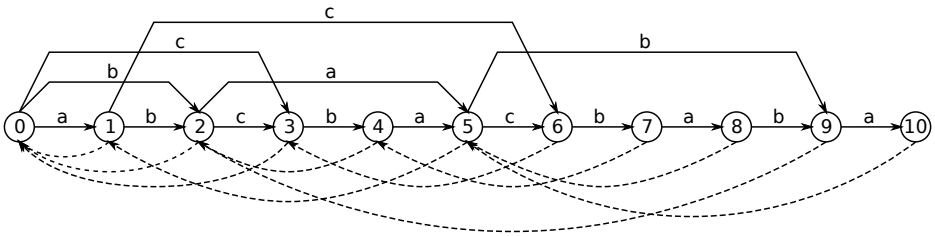

FIGURE 1 – Oracle des facteurs construit à partir du mot $w = abcbacbaba$

3 Combinaison avec l'oracle des facteurs

3.1 Modélisation

L'oracle des facteurs est une structure issue de la bioinformatique et de la théorie des langages [1, 11] ayant déjà fait ses preuves pour l'improvisation musicale dans des systèmes comme OMax [3, 2], ImproTek [13] ou PyOracle [16]. Cette structure permet de conserver la linéarité de ce qui a été appris et de créer des liens entre des zones dans la mémoire possédant un contexte commun. Un exemple d'oracle des facteurs est présenté dans la Figure 1.

Nous avons alors développé un modèle qui combine l'aspect probabiliste des modèles de langage qui prennent en compte les aspects multidimensionnels avec la mise en contexte de l'oracle des facteurs. Nous nous sommes inspirés d'un des éléments d'improvisation proposés par Marilyn Crispell dans [6] :

> *Le développement d'un motif doit être fait de manière logique, organique, ordonné (improvisation en tant que composition spontanée), pas cependant d'une manière préconçue, mais plutôt d'une manière basée sur l'intuition enrichie par des connaissances (de tout ce qui a été étudié, joué, écouté, des différents styles musicaux auxquels on a été exposé, etc. au cours de notre vie - y compris toutes les expériences personnelles) ; le résultat étant un vocabulaire musical personnel.*

D'un côté, on crée un module probabiliste comprenant l'ensemble des différents sous-modèles que l'on souhaite prendre en considération ainsi que les valeurs des coefficients d'interpolation et de lissage nécessaires au modèle probabiliste global. Ce module peut être obtenu à partir d'un apprentissage effectué en différé sur un corpus important, mais peut également être appris (ou mis à jour) en temps réel. Dans la citation de Marilyn Crispell, il correspond à l'ensemble des connaissances acquises au cours de la vie de notre système.
D'un autre côté, on construit un oracle des facteurs dont la construction des

états, des transitions et des liens suffixes ne dépend que d'une dimension choisie (typiquement la mélodie), mais dont les états contiennent l'ensemble de l'information musicale de l'instant qu'ils représentent. Dans la citation de Marilyn Crispell, cela correspond à la logique du contexte dans lequel le motif doit être développé. L'oracle est construit en direct avec des informations fournies par des musiciens ou en différé avec des informations lues dans un corpus (généralement plus restreint que celui du module probabiliste).

À chaque étape de la navigation dans l'oracle, on va alors chercher des informations dans le module probabiliste afin d'orienter notre parcours. À partir d'un état, et connaissant les différents états accessibles (au sens du parcours dans un oracle des facteurs défini dans [2]) et l'information musicale qu'ils contiennent, on calcule pour chaque possibilité un score correspondant à l'interpolation des différents sous-modèles du module probabiliste. On peut alors prendre une décision informée du chemin à prendre, ou alors normaliser les scores des différentes possibilités pour obtenir des probabilités de transition et faire un choix aléatoire suivant les probabilités obtenues.
Notons $\mathrm{Att}(i)$ l'ensemble des états atteignables depuis l'état i au sens de la navigation dans l'oracle des facteurs défini dans [2]. Notons μ_i le contenu musical de l'état i, c'est à dire l'ensemble des variables musicales stockées dans l'état i de l'oracle. Si l'on souhaite générer une mélodie, on a alors pour tout $j \in \mathrm{Att}(i)$, la probabilité de transition dans l'oracle de l'état i à l'état j connaissant le contexte passé définie par :

$$P(i \to j | X_{1:t}) = \frac{P(M_t(\mu_j)|X_{1:t})}{\sum\limits_{k \in \mathrm{Att}(i)} P(M_t(\mu_k)|X_{1:t})} \tag{6}$$

où $M_t(\mu_i)$ est la mélodie dans le contenu musical de l'état i. En pratique, pour $X_{1:t}$ on utilise le contenu musical des états précédents et courants dans le parcours de l'oracle. La Figure 2 illustre le processus.

3.2 Expérience à effectuer

Afin d'évaluer une telle méthode, des tests d'écoute effectués par des auditeurs experts et des improvisateurs professionnels sont nécessaires. L'idée serait ici de générer différentes improvisations à partir d'un oracle construit sur la même pièce musicale :

- des improvisations de type OMax sans aucune utilisation de module probabiliste,
- des improvisations avec un module probabiliste entrainé sur un corpus similaire à la pièce musicale sur laquel est construit l'oracle,
- des improvisations avec un module probabiliste entrainé sur un corpus plus varié.

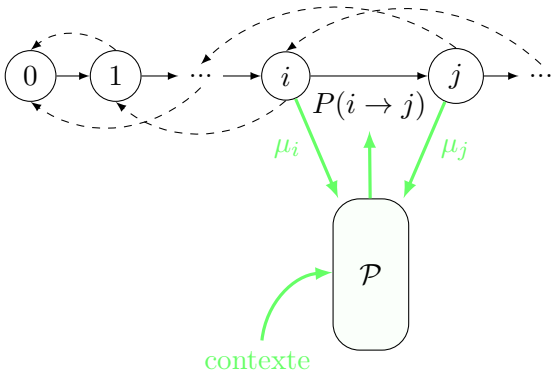

FIGURE 2 – Utilisation d'un modèle probabiliste de language multidimensionnel $\mathcal{P}$ par un oracle des facteurs.

Afin de faire apparaître les différentes dimensions musicales, l'ensemble des variables musicales considérées sont incluses dans l'oracle, même dans le cas des improvisations de type OMax. Seule la construction de l'oracle ne suit que la dimension mélodique.
L'objectif ici est de vérifier si l'utilisation d'un module probabiliste correctement entrainé permet un meilleur guidage de l'improvisation.

4 Communication entre oracles par *belief propagation*

Dans cette section, nous présentons un modèle permettant de faire communiquer plusieurs oracles des facteurs correspondant chacun à une dimension musicale différente ou à un musicien différent. L'idée est de s'approcher des principes des systèmes multi-agents plus représentatifs d'une situation d'improvisation collective réelle. Nous présentons tout d'abord les outils théoriques permettant de mettre en place l'algorithme de propagation de croyance. Puis nous présentons une méthode pour utiliser ces outils dans le cas de l'improvisation musicale multidimensionnelle.

4.1 Graphe de clusters et propagation de croyance

4.1.1 Graphe de clusters

Soit D un ensemble de variables aléatoires. On définit un facteur ϕ sur D comme une fonction du domaine de définition de D vers $\mathbb{R}$. L'ensemble D est appelé la portée du facteur et est noté Scope$[\phi]$. À noter que la notion de facteur englobe à la fois les probabilités jointes et les probabilités conditionnelles.

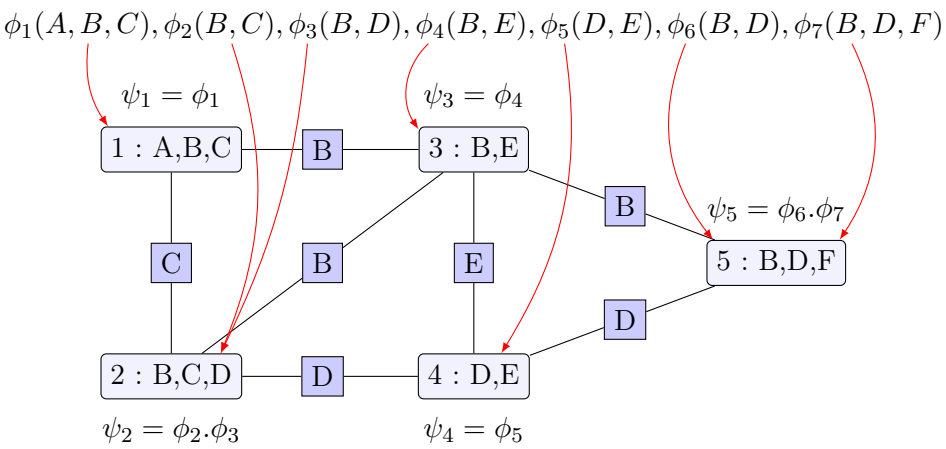

FIGURE 3 – Exemple de graphe de clusters.

Un graphe de clusters $\mathcal{U}$ pour un ensemble de facteurs Φ sur un ensemble de variables X est un graphe non orienté pour lequel à chaque nœud i est associé un sous-ensemble de variables $C_i \subseteq X$ appelé cluster. À chaque arête entre une paire de clusters C_i et C_j est associé un *sepset* $S_{i,j} \subseteq C_i \cap C_j$ [10].

Soit l'ensemble de facteurs $\Phi = \{\phi_1, ..., \phi_k\}$. On attribue chaque ϕ_k à un cluster $C_{\alpha(k)}$ tel que la portée de ϕ_k vérifie $\text{Scope}[\phi_k] \subseteq C_{\alpha(k)}$. Et on définit alors le potentiel initial du cluster C_i par

$$\psi(C_i) = \prod_{k;\alpha(k)=i} \phi_k \ .$$

La Figure 3 est un exemple de graphe de clusters avec une répartition possible des ϕ_k parmi les clusters.

Le graphe de clusters doit respecter deux propriétés :

Family Preservation : pour chaque facteur $\phi_k \in \Phi$, il existe un cluster C_i tel que $\text{Scope}[\phi_k] \subseteq C_i$. De cette manière, on s'assure que l'on peut assigner tous les facteurs à un cluster et donc que l'ensemble de l'information que l'on souhaite représenter est incluses dans le graphe de clusters.

Running Intersection Property : pour chaque paire de clusters C_i,C_j et une variable $A \in C_i \cap C_j$, il existe un unique chemin entre C_i et C_j sur lequel tous les clusters et *sepsets* contiennent A. Cela équivaut à vérifier que, pour chaque variable A, l'ensemble des clusters et *sepsets* contenant A forme un arbre. Cette propriété permet de vérifier deux choses. Tout d'abord, l'existance d'un chemin permet à l'informa-

tion sur une variable A de circuler entre tous les clusters contenant A. Deuxièmement, l'unicité de ce chemin empêche l'information de tourner en boucle et donc de faire exploser certaines croyances sans fondement.

L'exemple de la figure 3 respecte ces propriétés.

4.1.2 Algorithme de *Belief Propagation*

L'algorithme de propagation de croyance est un algorithme basé sur le passage de messages entre clusters. Le message du cluster i transmis au cluster j sur les variables du *sepset* $S_{i,j}$ est noté $\delta_{i\to j}(S_{i,j})$ et est défini par :

$$\delta_{i\to j}(S_{i,j}) = \sum_{C_i - S_{i,j}} \psi_i \prod_{k\in(N_i-\{j\})} \delta_{k\to i} \tag{7}$$

où N_i est l'ensemble des voisins de i.
Dans l'exemple de la figure 3, on a entre les clusters 1 et 3 :

$$\delta_{1\to 3}(B) = \sum_{A,C} \psi_1(A,B,C)\delta_{2\to 1}(C)$$

$$\delta_{3\to 1}(B) = \sum_{E} \psi_3(B,E)\delta_{2\to 3}(B)\delta_{4\to 3}(E)\delta_{5\to 3}(B)$$

On peut noter que le message $\delta_{i\to j}(S_{i,j})$ ne dépend pas de $\delta_{j\to i}(S_{i,j})$. Cela permet d'éviter de répéter à un cluster l'information que l'on vient de recevoir de ce même cluster, et donc de ne pas renforcer une croyance sans fondement.

L'algorithme de propagation de croyance se déroule en plusieurs étapes :

1. on assigne chaque facteur $\phi_k \in \Phi$ à un cluster $C_{\alpha(k)}$,
2. on définit alors les potentiels initiaux $\psi_i(C_i) = \prod_{k:\alpha(k)=i} \phi_k$,
3. on initialise l'ensemble des messages à 1,
4. on répète la mise à jour des messages en suivant la formule (7).
5. on calcule les croyances finales par :

$$\beta_i(C_i) = \psi_i \prod_{k\in N_i} \delta_{k\to i} \tag{8}$$

Pour un cluster, la croyance obtenue est un nouveau facteur correspondant à la mise à jour par inférence des connaissances des autres clusters de son potentiel initial.

4.1.3 Propriétés de convergence de l'algorithme de *Belief Propagation* sur des graphes de clusters

La convergence de l'algorithme de propagation de croyance n'est pas garantie pour une graphe de clusters quelconque. La seul cas particulier de graphes de clusters pour lequel l'algorithme de propagation de croyance garantie une convergence et réalise une inférence exacte est le cas des arbres de clusters. On peut cependant utiliser la notion de calibration pour se donner une idée sur la convergence de l'algorithme. Un graphe de clusters est calibré si chaque pair de clusters adjacent C_i,C_j tombe d'accord sur leur *sepset* $S_{i,j}$, c'est à dire que

$$\sum_{C_i - S_{i,j}} \beta_i(C_i) = \sum_{C_j - S_{i,j}} \beta_j(C_j) \tag{9}$$

Si l'algorithme de propagation de croyance converge, alors le graphe est calibré. De cette manière, le graphe doit être calibré si on souhaite avoir une chance pour que l'algorithme est convergé.

L'ordre dans lequel des messages sont passés entre les clusters influe également sur la convergence de l'algorithme. Le pire cas étant le cas de passage de message synchrone où tous les messages du graphes sont mis à jour en même temps. Cependant, il existe des heuristiques afin d'optimiser les chances de convergence de l'algorithme.

Bien que la convergence ne soit pas garantie en théorie, l'algorithme de propagation de croyance s'est montré très efficace en pratique [10].

4.2 Modélisation

L'objectif ici est d'utiliser les méthodes de propagation de croyance pour faire de l'improvisation multidimensionnelle, et faire communiquer plusieurs oracles ensemble. Une possibilité est de créer différents oracles correspondant à différentes dimensions musicales que l'on souhaite représenter. Le parcours de ces oracles serait alors dirigé à la fois par le module probabiliste qui fournit les potentiels initiaux de transitions séparément pour chaque oracle, mais aussi par la communication entre les oracles par passage de messages. Ainsi, les différents oracles effectuent un choix global (de connaissance interne et externe) d'un parcours à effectuer.

La Figure 4 est un exemple de graphe de clusters pouvant s'appliquer au cas étudié dans [7] où l'on prend en considération des modèles de n-grammes sur les dimensions mélodiques et harmoniques, et des modèles représentant les relations directes entre mélodie et harmonie. Deux oracles sont créés, l'oracle M pour le choix de la mélodie et l'oracle C pour le choix de l'harmonie. Pour chaque oracle, on a créé deux clusters. Un premier pour l'aspect temporel de la dimension représentée, et un deuxième pour les relations directes entre dimensions.

Il est possible d'étendre ce type de représentation pour un nombre de di-

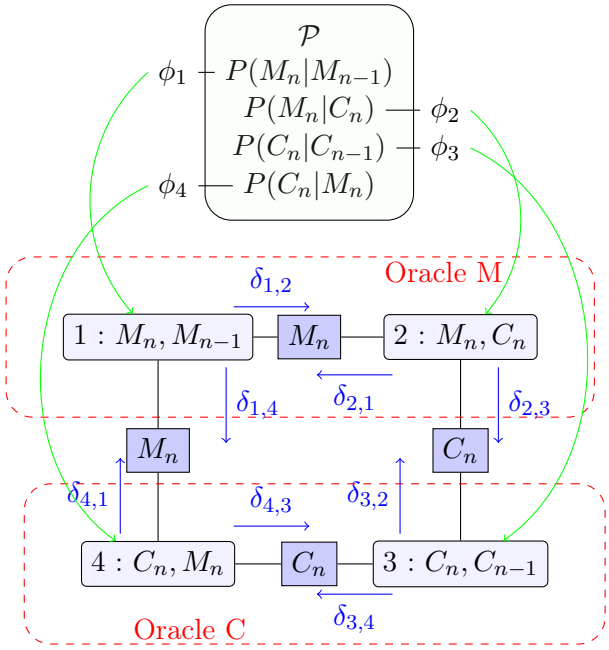

FIGURE 4 – Utilisation de graphes de clusters avec l'oracle des facteurs.

mensions ou un nombre de sous-modèles plus grand. Il faut cependant être attentif à ce que le graphe de clusters créé respecte bien les propriétés de *family preservation* et de *running intersection*. À noter également qu'il est tout à fait possible de considérer l'utilisation de plusieurs modules probabilistes (jusqu'à un par oracle), pouvant être chacun appris sur un musicien différent.

4.3 Expérience à effectuer

Encore une fois, afin d'évaluer une telle méthode, des tests d'écoute effectués par des auditeurs experts et des improvisateurs professionnels sont nécessaires. L'idée serait ici de générer différentes improvisations multidimensionnelles :

- des improvisations multidimensionnelles de type OMax générées à partir d'un oracle des facteurs unique,
- des improvisations multidimensionnelles de type OMax générées avec un oracle par dimension mais sans communication entre les oracles,
- des improvisations multidimensionnelles générées avec un oracle par dimension et un module probabiliste mais sans communication entre les oracles,
- des improvisations multidimensionnelles générées avec un oracle par dimension et un module probabiliste avec communication entre les

oracles avec l'algorithme de propagation de croyances.
L'objectif ici est de vérifier si la communication entre plusieurs oracles par passage de messages apporte un réalisme aux improvisations générées et permet une meilleure cohérence entre les différentes dimensions lors de la génération.

Références

[1] Cyril Allauzen, Maxime Crochemore, and Mathieu Raffinot. Factor oracle : A new structure for pattern matching. *SOFSEM'99, Theory and Practice of Informatics*, pages 291–306, 1999.

[2] Gérard Assayag and Georges Bloch. Navigating the oracle : A heuristic approach. In *Proceedings of the International Computer Music Conference*, pages 405–412, 2007.

[3] Gérard Assayag and Shlomo Dubnov. Using factor oracles for machine improvisation. *Soft Computing*, 8-9 :604–610, 2004.

[4] Laurent Bonasse-Gahot. An update on the SoMax project. Technical report, IRCAM, 2014.

[5] Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University, 1998.

[6] Marilyn Crispell. Elements of improvisation. In John Zorn, editor, *Arcana : Musicians on Music*, pages 190–192. 2000.

[7] Ken Deguernel. Apprentissage de structures multi-dimensionnelles pour l'improvisation musicale. Master's thesis, Inria Nancy Grand Est, 2015.

[8] Frederick Jelinek and Robert L. Mercer. Interpolated estimation of Markov source parameters from sparse data. In *Pattern Recognition in Practice*, pages 381–397, 1980.

[9] Dietrich Klakow. Log-linear interpolation of language models. In *Proceedings of the 5th International Conference on Spoken Language Processing*, pages 1695–1698, 1998.

[10] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models Principles and Techniques*. MIT Press, 2009.

[11] Arnaud Lefebvre and Thierry Lecroq. Computing repeated factors with a factor oracle. In *Proceedings of the 11th Australasian Workshop On Combinatorial Algorithms*, pages 145–158, 2000.

[12] Jérôme Nika and Marc Chemillier. Improtek, integrating harmonic controls into improvisation in the filiation of OMax. In *Proceedings of the International Computer Music Conference*, pages 180–187, 2012.

[13] Jérôme Nika, José Echeveste, Marc Chemillier, and Jean-Louis Giavitto. Planning human-computer improvisation. In *Proceedings of the International Computer Music Conference*, pages 330–338, 2014.

[14] Charlie Parker and Jamey Aebersold. *Charlie Parker Omnibook*. Alfred Music Publishing, 1978.

[15] Stanisław A. Raczynski, Satoru Fukayama, and Emmanuel Vincent. Melody harmonisation with interpolated probabilistic models. *Journal of New Music Research*, 42(3) :223–235, 2013.

[16] Greg Surges and Shlomo Dubnov. Feature selection and composition using pyoracle. In *Proceedings of the 2nd International Workshop on Musical Metacreation*, 2013.