Utilisation de la programmation spatiale pour l'analyse et la représentation symbolique musicale

Mémoire de stage de Master 2 ATIAM année 2009-2010

Stage effectué à l'IRCAM

(Institut de Recherche et Coordination Acoustique/Musique)

dans l'équipe Représentations musicales

Du 01 Mars au 31 Aout 2010

Encadré par

Moreno Andreatta (IRCAM/CNRS)
Jean-Louis Giavitto(IBISC/CNRS)
Antoine Spicher(LACL)
Olivier Michel(LACL)
Carlos Agon(IRCAM)

Louis Bigo

Remerciements

Je tiens tout d'abord à remercier l'équipe Représentations musicales de l'IRCAM pour son accueil. Ce stage s'est déroulé dans un contexte agréable et riche en échanges.

Merci à Antoine Spicher et Olivier Michel pour le temps et la patience qu'ils ont consacré à m'initier aux concepts de la programmation spatiale.

Je remercie évidemment mes encadrants Moreno Andreatta et Jean-Louis Giavitto, mais aussi Carlos Agon pour leurs conseils et leur disponibilité.

De nombreuses réunions et discussions ont été menées au cours de ce stage avec ces cinq personnes. Je les remercie chaleureusement pour la confiance qu'ils m'ont accordé ainsi que l'ouverture d'esprit qu'ils m'ont apporté.

Enfin merci aux étudiants de Master ATIAM pour cette année riche et intense.

Table des matières

Chapitre 1

Introduction

1.1 Contexte

Les travaux de recherche effectués au cours de ce stage constituent le prolongement de reflexions menées lors de la rencontre de deux communautés à l'occasion du séminaire MaMux (Mathématiques, musique et relations avec d'autres disciplines) en Novembre 2009 à l'Ircam [mam09]. L'enjeu de ce projet consiste à concilier programmation spatiale et représentations musicales qui sont les thématiques de ces deux communautés.

Le stage s'est déroulé à l'Ircam (Institut de Recherche et Coordination/Acoustique Musique) au sein de l'équipe Représentations musicales, en collaboration avec le LACL (Laboratoire d'Algorithmique Complexité et Logique). Les travaux ont été encadrés par Moreno Andreatta (IRCAM/CNRS), Jean-Louis Giavitto(IBISC/CNRS), Antoine Spicher(LACL), Olivier Michel (LACL) et Carlos Agon (IRCAM).

L'équipe *Représentations musicales* de l'Ircam mène des réflexions sur des représentations de haut niveau de concepts et structures musicales, appuyées sur des langages informatiques originaux. Ces recherches débouchent sur l'implantation de modèles qui peuvent se tourner vers la création comme vers l'analyse musicale. La diffusion des travaux de l'équipe au sein d'une communauté importante de musiciens a contribué à l'émergence de formes de pensée originales, dues à l'utilisation caractéristique de supports informatiques particuliers. L'une des principales ambitions de ce stage est d'initier l'intégration de la programmation spatiale à ces supports.

La programmation spatiale vise à intégrer la notion d'espace dans le calcul. Le projet MGS poursuit deux objectifs complémentaires : l'étude et le développement de l'apport de notions de nature topologique dans les langages de programmation d'une part, et l'application de ces notions à la conception de nouvelles structures de données et de contôle à la fois expressives et efficaces pour la simulation de systèmes dynamiques à structure dynamique d'autre part. Ces études se concrétisent par le développement d'un langage de programmation expérimental et de son application à la modélisation et à la simulation de systèmes dynamiques, en particulier dans le domaine de la biologie et de la morphogénèse. Ce langage est également nommé MGS.

1.2 Présentation du sujet

L'analyse musicale constitue un exercice se consacrant à l'étude de pièces musicales. Cette discipline a la particularité de pouvoir être abordée sous un grand nombre d'angles et de manières très diverses.

Plusieures notions spatiales issues de la géométrie, de la topologie et de la théorie des groupes ont montré leur utilité dans le domaine de l'analyse musicale [And03], [RM06], [Maz07], [Choa], [Wil]. L'objectif de ce stage est de partir des approches déjà proposées et de les étendre dans le cadre général de la programmation spatiale, plus particulièrement du projet MGS.

Le langage MGS permet la manipulation et la transformation de données organisées dans l'espace. Il s'agit alors de réfléchir à des représentations pertinentes d'objets musicaux qui nous permettront de les considérer d'un point de vue spatial. Une problématique apparaît alors : Dans quelles mesures l'analyse et la synthèse musicale peuvent-elles s'appuyer sur des outils topologiques ? Ce rapport n'a pas la prétention de répondre à cette question délicate. Toutefois on présentera sous forme exploratoire plusieurs points de vue résultant du calcul spatial dans différents cadres musicaux.

Nous présenterons dans un premier temps les aspects généraux de la programmation spatiale, et plus particulièrement les principales notions du projet MGS qui sont intervenus au cours des travaux de ce stage (chapitre 2). Nous proposons ensuite trois applications de ces outils spatiaux pour différents problèmes appartenant à l'analyse musicale : le calcul des séries tous intervalle, la représentation néoriemannienne et l'analyse de suites d'accords (chapitre 3). Enfin, nous exposerons le développement d'une chaine de traitement proposant des représentations symboliques dites simpliciales pour l'analyse de séquences mélodiques (chapitre 4).

Chapitre 2

La programmation spatiale et le langage MGS

Le langage MGS est un langage fonctionnel généraliste [GM01b], [Gia03] dont la particularité est d'offrir un point de vue topologique sur les structures de données manipulées. Ce langage a l'originalité de proposer la manipulation de données sous formes de *collections topologiques* ainsi que leur évolution sous la forme de *transformations*. Les *collections topologiques* proposent un support uniforme pour la présentation d'agrégats de données et les *transformations* fournissent un cadre unifié pour la manipulation par réécriture de ces collections, à travers la définition par cas de fonctions.

Nous proposons dans ce chapitre d'exposer ces deux principes fondamentaux autour desquels s'articulent les travaux déjà réalisés dans le cadre du projet MGS : les collections topologiques et les transformations.

2.1 Les collections topologiques

2.1.1 Présentation des collections topologiques

L'unique source d'organisation des données qu'autorise MGS est le concept de *collection topologique* [GM02], offrant au programmeur un point de vue topologique pour traiter ses constructions. On entend par *collection*, un agrégat de données, c'est-à-dire un ensemble de valeurs scalaires ou de (sous-)collections muni d'une organisation précisant comment ces données sont agencées les unes par rapport aux autres. Une collection est donc une *structure de données* au sens algorithmique du terme.

Une *collection topologique* est une collection dont la structure est capturée par une *relation de voisinage* entre les données, c'est-à-dire en fournissant à partir d'un des éléments de la collection, l'ensemble des autres données qui lui sont directement liées.

L'origine de ce concept vient du souhait d'unifier les structures de données standards (telles que les séquences, les (multi-)ensembles, les graphes, les tableaux, les matrices, etc) dans un même cadre. La généricité offerte par les collections topologiques provient des « déplacements usuels » suivis dans la structure de données, c'est-à-dire de l'ordre de parcours de ses éléments. Cet ordre est capturé par une notion de voisinage rapprochant deux éléments contigus dans la structure. Cette relation de voisinage est fondée sur le fait que le choix d'un programmeur pour une structure de données est motivé par l'utilisation algorithmique de l'objet représenté.

2.1.2 Notions de voisinage

Une collection topologique est caractérisée par les relations de voisinage liant les éléments qui la composent. C'est à dire qu'il est possible à partir de tout élément de déterminer ses voisins directs. Nous noterons « , » la relation de voisinage : x , y signifie que l'élément y est un voisin de l'élément x (dans une certaine collection).

La relation de voisinage permet de spécifier une notion de *sous-collection* constituée par certains sous-ensembles d'éléments d'une collection : un sous-ensemble P de positions d'une collection c dénote une sous-collection de c lorsque pour tout couple d'éléments $<p_1, p_2>$ de P, p_1 est en relation avec p_2 par clôture réflexive, transitive et symétrique de la relation de voisinage de c (on dira que P est *connexe*). Cette relation permet aussi de spécifier la notion de *chemin* : un chemin est une suite d'éléments tous distincts $[x_0, x_1, \ldots, x_n]$ tels que x_i , x_{i+1}.

Cette notion fondamentale de voisinage entre les éléments permet de représenter une collection par un *graphe de voisinage*. Les noeuds correspondent aux éléments de la collection, et sont donc décorés par leur valeur. Les arcs sont orientés et correspondent aux relations de voisinages. L'une des motivations du projet MGS consiste à augmenter l'expressivité de cette notion de graphe en offrant la possibilité de décorer les arrêtes, ainsi que d'inclure des éléments de plus haute dimension. Cette idée de généralisation du graphe de voisinage à une structure de dimension supérieure aboutit à la notion de *complexe cellulaire*. Les complexes cellulaires sont des structures mathématiques issus de la *topologie algébrique* [Spi06], [Mun84] permettant la représentation discrète d'espaces topologiques. L'utilisation qui en est faite par MGS est la création d'espaces topologiques arbitrairement complexes (c'est-à-dire avec des voisinage plus ou moins hétérogènes) servant de structures et de relations de voisinage pour les collections topologiques. De plus, le cadre offert par les complexes cellulaires permet une manipulation dynamique de la structure qu'il est possible de modifier pour construire des espaces plus complexes par transformation de relation de voisinage. En autorisant les modifications topologiques, les collections topologiques constituent un outil élégant pour supporter les changements de structures dans le cadre de modélisations de systèmes dynamiques à structure dynamique. La possibilité de manipuler des complexes cellulaires permet notamment la modélisation d'objets physiques et la simulation du comportement évolutif de systèmes biologiques.

Un complexe cellulaire est un ensemble d'objets élémentaires de dimension variable liés les uns aux autres par des relations de voisinage. Un objet de dimension k est appelé une *k-cellule*. Les 0-cellules sont des points, les 1-cellules des courbes, les 2-cellules des surfaces, les 3-cellules des volumes, etc. Une relation de voisinage entre deux cellules est appelée une relation d'incidence. Les relations d'incidence reposent sur la notion de bord. Par exemple, un arc sera bordé par deux points. Plus généralement,une k-cellule sera bordée par des cellules de dimension inférieure à k, et bordera des cellules de dimension supérieure. Les cellules de dimension $k-1$ incidentes à une k-cellule sont appelées ses faces, celles de dimensions $k+1$ ses cofaces. On observe un tel complexe cellulaire dans la figure 2.1. Le bord de la 2-cellule f est constitué par les trois 0-cellules v_1, v_2, v_3 ainsi que pas les trois 1-cellule e_1, e_2, e_3 qui sont elles même bordées par les cellules v_1, v_2, v_3.

On introduit l'opérateur `<` (resp. `>`) correspondant à une relation d'incidence entre une cellule et une de ses faces. Cet opérateur traduit donc la notion de face. L'expression `x < y` (resp. `x > y`) désigne deux cellules `x` et `y` incidentes tel que `x` est d'une dimension inférieure (resp. supérieure) à `y`.

Enfin, on introduit la notion de $<n, p>$-voisinage entre deux n-cellules. Deux n-cellules sont dites $<n, p>$-voisines si elles sont incidentes à une même p-cellule. Dans la figure 2.1, les cellules v_1 et v_2 sont $<0, 1>$-voisines par l'arc e_1.

Un complexe cellulaire définit uniquement la structure d'un espace à la façon d'une matrice dans laquelle aucun élément ne serait défini. À partir de cette structure, on définit une *collection topologique*

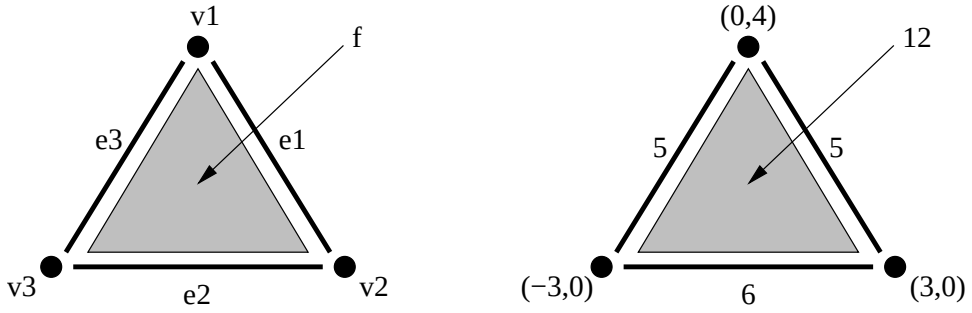


FIG. 2.1 – Sur la gauche, un exemple de complexe cellulaire : il est composé de trois 0-cellules (v_1, v_2, v_3), de trois 1-cellules (e_1, e_2, e_3), et d'une 2-cellule f. Le bord de f est constitué de ses cellules incidentes v_1, v_2, v_3, e_1, e_2 et e_3. Plus particulièrement, les trois arcs sont les faces de f, et par conséquent, f est une coface de e_1, e_2 et e_3. Sur la droite, des données sont associées aux cellules topologiques : des positions pour les sommets, des longueurs pour les arcs et une aire pour f.

comme l'association de données aux cellules topologiques du complexe. La figure 2.1 de droite présente un exemple de collection topologique définie sur le complexe de gauche.

Les chemins permettent de spécifier les sous-collections comme des séquences d'éléments *voisins* deux à deux (nous dirons aussi *contigus* ou *adjacents*). Les éléments constituant les chemins sont de même dimension. Soit n la dimension des cellules d'un chemin ; la relation de voisinage entre deux n-cellules est donnée par le $<n, p>$-voisinage défini précédemment. Ainsi, deux n-cellules voisines consécutives dans un $<n, p>$-chemin ont une p-cellule incidente en commun.

Par exemple, comme il a été dit précédemment, le voisinage entre les éléments des structures de données standards peut être représenté par un graphe de voisinage. Le $<0, 1>$-voisinage permet, sur ce type de structure, de parcourir les sommets en suivant les arcs, construisant ainsi des chemins de sommets dans le graphe de voisinage.

2.1.3 Quelques voisinages

Bien que leur définition soit générique, MGS propose de classer les collections topologiques pour retrouver et manipuler plus facilement des structures de données déjà connues. Ces différentes classes de collections topologiques sont appelées *types*. Les différents types de collections sont distingués suivant les propriétés vérifiées par la relation d'incidence : en agissant sur celle-ci, on peut introduire une forme de régularité ou d'uniformité dans la structure. Les quelques sections suivantes présentent des collections topologiques spécifiques proposées dans l'interprète MGS ainsi que la forme la plus générale de collection, les *chaînes abstraites*. Nous commençons par détailler les concepts liés à la notion de type de collections topologiques.

Généralités sur les types des collections

Chaque type de collection est caractérisé par un nom. Ce nom t est notamment utilisé pour
- définir un prédicat éponyme vérifiant que son argument est une collection du type t
- dénoter une collection topologique vide, `t:()` ;
- spécifier une fonction de conversion `tify` d'une valeur quelconque en une valeur de type t.

Par exemple, le type des séquences MGS est noté `seq` ; les expressions

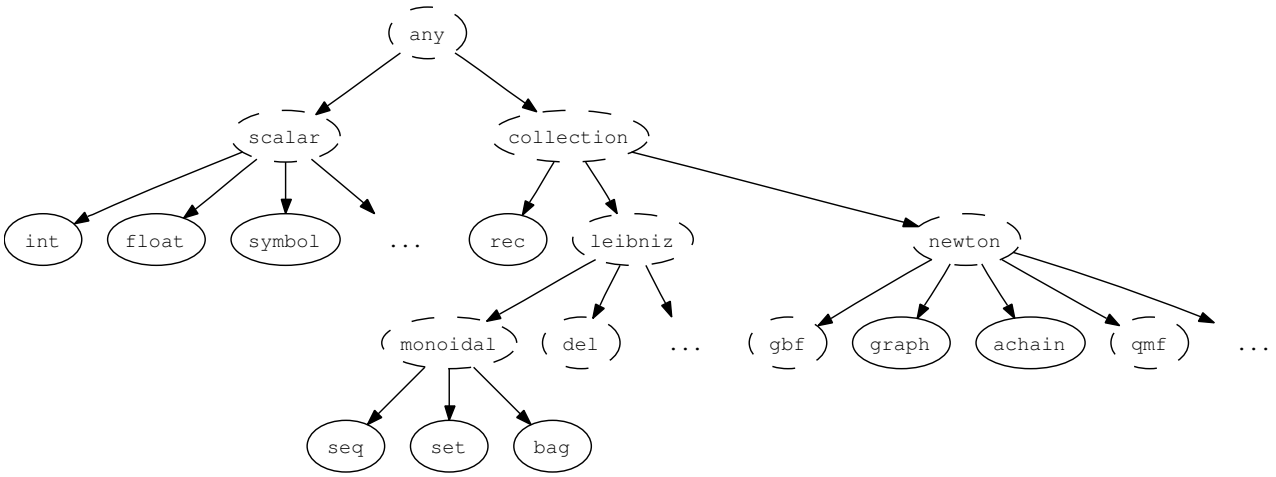


FIG. 2.2 – Hiérarchie des types MGS.

```
seq((1,2,3)) ;;
seq(1) ;;
```

retournent respectivement les valeurs booléennes `true` et `false` (les séquences MGS sont notées entre parenthèses et les éléments sont séparés par des virgules). La séquence vide est notée `seq:()`, et la fonction `seqify` permet de convertir toute valeur MGS (scalaire comme collection topologique) en séquence.

Les types de collection sont organisés hiérarchiquement et cette hiérarchie est étendue à tous les types de valeurs MGS. La figure 2.2 présente cette organisation.

Collection leibnizienne et collection newtonienne. Les mathématiciens Leibniz et Newton avaient chacun une conception de l'espace différente. Pour le premier, l'espace est défini à partir des éléments qu'il contient et de leurs « positions » relatives les uns par rapport aux autres : *l'espace est construit de façon relative aux éléments qu'il contient.* À l'inverse, Newton possède une vision de l'espace comme d'un objet ayant une existence propre sur lequel d'autres objets sont placés et se déplacent.

Cette distinction apparaît dans les collections MGS. Certaines sont construites sur une relation entre les éléments ; leur structure résulte d'un calcul sur les données qui la compose. Elles sont dites *leibniziennes*. Dans ces collections, il n'est pas possible d'associer la valeur spéciale `<undef>` à une position, cette dernière n'existant que si une valeur la décore. La collection vide ne contient aucune position ; celles-ci sont créées au fur et à mesure de l'ajout des éléments. Parmis ces collections, nous distinguons les sequences, les ensembles et les multi-ensembles. La concaténation de deux collections leibniziennes rassemble les éléments des collections et recalcule le voisinage relatif entre les données.

Les collections *newtoniennes* s'opposent aux leibniziennes par la définition *a priori* de leur structure. L'espace engendré est alors décoré par des données. Dans ces collections, les positions existent indépendamment de leur décoration. On associe la valeur spéciale `<undef>` aux positions non décorées. La collection vide contient donc toutes les positions de l'espace décorées par la valeur `<undef>`. Nous présenterons parmi ces collections :

– les *enregistrements*, correspondant aux tables d'association ou dictionnaires,
– les *GBF*, extension de la notion de tableau

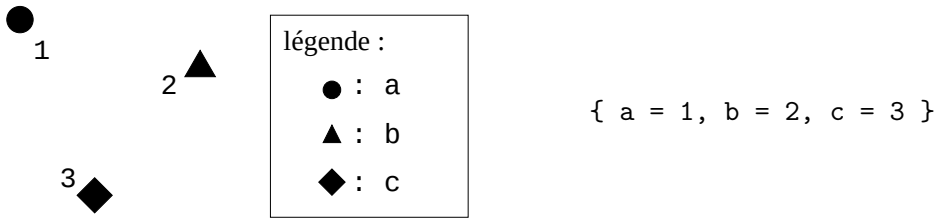


FIG. 2.3 – Exemple d'enregistrement : la topologie d'un enregistrement est un graphe sans arc où chaque sommet représente une clé et les décorations, les valeurs associées.

Les enregistrements

Il s'agit d'un dictionnaire associant une valeur MGS à une clé également appelée *champ* ; bien que les clés puissent être des valeurs MGS quelconques, l'utilisation des symboles est privilégiée. Dans les éléments de syntaxe qui suivent, nous utilisons des identifiants pour nommer les champs.

La structure d'enregistrement est celle d'une table d'association lorsqu'elle est utilisée avec des clés de valeur quelconque. Une utilisation restreinte aux clés de type `symbol` rapproche les enregistrements MGS des enregistrements Pascal, des `struct` en C ou encore des `record` OCaml. Voici la syntaxe réservée à l'utilisation de clés de type `symbol` :

- *Construction* : un enregistrement se construit à l'aide des accolades :

  ```
  { pitch = 60, duration = 500 }
  ```

  est un expression qui construit un enregistrement avec deux champs identifiés par les symboles `'pitch` et `'duration` dont les valeurs associées respectives sont l'entier `60` et l'entier `500`.
- *Accès* : il se fait grâce à la notation pointée :

  ```
  { pitch = (60+12), duration = 500 }.pitch
  ```

  retourne la valeur du champ `'pitch`, c'est-à-dire `72`. L'accès à un champ inexistant renvoie la valeur `<undef>`.
- *Concaténation* : une surcharge de l'opérateur `+` permet de fusionner deux enregistrements pour en calculer un nouveau. Cette concaténation est *asymétrique* avec priorité à droite :

  ```
  { pitch = 60, duration = 500 } + { pitch = 64, onset = 1500 } ;;
  ```

  retourne l'enregistrement `{ pitch = 64, duration = 500, onset = 1500 }`

Relation d'incidence. Les enregistrements sont une sorte de collection topologique très particulière où aucun élément n'a de voisin. Il s'agit en effet d'une structure sans organisation : la topologie correspondante est un complexe cellulaire de dimension 0, c'est-à-dire un graphe de voisinage *sans arc*. Chaque sommet représente une clé et la valeur associée à chaque clé décore le sommet en question. La figure 2.3 schématise une telle topologie. L'espace des positions est toujours défini comme l'ensemble des clés possibles : il s'agit d'une collection newtonienne.

Typage. Le type des enregistrements est `rec` (voir figure 2.2). Il s'agit d'un type concret : `rec:()` dénote la table d'association vide (`<undef>` est associé à chaque clé).

Les séquences

Il s'agit d'une collection monoïdale. Les séquences sont construites par ajouts successifs des données. Cet opérateur d'ajout est *associatif à droite* ; les parenthèses sont donc inutiles :

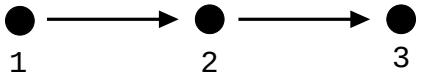


FIG. 2.4 – Exemple de séquence : la topologie d'une séquence est un graphe linéaire où chaque sommet représente un élément et chaque arc la contiguïté.

```
1 : : (2 : : (3 : : seq:())) ;;
```

construit une séquence de 3 éléments. On peut réécrire cette expression de la façon suivante :

```
1 : : 2 : : 3 : : seq:() ;;
1, 2, 3, seq:() ;;
1, 2, 3 ;;
```

La virgule MGS construit par défaut une séquence.

Relation d'incidence. Dans une séquence, chaque élément possède un voisin à droite : il s'agit de la tête de la queue sur laquelle il a été ajouté. La topologie induite par ce voisinage est celle d'un complexe cellulaire de dimension 1 linéaire où les sommets sont décorés par les éléments de la séquence et où un arc, orienté d'un sommet à un autre exprime la contiguïté des éléments dans la séquence. La figure 2.4 schématise une telle topologie. L'espace des positions est construit au fur et à mesure de l'ajout des éléments : il s'agit d'une collection leibnizienne.

Typage. Le type des séquences est `seq` (voir figure 2.2). Il s'agit d'un type concret, `seq:()` dénote la séquence vide.

Les multi-ensembles

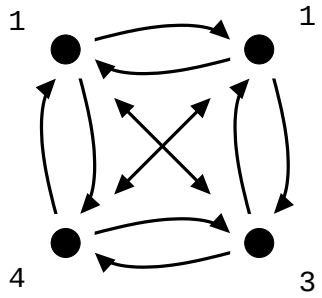
Les *multi-ensembles* sont des ensembles dans lesquels plusieurs occurrences d'un même élément peuvent coexister. Comme les séquences, il s'agit d'une collection monoïdale. L'opérateur d'ajout est *associatif* et *commutatif* ; l'ordre dans lequel les éléments sont ajoutés n'a pas d'importance. Les expressions

```
‘‘do’’, ‘‘sol’’, ‘‘sol’’, ‘‘sol’’, ‘‘la’’, ‘‘la’’, bag:() ;;
‘‘sol’’, ‘‘do’’, ‘‘sol’’, ‘‘la’’, ‘‘la’’, ‘‘sol’’, bag:() ;;
```

calculent le même multi-ensemble contenant deux occurrences de la chaine de caractères `‘‘la’’`, une de `‘‘do’’` et trois de `‘‘sol’’`. La virgule est surchargée pour spécifier l'ajout d'un élément dans un multi-ensemble.

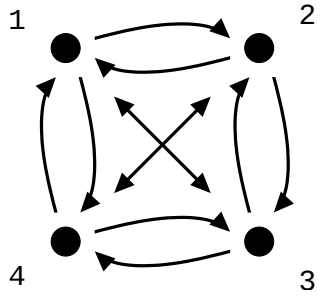
Relation d'incidence. La topologie associée à cette construction est un graphe complet où chaque élément est voisin de tous les autres. Il s'agit d'un complexe cellulaire de dimension 1. La figure 2.5 schématise une telle topologie. L'espace des positions est construit au fur et à mesure de l'ajout des éléments : il s'agit d'une collection leibnizienne.

Typage. Le type des multi-ensembles est `bag` (voir figure 2.2). Il s'agit d'un type concret, `bag:()` dénote le multi-ensemble vide.




```
(1, 1, 2, 3, bag:())
```

FIG. 2.5 – Exemple de multi-ensemble : la topologie d'un multi-ensemble est un graphe complet où chaque sommet représente un élément.




```
(1, 2, 3, 4, set:())
```

FIG. 2.6 – Exemple d'ensemble : la topologie d'un ensemble est un graphe complet où chaque sommet représente un élément.

Les ensembles

Contrairement aux multi-ensembles, il ne peut y avoir plus d'une occurrence d'un même élément. L'opérateur d'ajout est donc *associatif*, *commutatif* et *idempotent*. L'expression

```
``do'', ``do'', ``sol'', ``la'', ``la'', ``la'', set:() ;;
```

évalue un ensemble contenant 3 éléments : ``do'', ``sol'' et ``la''. La virgule est surchargée pour spécifier l'ajout d'un élément dans un ensemble.

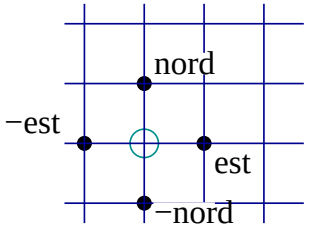
Relation d'incidence. La topologie associée à cette construction est un graphe complet où chaque élément est voisin de tous les autres et n'apparaît qu'une seule fois. Il s'agit d'un complexe cellulaire de dimension 1. La figure 2.6 schématise une telle topologie. L'espace des positions est construit au fur et à mesure de l'ajout des éléments : il s'agit d'une collection leibnizienne.

Typage. Le type des ensembles est `set` (voir figure 2.2). Il s'agit d'un type concret, `set:()` dénote l'ensemble vide.

Les collections GBF

Les *collections GBF (Group based field)* [GM01a] sont des structures de données organisés sur les éléments d'un groupe mathématique. Ce type de collection topologique est utilisée dans le cadre d'une étude portant sur des éléments agencés suivant des voisinages réguliers. Les GBF sont le support de

collections représentant des espaces homogènes. On rencontre de telles structures de données lorsqu'on discrétise de manière régulière une zone de l'espace physique. Nous les utiliserons dans le contexte de la représentation Néoriemannienne (chapitre 3). Les GBF appartiennent au groupe des collections newtoniennes. Il s'agit donc d'un espace généré « sans limite » à partir de relations de voisinages générique. Les divers points de l'espace peuvent alors être décorés par des valeurs. Ces valeurs sont susceptibles de subir des transformations mais l'espace lui, contrairement aux collections leibniziennes, reste invariant. Le maillage carré constitue un GBF dont les voisinages réguliers sont évidents :



L'espace est ici généré par quatre voisinages réguliers opposés deux à deux. Chaque position de l'espace possède une position voisine suivant les quatre directions. Afin de générer un tel espace, on construit une structure de groupe mathématique dont les générateurs sont deux des quatres directions, et dont l'opération est constitué par les déplacements élémentaires suivant ces directions. On remarque les propriétés suivantes quant à l'opération :

- elle est associative,
- pour chaque direction $d \in \mathcal{D}$, il existe un déplacement inverse noté $-d \in \mathcal{D}$,
- et il existe un déplacement nul qui consiste à « ne pas se déplacer ».

L'application des déplacements en un point est l'action du groupe sur l'espace des points. Pour définir un espace homogène arbitraire, il faut donc spécifier deux choses :

1. le groupe des déplacements à partir des déplacements élémentaires $\mathcal{D}$;
2. l'ensemble des points sur lequel le groupe va agir.

Dans l'interprète MGS, le second point est traité en considérant que le groupe des déplacements agit sur lui-même : un élément du groupe correspond alors à un point de l'espace homogène et l'action du groupe correspond simplement à la loi du groupe : $\mathcal{V}oisin(d, p) = p + d$.

Pour spécifier un GBF, on utilise une *présentation* [LS01] : il s'agit d'une liste de générateurs et d'une liste d'équations entre ces générateurs. La syntaxe d'une présentation est la suivante :

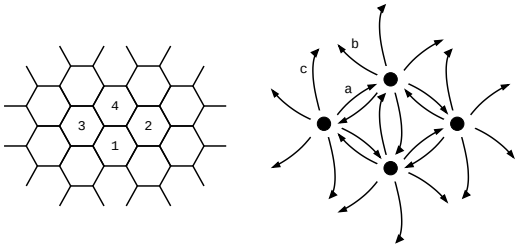
$$\langle g_1, g_2, \ldots, g_n ; e_1, e_2, \ldots, e_m \rangle$$

où $g_1, g_2, \ldots, g_n$ sont les générateurs et $e_1, e_2, \ldots, e_m$ les équations entre générateurs. Tout élément du groupe s'obtient alors comme une somme de générateurs. Ici, ils correspondent aux déplacements élémentaires qui définissent le voisinage homogène.

Dans notre exemple de grille, la présentation du GBF est :

$$\langle nord, est ; nord + est = est + nord \rangle$$

Relation d'incidence. À partir de la présentation d'un groupe, le graphe de Cayley fournit la topologie engendrée par le groupe : les nœuds sont les éléments du groupe et deux éléments sont liés par un arc si leurs positions ne diffèrent que de l'ajout ou du retrait d'un générateur. Il s'agit donc d'un complexe cellulaire de dimension 1. La figure 2.7 schématise une telle topologie. L'espace des positions est toujours déterminé par la présentation du groupe des déplacements indépendante des valeurs : il s'agit d'une collection newtonienne.




```
gbf hexa = < a, b, c ; a + b = c > ;;
(1,2) : :(3,4) : :seq:() following |a>, |b> ;;
```

FIG. 2.7 – Exemple de collection GBF : la topologie d'une collection GBF est donnée par le graphe de Cayley associé à la présentation du groupe des déplacements. Les deux graphes sont deux représentations duales du graphe de Cayley [LS01] de la présentation `hexa`.

Typage. Le type résultant en MGS est un type abstrait appelé `gbf`. Le mot clé du même nom est alors utilisé pour spécifier des groupes particuliers et ainsi générer de nouveaux voisinages uniformes. Les déplacements élémentaires sont commutatifs entre eux. La topologie de la grille carrée définie précédemment est construite en MGS de la façon suivante :

```
gbf grille = < nord, est > ;;
```

crée un nouveau type de GBF nommé `grille` à partir des deux générateurs `nord` et `est`. Ces derniers sont à l'origine de nouvelles valeurs MGS de type `posgbf`. Une *arithmétique* des déplacements est en effet mise en place pour manipuler les positions des GBF de type `grille` ; elle est fondée sur deux nouvelles valeurs `|nord>` et `|est>`, et les opérateurs `+`, `-` et `*`. Par exemple

```
|nord> + |est> == |est> - |nord> + 2*|nord> ;;
```

retourne la valeur booléenne `true`. Ces déplacements expriment également les coordonnées des points, partant d'une origine arbitrairement choisie et permettent d'identifier les positions des éléments du GBF. La valeur `grille:()` correspond alors à la grille générée par le groupe associé aux générateurs `nord` et `est`, où toutes les positions ont une valeur indéfinie. La collection étant vide, la valeur `<undef>` est associée à chaque point du GBF. Afin de spécifier les décorations, MGS fournit l'opérateur `following` :

```
      (‘a, ‘a, ‘a)
  : : (‘b, ‘b)
  : : (‘c, ‘c, ‘c, ‘c)
  : : seq:() following |nord>, |est> ;;
```

complète le GBF `grille:()` à partir de l'origine de l'espace engendré comme le montre la figure 2.8.

Les chaînes abstraites

Cette collection est la plus générale en terme de voisinage arbitraire que l'interprète MGS fournit. Elle correspond à une implantation des complexes cellulaires qui ont été décrits précédemment.

Relation d'incidence. Contrairement aux collections présentées jusqu'ici, les chaînes abstraites ne sont pas limitées à un graphe de voisinage où les sommets sont les éléments et les arcs la relation de voisinage. Elles correspondent à la construction de complexes cellulaires de dimension arbitraire. La figure 2.9 schématise une telle topologie. L'espace des positions est défini seul dans un premier temps.

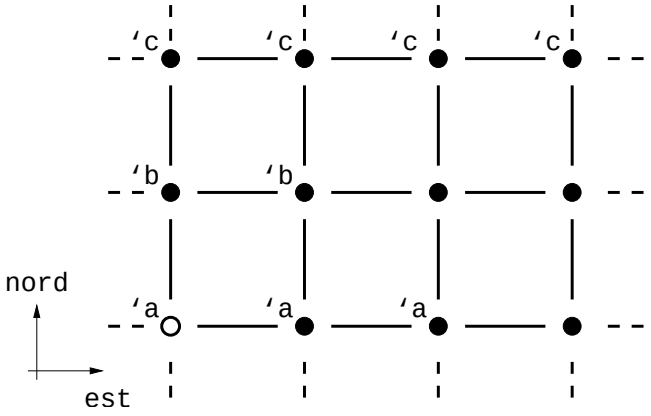
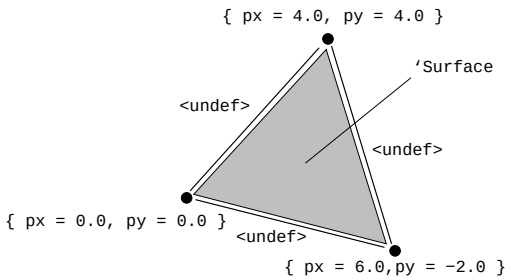


FIG. 2.8 – Décoration dans un GBF à l'aide de l'opérateur `following` (voir le programme décrit dans le texte).

Il est ensuite décoré ce qui amène à considérer les chaînes abstraites comme un type de collections newtoniennes.

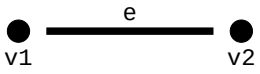
Typage. Un type de données scalaire a été créé pour une nouvelle sorte de valeur appelée `acell` (pour *cellule abstraite*). Une `acell` est une cellule topologique ; c'est à travers ces objets que les complexes cellulaires vont être représentés : un complexe cellulaire n'est pas un objet élémentaire manipulable explicitement ; en revanche, il peut être parcouru à travers ses éléments constituants, les cellules topologiques représentées par les valeurs de type `acell`, en utilisant la relation d'incidence qui les relie. Des cellules topologiques fraîches peuvent être créées à l'aide de la fonction `new_acell`. Celle-ci demande en paramètre la dimension de la cellule créée ainsi que la liste de ses cofaces et de ses faces. Par exemple, pour créer un arc avec ses deux sommets, on écrit :




```
v1 := add_acell(0, seq:(), seq:()) ;;
v2 := add_acell(0, seq:(), seq:()) ;;
v3 := add_acell(0, seq:(), seq:()) ;;
e1 := add_acell(1, (v1,v2), seq:()) ;;
e2 := add_acell(1, (v2,v3), seq:()) ;;
e3 := add_acell(1, (v3,v1), seq:()) ;;
f  := add_acell(2, (e1,e2,e3), seq:()) ;;
  { px = 0.0, py = 0.0 } * v1
+ { px = 4.0, py = 4.0 } * v2
+ { px = 6.0, py =-2.0 } * v3
+ 'Surface * f ;;
```

FIG. 2.9 – Exemple de chaîne abstraite : la topologie est définie de façon explicite par la création de nouvelles cellules ; les décorations sont ensuite placées sur les positions en utilisant les opérateurs de somme et de produit.

```
v1  := new_acell(0, seq:(), seq:()) ;;
v2  := new_acell(0, seq:(), seq:()) ;;
e   := new_acell(1, (v1, v2, seq:()), seq:()) ;;
```



La cellule associée à la variable `e` est une cellule de dimension 1. Afin de conserver un caractère fonctionnel pur, la fonction `new_acell` fait une copie des faces et des cofaces données en paramètre. En effet, définir `v1` et `v2` comme faces de `e` signifie que `e` est une coface de `v1` et `v2`, ce qui va à l'encontre de la définition de ces deux mêmes cellules. De la même façon, si `v1` et `v2` étaient des éléments de complexes cellulaires différents, ceux-ci seraient copiés et unis en une nouvelle entité. Par souci de performance, une fonction équivalente à `new_acell` mais fonctionnant par effets de bord est également disponible ; il s'agit de `add_acell`. L'expression suivante

```
v1'  := add_acell(0, seq:(), seq:()) ;;
v2'  := add_acell(0, seq:(), seq:()) ;;
e'   := add_acell(1, (v1', v2', seq:()), seq:()) ;;
```

produit le même arc que précédemment, excepté que `v1'` et `v2'` sont dans ce cas effectivement les faces de `e'`. Les cellules topologiques associées à `v1'` et `v2'` ne sont pas copiées mais leurs définitions sont mises à jour. Sachant que la fonction `member` teste l'appartenance d'une valeur à une collection, les expressions

```
member(faces(e), v1) ;;
member(faces(e'), v1') ;;
```

retournent respectivement `false` et `true`. Comme le montre l'exemple, des accesseurs sont fournis pour parcourir le complexe cellulaire :

- `faces(`c`)` retourne la liste des faces de c ;
- `cofaces(`c`)` retourne la liste des cofaces de c ;
- `icells(`c`)` retourne la liste des cellules incidentes à c ;
- `pcells(`c`,`p`)` retourne la liste des p-voisines de c.

Une fois créé, l'espace défini par les cellules topologiques abstraites peut être décoré par des valeurs du langage. On obtient alors le nouveau type de collection topologique qui nous intéresse, les `achain` pour *chaînes abstraites*. Les *chaînes topologiques* sont des objets issus de la topologie algébrique. Elles correspondent à des fonctions partielles associant aux cellules d'un complexe cellulaire une valeur. Elles sont souvent représentées par une somme formelle, style d'écriture utilisé également dans l'interprète. En reprenant l'exemple précédent, on écrit :

```
‘vertex * v1 + ‘edge * e ;;
```

pour construire la chaîne topologique associant la valeur symbolique `‘vertex` à la cellule associée à la variable `v1` et la valeur `‘edge` à `e`.

Afin de conserver l'intégrité des chaînes topologiques ainsi créées, il est nécessaire d'interdire la construction de toute nouvelle cellule topologique par la fonction `add_acell` ; l'utilisation de cette fonction est par conséquent prohibée sur les cellules utilisées dans la définition d'une chaîne topologique. Il reste néanmoins possible d'ajouter des cellules à un complexe grâce à la primitive `new_acell` qui crée une copie du support utilisé dans la définition de ce nouvel objet. Les opérations topologiques comme l'insertion, la suppression ou le raffinement(what ?) des cellules topologiques, opérations classiques dans certains modeleurs topologiques, ne sont pas fournies par l'interprète MGS ; l'objectif de celui-ci n'est en effet que de montrer l'utilité des transformations que nous aborderons dans la section suivante, pour programmer ces opérations.

2.2 Les transformations

En MGS, la modification topologique d'une collection se fait par le biais d'une *transformation*. Une transformation est spécifiée par un motif filtrant qui conduit à une évolution locale de la structure considérée. Pour une collection C donnée, une transformation comprend :

1. une sous-collection A de C
2. le calcul d'une nouvelle collection B à partir de A
3. la substitution de B en lieu et place de A dans C

Pour spécifier une transformation (figure 2.10) il faut préciser : comment sélectionner A (à l'aide des motifs), comment est calculé A' (à l'aide de la partie droite des règles) et parfois les contraintes de la substitution de A par A' dans C (en général la manière de substituer les parties dépend de la nature de la topologie de C).

Une transformation s'écrit :

$$\texttt{trans}\ T = \{\ \ldots \quad m \Rightarrow f(m); \quad \ldots\ \}$$

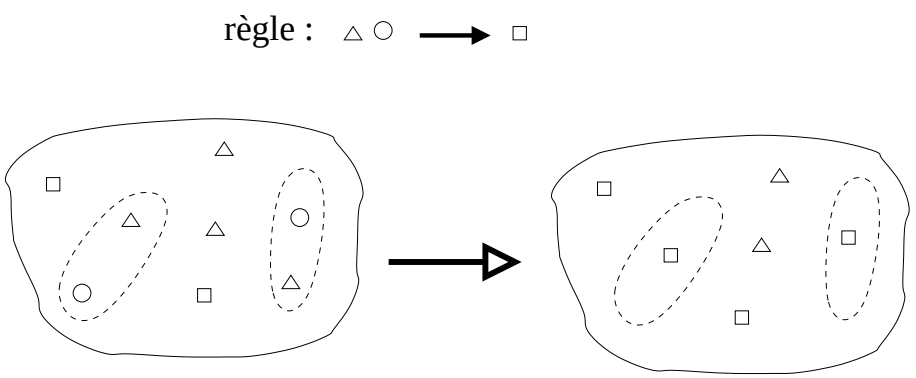


FIG. 2.10 – Transformation d'une collection. Une règle spécifie qu'une sous-collection A doit être remplacée par la sous-collection A' calculée à partir de A. La collection calculée dans la partie droite de la règle dépend de la sous-collection filtrée dans sa partie gauche et éventuellement de son voisinage.

Une règle dans une transformation correspond à une paire `pattern => expression`. Quand une règle est appliquée à une collection topologique, les sous-collections vérifiant le `pattern` sont remplacés par des collections topologiques via l'application de la fonction `expression`. Un *pattern basique* désigne une cellule satisfaisant des contraintes spécifiées par des gardes. Par exemple, l'expression `c / c=3` filtre une cellule décorée par la valeur 3. La garde est un prédicat se placant après le symbole `/`. La variable `c` peut être utilisée dans la garde et partout ailleurs dans la règle pour désigner la valeur de la cellule satisfaisant la condition, ou pour désigner la cellule elle-même, suivant le contexte.

Un pattern est une *composition* de patterns basiques. Il éxiste trois opérateurs de composition :

1. La simple juxtaposition (par exemple "`x y`") ne contraint pas les arguments de la composition.
2. Lorsque deux patterns basiques sont composés à l'aide d'une virgule (par exemple "`x , y`", cela signifie que les cellules désignées par `x` et `y` doivent être p-voisines. La dimension p dépend de la collection topologique et peut être spécifiée explicitement durant l'application de la transformation si besoin. La valeur par défault pour p est 1.

3. Le dernier opérateur de composition correspond à l'opérateur de face : un pattern "`x < y`" (resp. "`x > y`") désigne deux cellules `x` et `y` tel que $x < y$ (resp. $x > y$) (voir sous-section 2.1.2).

On s'est restreint dans ce chapitre à présenter les outils spatiaux utilisés dans les travaux menés au cours de ce stage. Cependant, le langage MGS regroupe un grand nombre de notions supplémentaires [Spi06] qui trouveront probablement leur utilité dans les travaux futurs.

Chapitre 3

Applications illustrant l'approche

La programmation spatiale vise à intégrer l'espace dans le calcul, soit comme contrainte, soit comme ressource, soit encore comme résultat. C'est pourquoi nous proposons dans ce chapitre trois exemples du domaine de l'analyse musicale illustrant chacun de ces aspects.

Dans un premier temps, nous proposons une solutions originale au problème du calcul des séries tous intervalle. Nous expliquerons comment la définition d'un espace pourra constituer une contrainte forcant le calcul vers un certain résultat.

Nous exposons ensuite comment la définition d'un espace sous la forme d'un maillage hexagonal peut servir de support ressource pour l'analyse de progressions harmoniques dans la tradition musicologique néo-riemannienne.

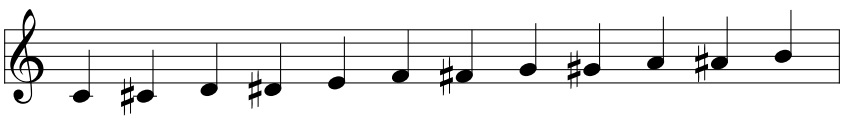
Enfin, nous montrerons que l'étude de séquences d'accords peut aboutir à des résultats sous forme d'espaces topolgiques dont il est possible d'extraire des propriétés spatiales trouvant une cohérence dans le domaine musical.

3.1 Calcul des séries tous intervalles

3.1.1 Les séries tous intervalles

Imaginé par Arnold Schoenberg dans les années 1920, le *dodécaphonisme* a boulversé les méthodes de composition du XXème siècle. Ce courant musical est basé sur l'utilisation de séquences de notes appelées *séries dodécaphoniques*.

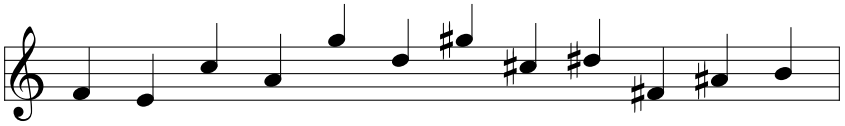
Une série dodécaphonique consiste en une suite de 12 notes incluant une fois seulement chacune des 12 notes de la gamme chromatique. La gamme chromatique elle-même est un exemple trivial de série dodécaphonique :



Le fait de pouvoir utiliser les 12 notes dans n'importe quel ordre laisse au compositeur un grand

nombre de possibilités[1]. La technique dodécaphonique ne distingue pas les notes à une octave près. Pour cette raison il est pratique de se placer dans le cercle chromatique pour les étudier (figure 3.1). Outre la volonté d'exploiter au maximum les classes de hauteurs dans une série dodécaphonique, certains compositeurs se sont interessés aux intervalles qu'elles contiennent. Un intervalle correspond à la distance, en terme de hauteur, entre deux notes consécutives. Il apparait clairement que le seul interval présent dans la séquence chromatique précédente est le demi-ton. Si l'on se place dans le cerle chromatique, il apparait que 11 intervalles de hauteur sont possibles.

Une *série tous intervalles* est une série de 12 notes qui utilise une seule fois chacune des notes de la gamme chromatique et comprend une seule fois chacun des onze intervalles. Alban Berg utilise une telle série dans sa *Suite Lyrique* :



L'énumération des séries tous intervalles est un problème bien connu des théoriciens de la musique[RM06]. En effet, ces séries étant très difficile à construire à la main, leur dénombrement l'est d'autant plus. Le fait de numéroter les notes de 1 à 12 et de caractériser les intervalles par le nombre de demi-tons qu'ils comprennent ramène la recherche des séries tous intervalles à un problème combinatoire. La première approche de ce problème, datant de 1964 [Eim64], a permis de déterminer qu'il éxiste 1928 séries tous intervalles parmis les $12! \simeq 479.10^6$ séries dodécaphoniques possibles. Dans la suite de cette section, on présentera différentes approches de ce problème. On abordera en premier lieu l'approche naïve par force brute consistant à tester la totalité des séries dodécaphoniques. Après avoir optimisé cet algorithme on proposera une résolution du problème plus élégante en se placant dans un espace approprié aux séries tous intervalles.

3.1.2 Approche par force brute

Comme évoqué précédemment, l'énumération des séries tous intervalles représente un problème combinatoire classique mais lourd en calcul. L'approche la plus intuitive consiste à lister la totalité des permutations de douze notes aboutissant à une série dodécaphonique, puis à réaliser sur chacune des solutions un test déterminant si sa structure intervallique fait intervenir les onze intervalles possibles. L'espace support de ce calcul consiste en l'ensemble des douze notes. Cet espace ne contient aucune information concernant les intervalles séparant les notes entre elles. D'un point de vue spatial, il s'agit d'une collection topologique correspondant à un graphe complet. Chaque élément de l'ensemble est associé à un sommet du graphe, décoré par une des douze notes. Il éxiste une arrête entre chaque paire de sommets comprise dans le graphe, ce qui montre qu'il n'existe aucun ordonnancement entre les éléments du graphe (figure 3.1).

Dans cet espace, une série dodécaphonique correspond à un chemin hamiltonien dans lequel chaque note n'apparait qu'une seule fois. Ces chemins correspondent aux $< 0, 1 >$-chemins de longueur maximale. En MGS, on spécifiera la recherche de ces chemins de la manière suivante :

```
n0, n2, ..., n11 / ais(n0,...,n11)
```

[1] A priori ce nombre est égale à $12! \simeq 479.10^6$. En réalité, en introduisant une relation d'équivalence entre deux séries dodécaphoniques étant l'une la transformée de l'autre via une transposition, une inversion et/ou une rétrogradation on peut montrer que le nombre de classes d'équivalence de séries est égale à 9985920 [Rei85]

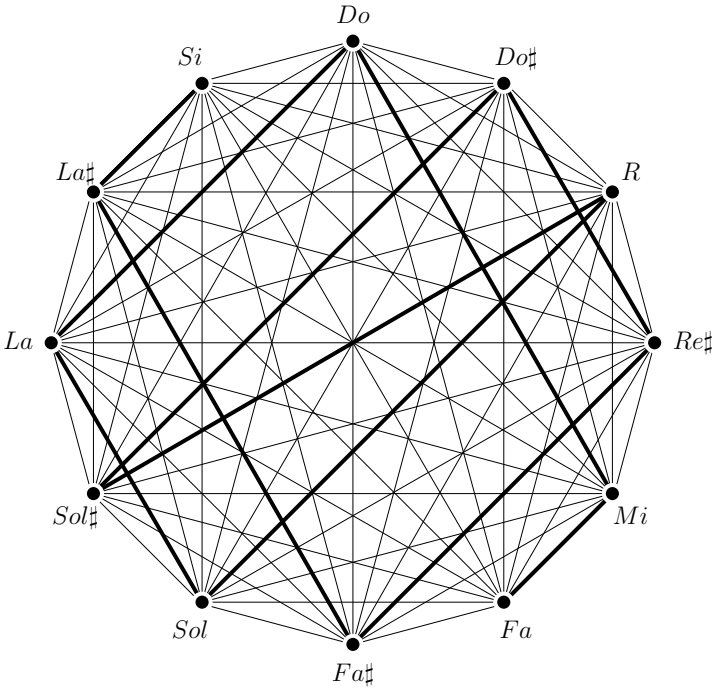


FIG. 3.1 – Représentation spatiale de la série tous intervalles de Alban Berg dans le cercle chromatique

Cette notation interprête la virgule comme la relation de voisinage entre les éléments qu'elle sépare. Ici, cette relation correspond au <0,1>-voisinage. La contrainte `ais(n0,...,n11)` est une fonction vérifiant que la structure intervallique de la série comprend les onze intervales. Cette instruction spécifie donc bien les solutions de notre problème. L'énumération des 1928 séries tous intervalles via ce procédé se fait en à peu près 15 minutes [2]. Toutefois, il est possible d'optimiser cette démarche, du point de vue de la spécification des solutions mais aussi de celui du support spatiale. Dans la suite, on se penchera sur ces deux aspects.

3.1.3 Optimisation de la recherche de solutions

Il apparait clairement dans l'approche précédente qu'une grande partie des calculs effectués est inutile. En effet, dans le cadre de notre recherche de séries tous intervalles, il n'est pas nécéssaire de construire une série dodécaphonique jusqu'à sa dernière note si un intervalle a déjà été répété une fois. Dans la plupart des cas en effet, les premières de ses douze notes nous permettent d'affirmer qu'une série n'est pas tous intervalles. Par exemple, si l'intervalle entre `n0` et `n1` est le même qu'entre `n1` et `n2`, il n'est alors pas utile d'aller chercher toute les combinaisons possibles parmis les notes suivantes. En d'autres termes, le prédicat `ais` pourrait être appliqué de manière *distribué* au cours de la recherche des séries tous intervalles plutôt qu'en aval des permutations de douze notes uniquement. Supposons dans notre graphe complet (figure 3.1) que les arêtes soient décorés par les intervalles séparant les notes associées aux sommets qui les bordent. Par exemple, l'arrête séparant les sommets *mi* et *sol* serait décorée par la valeur 3 car 3 demi-tons séparent les notes *mi* et *sol*. Cette vision de l'agencement des notes entre elles nous permet de tenir compte de la structure intervallique d'une série au cours de

[2] Cette valeur dépend évidemment de différents paramètres, comme la machine utilisée, l'écriture de l'algorithme etc. Cependant, c'est ici l'ordre de grandeur qui nous intéresse afin de pouvoir le comparer avec les différentes optimisations qui suivent. Les tests suivants ont donc été réalisés dans les mêmes conditions.

sa construction. L'instruction MGS s'écrira de cette manière :

```
n0 < i1  > n1
   < i2  > n2  / (i2 !=i1)
   < i3  > n3  / (i3 !=i1) / (i3 !=i2)
   ...
   < i11 > n11 / (i11 !=i1) / ... / (i11 !=i10)
```

Cette instruction est relativement proche de la précédente, mais on remarquera cette fois l'utilisation des relations de voisinages $< i_p >$ plutôt que de la virgule. Rappelons que les opérateurs $<$ et $>$ expriment la relation d'incidence entre les termes situés à gauche et à droite de l'expression. De cette manière, on peut nommer chacun des <0,1>-voisinages contenus dans l'espace des notes. Ainsi, `i`p correspond à l'intervalle séprant les notes `n`$(p-1)$ et `n`p. Une série tous intervalles correspond alors à un chemin dans lequel les `i`p sont tous distincts. Cette propriété est vérifiée par les gardes `(i`p `!=i`q`)` qui sont distribuées tout au long de la constitution de la série. La détermination des séries tous intervalles est ramené à 30 secondes grâce à cette optimisation.

3.1.4 Optimisation de l'espace support

Comme on a pu l'observer dans les deux propositions précédentes, il existe deux types de contraintes : les *contraintes logiques* et les *contraintes spatiales*. L'optimisation que l'on vient de voir fait intervenir les deux types de contraintes. Les contraintes spatiales spécifient les zones de l'espace potentielles pour la détermination de séries tous intervalles, sans prendre en compte les décorations des éléments de l'espace. D'un point de vue spatial, les deux solutions précédentes sont équivalentes. L'optimisation réalisée en second lieu ne permet pas une réduction des permutations candidates, mais une réduction de 479.10^6 à 9.10^6 du nombre de séries visitées par l'algorithme en empêchant la construction totale des séries qui s'avèrent fausse dès le début. D'un point de vue spatial, il est plus intéressant de déterminer les solutions par le biais de contraintes spatiales plutôt que de contraintes logiques. On cherche donc à spécifier les solutions de manière spatiale uniquement.

Dans notre précédente instruction, le grand nombre de séries candidates était dû au fait que les valeurs des intervalles n'avaient pas de sens d'un point de vue spatiale. En effet, chaque intervalle était représenté plusieures fois. Par exemple, on considérait comme deux éléments de l'espace distincts les demi-tons séparant *do* et *do*$\sharp$ et fa et $fa\sharp$. Une optimisation spatiale pourrait réunir les intervalles de même valeur en un unique élément de l'espace. Suivant cette idée, on propose d'étendre le graphe précédemment évoqué à un complexe cellulaire de dimension 2 dans lequel :

- les sommets (0-cellules) représentent les 12 notes
- les arcs (1-cellules) représentent les intervalles absolus entre toutes les paires de notes possibles
- les surfaces (2-cellules) représentent les 11 intervalles relatifs

Il n'est pas possible de se représenter le complexe cellulaire dans sa totalité, comprenant 12 sommets, 66 arrêtes, et 11 surfaces. Une *surface intervallique* (2-cellule) est bordée par les arcs correspondant à toutes les paires de note constituant l'intervalle en question. Ces surfaces sont représentables une à une pour chaque intervalle (figure x) mais il est intéressant de noter qu'elles ont des topologies très différentes en fonction du caractère générateur de l'intervalle.

Le demi-ton et la quarte sont des intervales générateurs de toute la gamme, c'est pourquoi les surfaces intervalliques qui leurs sont associés ne possèdent qu'un seul bord et n'ont donc aucun trous. Ce n'est pas le cas de la tierce mineure et de la tierce majeure qui bouclent à l'intérieur de la gamme, ne permettant de générer qu'une partie des douze notes. En fait, on réalise qu'une surface intervallique ne possèdera qu'un seul bord si et seulement si elle correspond à un nombre de demi-tons premier avec

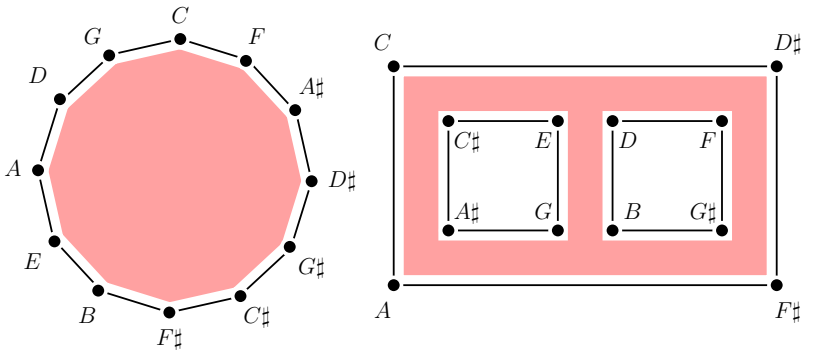


FIG. 3.2 – Représentation spatiale de l'intervalle de quarte (à gauche) et de tierce mineure (à droite)

12. Même si il n'est pas possible de se représenter le complexe cellulaire dans sa totalité, on comprend la manière dont il est agencé en observant la figure 3.3 représentant l'enchainement des cinq premières notes de la série de Berg.

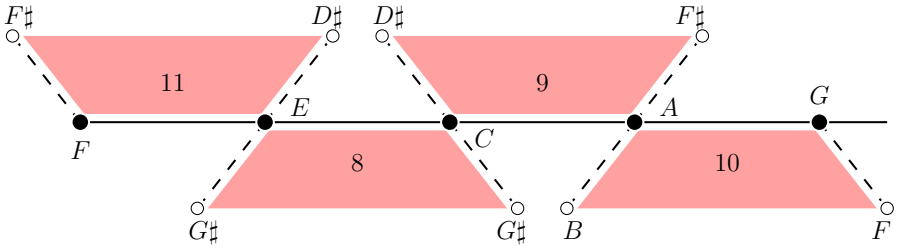


FIG. 3.3 – Représentation spatiale des cinq premières notes de la série tous intervalles de Berg

En associant de cette manière une cellule topologique à chacun des intervalles, on peut réécrire le filtrage des exemples précédents de cette manière :

```
n0 < i1  < I1  > i1  > n1
   < i2  < I2  > i2  > n2
   ...
   < i11 < I11 > i11 > n11
```

On s'apercoit en effet avec la figure 3.3 que la nouvelle contrainte spatiale consiste à insérer dans la construction d'un chemin les surfaces intervalliques. Plus précisément, pour deux notes consécutives $\mathtt{n}(p-1)$ et $\mathtt{n}p$ on doit avoir :

- $\mathtt{n}(p-1)$ et $\mathtt{n}p$ $< 0,1 >$-voisins via $\mathtt{i}p$
- $\mathtt{n}(p-1)$ et $\mathtt{n}p$ $< 0,2 >$-voisins via $\mathtt{I}p$
- $\mathtt{i}p$ incident à $\mathtt{I}p$

Cette approche du problème nous permet de déterminer les 1928 séries tous intervalles en 7 secondes seulement.

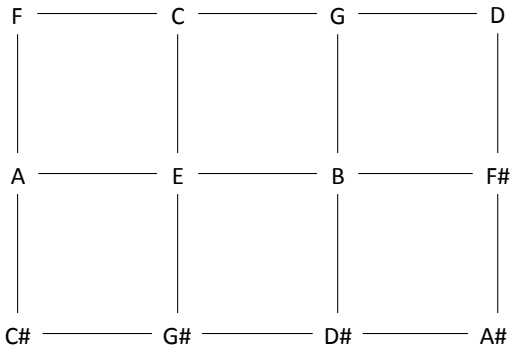


FIG. 3.4 – Le *Tonnetz* de L. Euler (1739)

3.2 Analyses néoriemmaniennes au sein de GBF

3.2.1 La théorie néoriemannienne

A la base introduites par Euler, les représentations géométriques en réseaux de notes de musique suscitent un vif intérêt chez de nombreux musicologues et compositeurs. Jean-Marc Chouvel détaille le fonctionnement et les avantages de ce type de visualisation [Choa, Chob]. Notamment, la possibilité de pouvoir visualiser les accords sous des formes géométriques au sein d'un réseau hexagonal, permet de mettre en évidence des propriétés du point de vue de l'harmonie qu'il était difficile d'isoler via une analyse de partition traditionnelle et ce, aussi bien au sein d'une pièce tonale qu'atonale. Jean-Marc Chouvel explique entre autres que le maillage hexagonal permet la mesure des distances harmoniques, révélant ainsi d'intéressantes propriétés lorsqu'on les compare aux distances contrapunctiques. Par exemple, les notes *do* et *sol* sont distantes dans la gamme chromatique, mais très proche sur le plan harmonique.

D'autre part, la représentation chromatique des notes ne permet à chaque note que d'avoir deux voisins, par exemple, les notes *si* et *do*♯ pour la note *do*. En harmonie tonale, le rôle domainant des accords parfaits mineurs et majeurs montre qu'une note possède plus de deux voisins harmoniques notables. La nécéssité de multi-voisinage d'une note aboutit l'idée de représentation planaire des notes de musique. Euler est le premier à proposer une telle représentation valorisant les intervalles de tierce majeure (vers le bas) et de quinte (vers la droite) avec son *tonnetz* [Eul39] (Figure 3.4). Il est important de remarquer que chaque direction dans un tel graphe implique en fait deux intervalles opposés dont la sommme forme une octave. Par exemple, l'intervalle opposé de la tierce majeure sera la quinte augumentée. C'est la raison pour laquelle le Tonnetz d'Euler permet en réalité des déplacements suivant quatre intervalles opposés deux à deux. Si l'on se place dans $\mathbb{Z}_{12}$ il devient évident que deux intervalles opposés correspondent à une sympathie harmonique équivalente.

Richard Cohn montre [Coh97] comment l'essence harmonique des accords parfaits ainsi que les propriétés des transformations les reliant ont mené à la définition évidente d'un réseau de notes, basé sur les intervalles de tierces et de quinte. En effet, un accord parfait, étant composé :

- d'une tierce majeure
- d'une tierce mineure
- et d'une quinte

il est logique de privilégier ces trois intervalles pour les voisinages de base afin d'obtenir pour un accord parfait une forme géométrique la plus compact possible, synonyme d'homogénéité harmonique

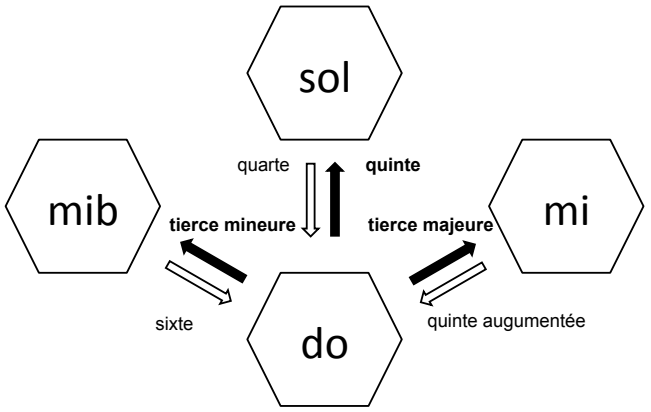


FIG. 3.5 – Construction d'un réseau hexagonal

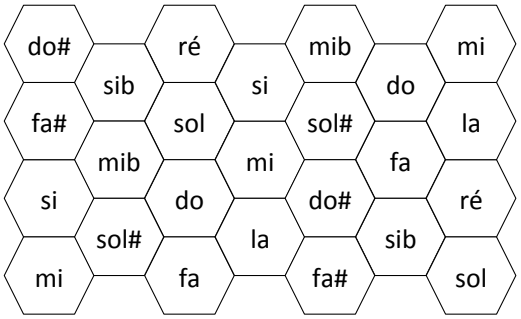


FIG. 3.6 – Un réseau de note néoriemannien

dans le réseau des notes. Une direction comprenant un intervalle et son opposé, cette démarche nous mène à la définition d'un réseau dans lequel chaque note possède 6 voisins, autrement dit, un réseau hexagonal (figure 3.5). On remarque dans ce maillage que la réunion d'un accord majeur, par exemple do, mi, sol correspond à une zone la plus compacte. Comme nous le verrons plus tard, ces trois intervalles ne sont pas les seuls possibles pour la construction d'un réseau hexagonal réunissant la totalité des notes. Toutefois, il est important de remarquer que le maillage hexagonal ne peut être généré par n'importe quelle combinaison de trois intervalles. En effet, on remarque la nécéssité que l'un des trois intervales corresponde à la somme des deux autres. Sans cela il n'est pas possible d'obtenir un pavage régulier de l'espaces. Dans notre cas, on a bien l'intervalle de quinte qui correspond à la somme d'une tierce majeure et d'une tierce mineure.

Il est important de rappeler que nous nous placons pour ce type d'étude dans le cercle chromatique et que deux notes sont considérées comme équivalentes si elles appartiennent à la même classe de hauteur, même si elle n'appartiennent pas à la même octave. Ainsi, il apparait sur la figure 3.6 qu'une même classe de hauteur apparait plusieures fois. Pour éviter cela, il est nécéssaire de faire boucler le maillage suivant les deux dimensions dans lesquelles il s'étend. On obtient un tore. Cette représentation est d'ailleurs souvent appellé *représentation hexagonale toroïdale.*

Une analyse harmonique dans le réseau de notes hexagonale réalisée sur une pièce de Modeste Moussorgski [Choa] révèle comment modéliser un enchainement harmonique par des déplacements spatiaux au sein de ce type de maillage (Figure 3.7).

C'est sur ce réseau que se base Jon Wild [Wil] dans son analyse sur la génération de "gammes

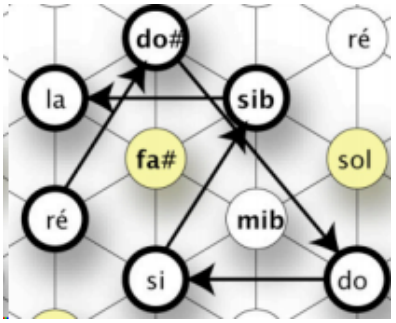


FIG. 3.7 – Un parcours mélodique des notes de basse dans une pièce de Moussorgski. Figure extraite de [Choa]

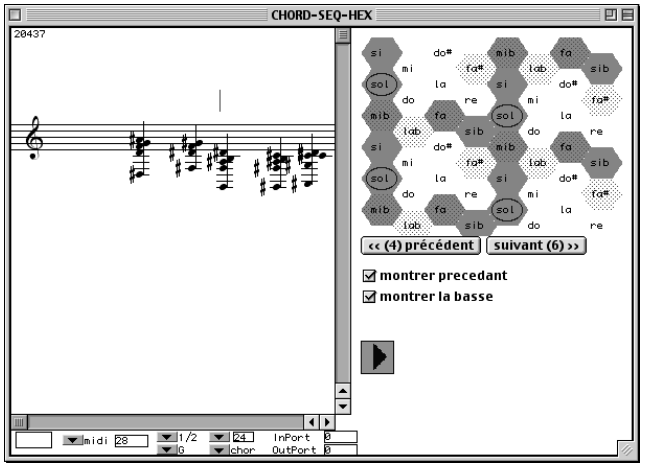


FIG. 3.8 – Interface Omiel

bien formées" introduites par Norman Carey et David Clampitt [CC89]. Cette étude montre elle aussi clairement la pertinence d'un tel maillage.

Une application pour la représentation visuelle des enchainements d'accord dans le réseau hexagonal (OMiel) a déjà été mise en place dans l'environnement Open Music par Benoit Matthieu [Mat02] (Figure 3.8). Outre la possibilité de visualiser les accords sur le damier hexagonal accompagné d'un curseur sur la partition, cet outil offre la possibilité d'entourer la basse réelle de l'accord ainsi que d'afficher en grisé l'accord précédent afin de suivre facilement les déplacements.

3.2.2 Utilisation des GBF pour l'analyse néoriemannienne

Comme on l'a précédemment évoqué, les GBF sont particulièrement adapté dans le cas d'une discrétisation de l'espace de manière régulière. La représentation néoriemannienne correspond à un tel espace. On rapelle qu'un GBF est fondé sur la structure d'un groupe mathématique. Par association avec les propriétés des réseaux de notes présenté précédemment, on propose de travailler sur un groupe mathématique dans lequel :

- les éléments sont les notes de musique
- les générateurs sont les intervalles de quinte, tierce mineure et tierce majeure

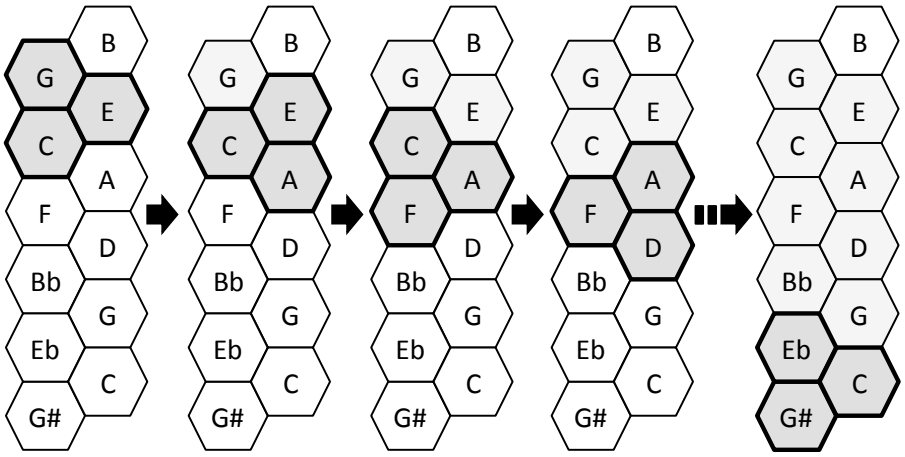


FIG. 3.9 – Progression d'accords extraite du deuxième mouvement de la Symphonie n°9 de L.van Beethoven projeté dans un GBF généré par les intervalles de tierce mineure, tierce majeure et quinte

– l'opération de composition interne correspond aux déplacements élémentaires suivant ces intervalles.

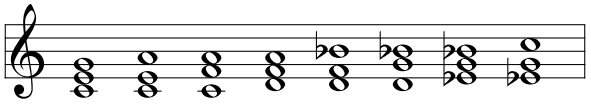
En MGS, la création d'un tel GBF s'écrit de la manière suivante :

```
gbf serie = < MinorThird, MajorThird, Fifth ;
              MinorThird + MajorThird = Fifth
              4 MinorThird = 0
              3 MajorThird = 0
              12 Fifth=0 > ; ;
```

On propose dans la suite de cette section d'étudier deux séquences d'accords dans le réseau hexagonal. Nous commencerons par analyser une progression harmonique issue du deuxième mouvement de la neuvième symphonie de Beethoven dans le maillage traditionnel néoriemannien. Puis nous nous pencherons sur une suite d'accords extraite du prélude n°4 Opus 28 de Frédéric Chopin afin de constater les limites de cette représentation et d'en proposer une adaptation cohérente.

Analyse d'un extrait du deuxième mouvement de la Symphonie n°9 de L. van Beethoven

Cette analyse part de la constation que la progression d'accords en question n'implique qu'un seul changement de note à chaque changement d'accords.



Chacun des accords constitue un accord parfait majeur ou mineur. Projeté dans le maillage néoriemannien, l'enchainement d'accords révèle la formation d'un chemin verticale régulier (figure 3.9).

Analyse d'un extrait du Prélude Op28 n°4 de F. Chopin

On réalise la même étude avec la suite d'accords extraite des huit premières mesures du Prélude Op.28 n°4 de Frédéric Chopin. Comme dans la séquence précédente, une note seulement varie à chaque

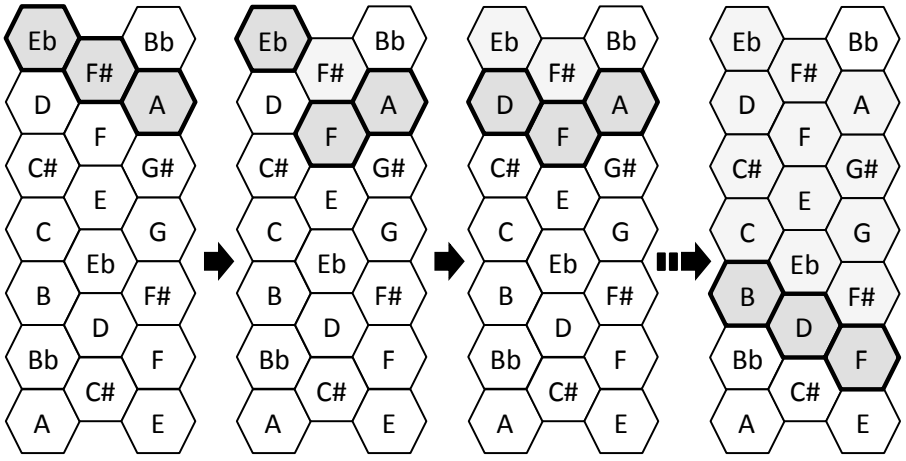
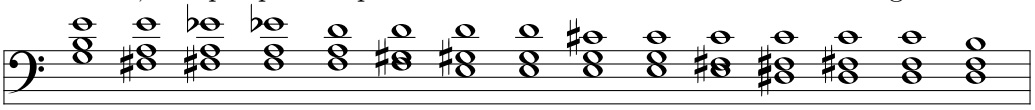


Fig. 3.10 – Progression d'accords extraite du Prélude Op28 n°4 de F. Chopin projeté dans un GBF généré par les intervalles de seconde mineure, tierce mineure et tierce majeure

changement d'acords, excepté pour le premier enchainement où deux notes changent.



La projection de cet enchainement dans le maillage hexagonale ne révèle pas de progression spatiale aussi claire que dans l'enchainement précédent. En effet, la représentation révèle des accords moins compacts et des déplacements dépourvus de toute régularité.

En analysant plus précisément la progression d'accord, on réalise que la quasi-totalité des enchainements d'accords révèlent une note évoluant suivant un interval de demi-ton. Le maillage que l'on a présenté dans la section précédente ne met pas en valeur l'intervalle de demi-ton. On cherche donc à définir un réseau plus adapté dans lequel l'interval de seconde mineure comporte plus de signification. Rappelant que l'un des intervalles doit correspondre à la sommme des deux autres, on remarque qu'en procédant seulement à un remplacement de l'interval de quinte par un interval de seconde mineure, on obtient un nouveau maillage conforme à cette règle. En effet, l'intervalle de tierce majeure correspond bien à la juxtaposition d'une tierce mineure et d'une seconde mineure. En projetant la progression d'accords du Prélude de Chopin dans le nouveau réseau de notes, on observe une évolution spatiale beaucoup plus significative. Comme pour l'enchainement d'accords de la section précédente, on percoit un cheminement vertical régulier balayant l'ensemble des notes de manière continu (figure 3.10). Un réseau organisé suivant de tels générateurs nous permet de constater une autre propriété intéressante : En effet, on observe sur la figure 3.10 que le balayage des notes s'effectue de manière quasi-totale, la seule note omise étant le Si♭. La tonalité du Prélude étant Mi mineure, la note Si♭ peut être considérée comme la note la plus étrangère à la tonalité du morceau car séparée d'un triton avec la tonique. L'intervalle de triton est en effet considéré comme la plus dissonant en harmonie tonale. L'exception réalisée sur cette note peut être interprétée comme une démarche relativement cohérente avec les usages tonales du courant musical auquel appartient le compositeur.

Il semble donc qu'il soit nécéssaire d'adapter les GBF à la séquence analysée. Si les premiers maillages hexagonaux ont été imaginé dans une démarche visant à valoriser les intervalles les plus importants en harmonie tonale, il semble que cette dernière règle ne conserve pas sa cohérence pour

d'autres exemples. Cela ne nous permet pas pour autant d'affirmer que la pièce de F. Chopin se situe dans un contexte moins tonale que celle de L. van Beethoven, néanmoins on peut y voir un signe cohérent avec l'évolution des courants musicaux ayant abouti à l'emergence de travaux dans l'atonalité.

On peut alors imaginer pouvoir associer des structures de GBF à certaines pièce, voir les considérer comme étant la signature de certains compositeurs ou de certains courants musicaux. Notemment, étant fréquent dans la musique contemporaine de rechercher des dissonances, on pourrait par exemple imaginer des collections GBF générées à partir d'intervalles de seconde mineure et de triton comme plus adaptés à l'analyse de ce type de pièces.

3.3 Analyse de suites d'accords

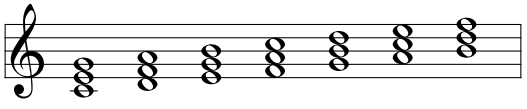
Une analyse mathématique d'objets musicaux repose souvent sur la définition de modèles dont les caractéristiques décrivent des propriétés musicales. Dans les sections précédentes, on s'est attaché à définir des espaces afin d'y réaliser des projections de certains problèmes musicaux. Ces projections nous ont permis de mettre en évidence des propriétés sur les éléments musicaux étudiés à l'aide d'un point de vue spatiale. Dans la suite de ce chapitre, on propose la démarche inverse, consistant à générer un espace *à partir* des objets musicaux étudiés. Plus spécialement, nous décrirons un mécanisme d'auto-assemblage afin de spécifier une représentation spatiale de la tonalité basée sur les accords triadiques. Puis nous proposons d'appliquer ce procédé à une représentation de la tonalité basée sur les accords à quatre sons, et à l'analyse d'une série d'accords du même prélude de F. Chopin.

3.3.1 Représentation spatiale de la tonalité

Une tonalité, au sens traditionnel, est souvent associé à une gamme de notes, comprenant une note associée à la tonique et une associée au mode (mineur ou majeur). Par exemple, la tonalité de *do majeur* est associé à une gamme de notes (do, re, mi, fa, sol, la, si) commencant par la tonique do et incluant la note mi indiquant qu'on se situe dans le mode majeur.

La tonalité peut aussi dans certains cas être caractérisée par le recouvrement des notes de la gamme par les accords triadiques correspondant aux différents degrés de la tonalité. On se propose d'analyser ces recouvrements d'un point de vue spatial. Pour la tonalité de do majeur, les accords à trois sons associés aux différents degrés de la tonalité sont les suivants :

$$I_{Do} = \{do, mi, sol\} \quad II_{Do} = \{re, fa, la\} \quad III_{Do} = \{mi, sol, si\}$$
$$IV_{Do} = \{fa, la, do\} \quad V_{Do} = \{sol, si, re\} \quad VI_{Do} = \{la, do, mi\}$$
$$VII_{Do} = \{si, re, fa\}$$



Cette définition de la tonalité peut être interprétée de manière géométrique. On représente chaque note de la gamme par un point. Puis, si deux notes appartiennent au moins à un même de ces accords, on relie les points associés par une arrête. Enfin, les accords triadiques apparaissent dans le réseau sous la forme de surfaces triangulaires, spécifiées pas les triplets de notes correspondantes. La figure 3.11 montre la représentation géométrique de la tonalité de do majeur.

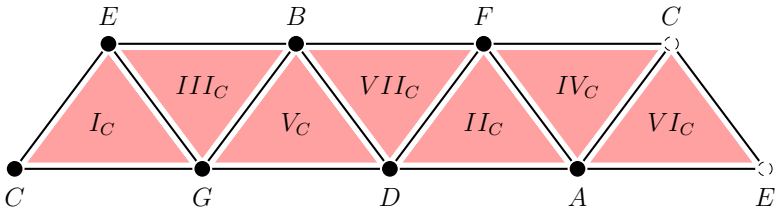


FIG. 3.11 – Representation de la tonalité de do majeur

Si l'on referme le ruban obtenu en appliquant la règle d'assemblage précédemment évoqué sur les notes située aux extrémités de la bande obtenue (do et mi), on obtient un ruban de Möbius. Cet espace aux propriétés remarquables a été mis en évidence par A. Schoenberg [Sch11] et repris dans le cadre de plusieurs analyses depuis [M$^+$02]. Le ruban de Möbius a pour propriété de n'avoir qu'une seule arrête. Une propriété notable consiste en le fait que cette arrête, qui constitue un $<0, 1>$-chemin, fait apparaitre le cercle des quintes, élément fondamental en harmonie tonale. Il est aussi possible de tirer des propriétés de la représentation duale de cet espace. Cette représentation duale correspond aussi à un ruban de Möbius dont l'arrête fait apparaitre un enchainement d'accords séparés de quintes [Maz07].

3.3.2 Auto-assemblage d'accords

L'exemple précédent montre quel intérêt nous apporte la représentation géométrique d'une succession d'accords. C'est la raison pour laquelle on propose à présent un algorithme d'auto-assemblage pour la construction automatique de structures topologiques associées à des séquences d'accords. L'idée générale est de laisser *réagir* entre eux les accords à la manière de molécules chimiques. La règle de réaction consiste alors à identifer des éléments communs aux accords afin de pouvoir procéder à des *appareillements* entre accords, la structure topologique résultante pouvant être riche en propriétés concernant la suite d'accords.

Présentation des complexes simpliciaux

Un *complexe simplicial* est un complexe cellulaire au sein duquel les cellules topologiques sont des *simplexes*. Un p-simplexe est une p-cellule qui a exactement $p + 1$ faces. Par exemple une arête, lorsqu'elle est bordée par deux sommets, est simplexe. Il s'agit bien d'une 1-cellule bordée par 2 0-cellules. Un hexagone n'est pas un simplexe car il s'agit d'une 3-cellule comportant 6 faces. La figure 3.12 expose la représentation géométrique des p-simplexes pour $p \in 0, 1, 2$.

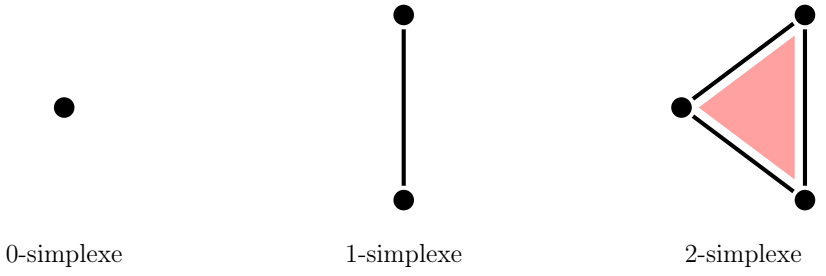


FIG. 3.12 – Représentation géométrique de trois simplexes

Un complexe composé exclusivement de simplexes est un *complexe simplicial*. Un graphe, ou encore le ruban de Möbius de la figure 3.11, sont des exemples de complexes simpliciaux.

Représentation d'accords

Les accords sont les éléments de base à assembler. Un accord étant un ensemble de notes, on propose dans la suite de représenter les accords par des *simplexes*, éléments topologiques présentés dans le chapitre 2. Dans ce contexte, un accord composé de n notes est représenté par un simplexe de $n-1$ dimensions, autrement dit, un $(n-1)$-simplexe. Par exemple, l'accord I_{Do} correspondra a la surface triangulaire $\{do, mi, sol\}$. Ce triangle sera bordé par les arrêtes $\{do, mi\}$, $\{mi, sol\}$ et $\{do, sol\}$. Notons que ces 3 1-simplexes correspondent directement aux trois intervalles fondamentaux de l'harmonie tonale, les tierces et la quinte, évoqués section 3.2. Cela explique en partie pourquoi le ruban de Moebius capture si bien les relations harmoniques au sein de la tonalité.

Mécanisme d'auto-assemblage

On souhaite construire des représentations simpliciales d'ensemble d'accords à l'aide d'un *processus de croissance par accrétion* [GS08]. Ce mécanisme est basé sur l'identification des bords des simplexes. Cette identification doit pouvoir se faire pour des simplexes de toute dimension. En effet, pour l'étude précédente, on réalise que appareiller les accords I_{Do} et III_{Do} nécéssite la reconnaissance de trois éléments : les sommets *mi* et *sol* ainsi que l'arrête $\{mi, sol\}$, comme on peut l'observer sur la figure 3.13.

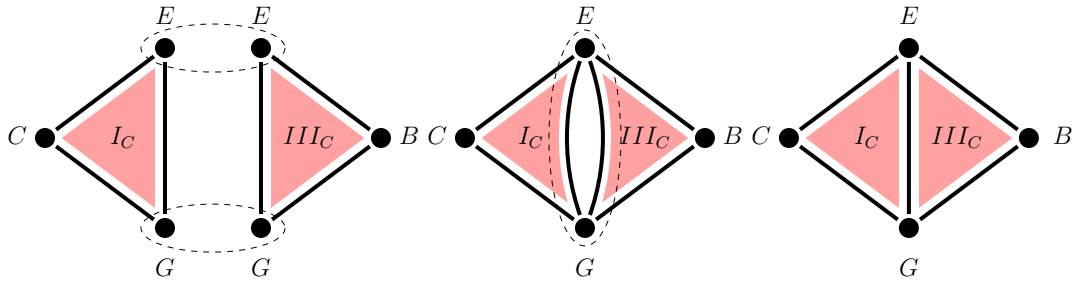


FIG. 3.13 – Identification de bords pour l'appareillement de simplexes

En MGS, ce processus d'appareillement peut s'écrire de cette manière :

```
transformation identification = {
  s1 s2 / (s1 == s2 & faces(s1) == face(s2))
  =>
  let c = new_acell( (dim s1)
                     (faces s1)
                     (union ((cofaces s1),
                            (cofaces s2))))
  in  s1*c
}
```

Cette règle spécifie que deux éléments `s1` et `s2` étant décorés de la même manière et possédant les même faces, sont substitués par un nouvel élément `c` (qui comporte comme cofaces l'union de celles

de `s1` et de `s2`) et comportant la même décoration que `s1` (et donc que `s2`). Dans la figure 3.13 la transformation `identification` est appelée deux fois. Lors de la première application (du complexe de gauche à celui du milieu), les sommets sont identifiés. Les deux opérations topologiques sont effectuées en parallèle. A la seconde application (du complexe du milieu à celui de droite), les deux arrêtes reliant les sommets associés aux notes *mi* et *sol* qui partagent a prèsent les mêmes bords se confondent à leur tour en un unique élément. Le simplexe résultant est une arrête possèdant comme cofaces les 2-simplexes I_{Do} et III_{Do}. Finalement, au stade le plus à droite, plus aucune réaction n'est susceptible de s'effectuer, car le point limite est atteint.

Conformément à ce à quoi on s'attendait, le point de stabilité de l'application de la transformation `identification` aux accords triadiques I_{Do}, ..., VII_{Do} correspond à la construction du ruban de Möbius (figure 3.11).

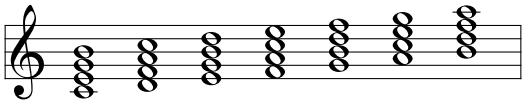
A présent que l'on dispose d'un mécanisme pour l'analyse automatique de séquence d'accords, on propose de l'appliquer dans deux cas : le recouvrement de la gamme diatonique par les accords à quatre sons, et le prélude Op.28 N°4 de F. Chopin.

3.3.3 Recouvrement de la gamme par des accords à quatre sons

Dans certains styles de musique, comme le jazz, les accords sont souvent plus riches que les degrés triadiques. Il est ainsi possible d'associer aux degrés d'une tonalité des accords à quatre sons aboutissant à une représentation de la tonalité de la même manière qu'avec les accords à trois sons. On propose d'utiliser le mécanisme précédemment décrit afin d'obtenir une nouvelle représentation de la tonalité à partir des accords à quatre sons recouvrant la gamme diatonique.

Les degrés associés sont les suivants :

$$I_{Do} = \{do, mi, sol, si\} \quad II_{Do} = \{re, fa, la, do\} \quad III_{Do} = \{mi, sol, si, re\}$$
$$IV_{Do} = \{fa, la, do, mi\} \quad V_{Do} = \{sol, si, re, fa\} \quad VI_{Do} = \{la, do, mi, sol\}$$
$$VII_{Do} = \{si, re, fa, la\}$$



La représentation géométrique d'un accord à quatre notes est un 3-simplexe, ce qui correspond à un tétraèdre. La construction du complexe simplicial associé à cette ensemble d'accords est impossible en 3 dimensions avec des tétraèdres réguliers, ce qui rend son étude topologique difficile. En effet, si il est possible d'assembler les uns à la suite des autres les simplexes associés aux sept tétraèdres, on se retrouve face à une impossibilité géométrique pour l'assemblage du premier et du dernier (figure 3.14).

Toutefois, si les tétraèdres étaient déformables, tout en restant topologiquement invariant, on pourrait visualiser la nature géométrique du complexe simplicial et réaliser qu'il s'agit d'un tore. La transformation `identification`, grâce à son implémentation dépourvue de contrainte dimensionnelle, nous permet de générer ce complexe simplicial et de constater effectivement qu'il s'agit bien d'un tore.

Contrairement au ruban de Möbius, le tore est un objet topologique orientable. Une question se pose donc : pourquoi la représentation de la tonalité basée sur ses degrés d'accords, perd t-elle sons caractère non orientable en considérant les accords à 4 sons plutôt qu'à 3 sons ?

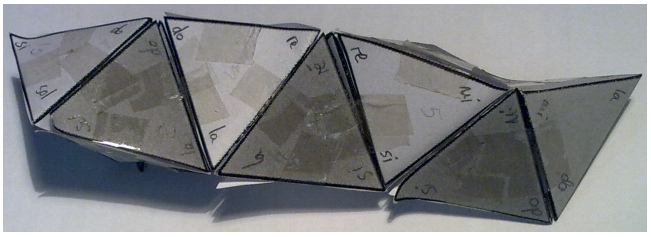


FIG. 3.14 – Représentation de la tonalité de Do majeur par assemblages d'occords à quatre sons

3.3.4 Représentation simpliciale d'un extrait du Prélude Op28 n°4 de F. Chopin

Dans la section 3.2, nous avons proposé de construire un espace dans lequel projeter une séquence d'accords issue d'un prélude de Chopin dans le maillage hexagonale issue de la tradition néoriemannienne. Nous proposons à présent d'étudier cette même séquence mais cette fois ci en temps qu'espace elle-même. Pour cela, de la même manière que dans le paragraphe précédent, nous associons un simplexe à chacun des accords de la séquence afin d'obtenir le complexe simplicial résultant.

La transformation `identification` nous permet d'obtenir cette représentation de l'ensemble des accords. Cette strucure est là encore difficilement visualisable. Toutefois, il est intéressant de constater que le 1-simplexe (composé de deux notes) appartenant au plus grand nombre de 2-simplexe (accords à trois sons) est composé des notes *mi* et *sol* qui révèlent la tonalité du Prélude (*mi* mineur).

Un sous-complexe associé à un enchainement de cinq accords de la séquence est représenté dans la figure 3.15.

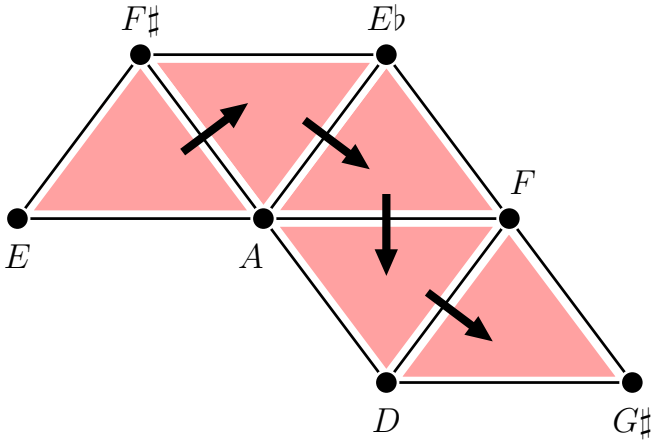


FIG. 3.15 – F. Chopin Prelude 28 simplical representation

Comme précédemment évoqué, une propriété remarquable de cette séquence réside dans le fait qu'une note seulement change à chaque enchainent d'accord. Cette caractéristique est valable pour les quatorze premiers accords en partant du deuxième. Composé d'accords à trois sons, cette progression d'accords correspond à un $<2, 1>$-chemin dans le complexe simplicial associé à l'ensemble des accords. Ce chemin est composé des 2-simplexes (les accords) connectés les uns aux autres par des 1-simplexes représentant les deux notes communes aux accords. Ce chemin est représenté par les flèches noire sur la figure 3.15 pour les cinq accords considérés. Une énumération des chemins de longueur maximale, d'une manière similaire à celle employée pour le calcul des séries tous intervalles dans la section 3.1, a été

effectuée. Il est intéressant de noter qu'il éxiste 120 chemins possibles, et que parmis ces chemins, celui utilisé par F. Chopin pour composer son Prélude est celui comportant la plus petite distance entre les accords. Ce résultat comporte une certaine logique dans le sens où les notes changantes entre les accords consécutifs de ce passage sont des demi-tons. Ce résultat confirme l'importance de l'intervalle de demi-ton nous ayant mené à le privilégier comme générateur pour la construction du maillage hexagonal support de l'étude de cette pièce (section 3.2).

Chapitre 4

Approche symbolique

Les travaux décrits dans ce chapitre proposent une approche originale pour l'analyse de séquences musicales. Pour cela, on propose de considérer ces séquences comme des successions d'évènements dans le temps. Les outils proposés par la Q-Analyse conviennent à ce type d'étude. On présentera dans une première partie les aspects généraux de la Q-Analyse. Puis nous montrerons comment ce type de raisonnement a aboutit à l'établissement d'une chaîne de traitement pour l'analyse de séquences mélodiques. Enfin, nous exposerons plusieurs exemples en appliquant cette démarche à des comptines.

4.1 Présentation de la Q-Analyse

L'idée de base de la Q-Analyse est de représenter une relation binaire entre deux ensembles à l'aide d'un complexe simplicial [VGb]. Ce concept initialement utilisé pour la modélisation des relations sociales [Atk74], l'analyse de trafic ou encore l'analyse des positions aux échecs, a récemment été repris avec pour objectif la résolution automatique de tests de QI [VS99], [VGa].

Il s'agit ici de se concentrer sur la manière dont les éléments d'un système peuvent être regroupés et la façon avec laquelle la structure de connexité résultante permet de caractériser les propriétés du système étudié [cas97]. Ces caractéristiques permettent alors de procéder à une extraction de connaissances du système [Val97]. La figure 4.1 donne un exemple de visualisation d'un complexe simplicial associé à l'étude de la connexité entre un ensemble de fleurs et un ensemble de couleurs. Notons qu'il a ici été choisi de représenter les fleurs par des sommets et les couleurs par des simplexes. Ce choix est arbitraire, et la démarche inverse, aboutissant à une représentation duale, offrirait une représentation différente du système, révélant de nouvelles propriétés.

On peut alors porter ce type d'étude à un système évoluant dans le temps. C'est ainsi qu'il a été imaginé d'appliquer ce type d'analyse aux contes de fées. Un exemple sur le conte "Le petit Chaperon Rouge" [GV] explique cette démarche, consistant à associer un ensemble de prédicats (rouge, dormir, la forêt, etc.) à chacune des scènes de l'histoire, afin d'en tirer une visualisation globale basée sur la déformation d'un complexe simplicial au fur et à mesure des scènes (figure 4.2), ainsi que sur les propriétés de connexité associée au "chemin" emprunté par cette déformation.

Pour l'analyse d'une séquence musicale, on génère une base de segments de référence, puis on dresse une matrice d'incidence à l'aide d'une découpe de la séquence en "scènes musicales". Cette matrice d'incidence nous permet de définir un complexe simplicial rendant visible certaines informations qui ne l'auraient pas forcément été via une analyse classique. L'étude de la connexité entre différents complexes pourrait alors révéler des informations intéressantes, par exemple sur la manière avec laquelle faire évoluer une pièce musicale à la manière d'une autre.

λ	tulipe	coquelicot	pensée	myosotis
rose	1	0	1	1
jaune	1	1	1	0
bleu	0	0	1	1
blanc	1	1	1	1

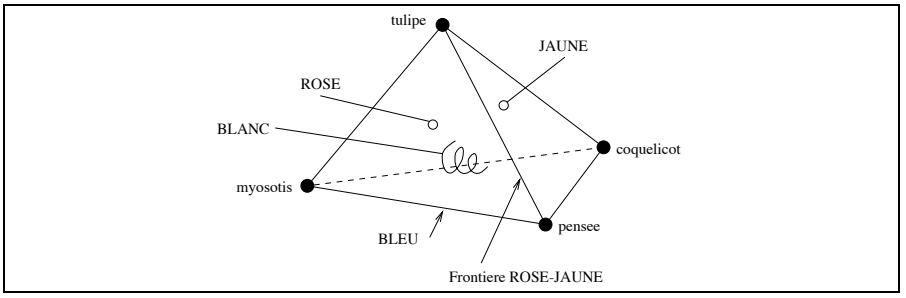


FIG. 4.1 – Complexe associé aux fleurs et à leurs couleurs

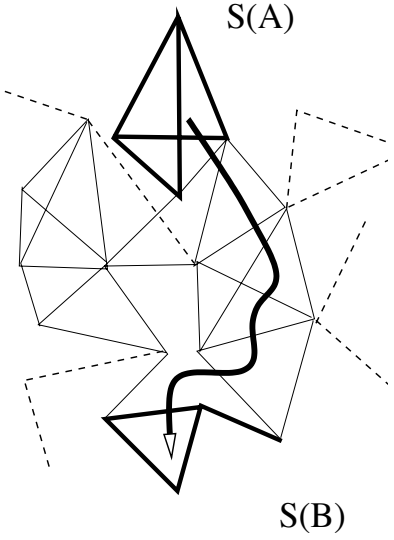


FIG. 4.2 – Exemple de déformation d'un complexe simplicial

Enfin, on conçoit facilement que ce type d'étude basé sur le calcul de représentations spatiales requiert des outils appropriés. Le langage MGS qui permet la manipulation de notions topologiques répond à ces attentes.

4.2 Etude symbolique de séquences musicales

4.2.1 Chaîne de traitement

Nous proposons ici une méthode d'analyse musicale reposant sur l'utilisation de la Q-analyse. L'application de cet outil à l'étude des contes de fées montre sa pertinence pour l'analyse de séquences d'évènements se succédant dans le temps. Or, une pièce musicale correspond à un type de structure

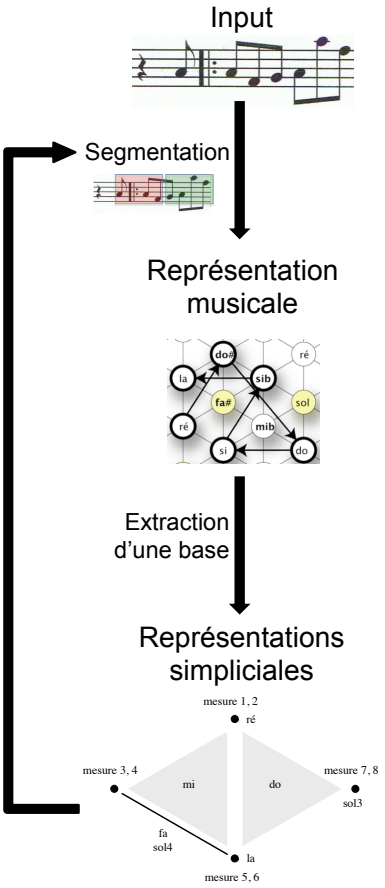


FIG. 4.3 – Chaine de traitement pour l'analyse d'une séquence musicale

similaire. On propose de développer la chaîne de traitement suivante (figure 4.3) pour l'extraction d'informations au sein d'une séquence musicale.

Cette chaîne prend en entrée une séquence midi. Un patch OpenMusic a été développé afin d'extraire les données midi et de les ré-écrire de manière a pouvoir être interprétés convenablement par un programme MGS. Le programme traite alors les données à la manière d'une collection topologique de type séquence. Le traitement consiste en une segmentation, l'extraction d'une base puis la construction d'une matrice d'incidence pour générer un complexe simplicial associé à la pièce musicale étudiée.

4.2.2 Segmentation

Tout comme pour l'analyse de contes de fées, on réalise rapidement la nécéssité de discrétiser le temps afin de porter notre étude sur les différentes *scènes* constituant la pièce. Dans notre cas, il est nécéssaire d'effectuer une segmentation sur la séquence musicale.

La segmentation musicale est un problème bien connu dans le domaine de l'analyse musicale. Un certain nombre de travaux ont été effectués dans ce domaine. Si l'analyse d'une succession d'accords se fait élément par élément, une séquence mélodique au contraire nécessitera dans la plupart des cas une judicieuse découpe au préalable. La segmentation peut être horizontale, mais aussi verticale. En effet certaines analyses se révèleront plus cohérentes si elles sont issue d'une découpe sur l'échelle

des hauteurs plutôt que sur l'échelle temporelle. On mesure rapidement l'impact des choix relatifs à la segmentation sur les résultats de l'analyse. Un grand nombre de travaux ont été menés à ce titre. Olivier Lartillot [Lar03] propose un système de segmentation basé sur la construction de motifs. Pierre-Yves Rolland quant à lui, se penche sur la détection de régularités [Rol98]. D'autres méthodes d'extraction de métrique [Meu02] ou de rythme [GM03] ont été menées. De nombreux autres travaux comme ceux de Cambouropoulos [Cam98], constituent également d'importants éléments dans le domaine de la segmentation automatique.

La segmentation de la pièce en *scènes musicales* reste une démarche arbitraire dont le paramétrage est fortement discutable. Notre étude en étant à un stade relativement exploratoire, nous proposons dans un premier temps une découpe de segments régulière au sein d'une mélodie. Pour des séquences musicales suffisamment cohérentes avec la métrique du morceau, les scènes musicales sont associées aux mesures du morceau. Toutefois, on permet à l'utilisateur de rentrer en paramètre de l'analyse la taille de découpe qu'il jugera la plus pertinente.

4.2.3 Extraction d'une base

Comme on l'a vu lors de la présentation de la Q-analyse, ce type d'étude comprend l'extraction d'une base, qui servira de référence pour l'analyse de chacun des segments. La base doit regrouper les éléments considérés comme les briques élémentaire de la séquence. Dans notre étude, nous nous en tiendrons à une constitution de base particulière, dite par *fragmentation*, dont nous exposons le principe à travers un exemple simple :

Nous proposons d'extraire la base associée à deux segments : [a,b,c] et [a,d].

Le mécanisme consiste à isoler les sous-segments communs de longueur maximale. Dans cet exemple, le seul sous-segment commun aux deux éléments est l'élément [a]. On place cet élément dans la base et on procède à une réduction des deux segments [a,b,c] et [a,d] par substitution du sous-segment commun [a]. Les deux segments résultant sont [b,c] et [d]. On vérifie à nouveau qu'ils ne comportent pas de sous-segments en commun. Si un nouveau sous-segment commun est détecté on procède à nouveau à une fragmentation. Si ce n'est pas le cas, on place les segments réduits dans la base. Les segments [b,c] et [d] ne comportent pas de sous-segments communs, ils rejoignent donc la base. Finalement, la base résultante est constituée des éléments [a], [b,c] et [d].

Ce processus se fait à partir d'un ensemble de segments déterminés précisément pour la constitution de la base. Il ne s'agit pas des segments associés aux scènes musicales que nous avons évoqué dans le paragraphe précédent. On réalise que ces segments représentent d'importants paramètres rentrant en jeu lors de l'analyse. On permet donc à l'utilisateur de spécifier la longueur de segment qu'il juge la plus cohérente pour la constitution de la base.

4.2.4 Représentation simpliciale

Une fois la segmentation réalisée et la base construite, les deux ensembles sont mis en relations afin de déterminer une matrice d'incidence. Cette matrice d'incidence met en évidence la présence ou non de chacun des éléments de la base dans chacun des segments. Voici la manière dont elle est constituée :

- une colonne est associée à un segment. Il peut s'agir par exemple d'une mesure.
- une ligne est associée à une brique élémentaire de la base.
- l'intersection d'une ligne et d'une colonne consiste en une valeur booléenne égale à :
  - 1 si le segment associé à la ligne contient l'élément de la base associé à la colonne.
  - 0 sinon

Cette matrice aboutit à la génération d'un complexe simplicial : On associe chacune des scènes musicales à un 0-simplexe, autrement dit à un sommet. Puis on associe un simplexe à chacun des éléments de la base. Ce simplexe est d'une dimension d'autant plus grande que le nombres de scènes dans lesquelles il apparait, de manière à ce qu'il puisse être bordé par tous les 0-simplexe correspondant aux scènes en question. Dans l'exemple précédent, si la matrice d'incidence fait apparaitre que l'élément [b,c] apparait dans 4 scènes, alors cet élément sera représenté sous la forme d'un 4-simplexe, autrement dit un tétraèdre. Le mécanisme d'auto-assemblage précédemment décrit nous permet de réunir ces simplexes en un même complexe simplicial.

Pour une matrice d'incidence, deux représentations duales sont possibles. On choisit d'associer un 0-simplexe à chacun des éléments de l'un des deux ensembles. La matrice d'incidence nous permet ensuite de représenter les éléments du deuxième ensemble par propriété d'incidence avec les éléments du premier. Rappelons que ce complexe est associé à une visualisation d'informations qui reste très relative aux choix de l'utilisateur. En effet, les paramètres de segmentation et de constitution de la base sont déterminant pour une telle représentation. Pour cela, on propose plusieurs complexes simpliciaux associés aux séquences analysées, privilégiant le contenu mélodique ou encore rythmique.

4.3 Exemples

On présente dans cette section divers résultats obtenus après application du processus d'analyse précédemment exposé.

Comme nous l'avons constaté, l'analyse d'une pièce musicale à l'aide de la Q-analyse est une chose complexe. Notre étude étant à un stade exploratoire et une séquence musicale étant très riche en paramètres, nous avons choisis dans un premier temps de travailler sur des séquences musicales les plus simples possibles. C'est la raison pour laquelle nous présentons ici l'étude simpliciale de contines traditionelles.

Nous exposerons dans un premier temps un exemple détaillé avec la comptine *Frère Jacques*. Puis nous exposerons les résultats provenant d'autres comptines.

4.3.1 Analyse détaillé de la comptine *Frère Jacques*

Voici un exemple détaillé de l'étude de la contine *Frère Jacques* (figure 4.4) suivant le processus exposé plus haut. Le patch OpenMusic nous permet d'extraire depuis la séquence midi associée à la contine une séquence textuelle que l'on va pouvoir traiter :

60	0	500	100	1	62	500	500	100	1	64	1000	500	100	1	60	1500	500	100	1

Cette séquence fournit cinq informations sous forme numérique, relatives à chacune des notes :

1. la hauteur
2. la date de début
3. la durée
4. la vélocité
5. le canal midi

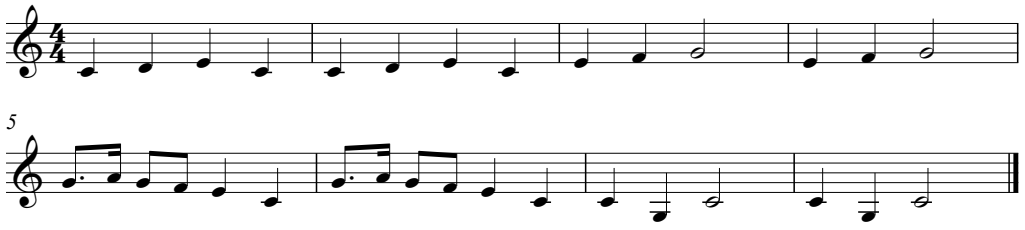


FIG. 4.4 – Séquence mélodique utilisée pour l'étude de la contine *Frère Jacques*

En MGS, on associera chacune des notes comprise dans la séquence midi à une structure de données de type enregistrement détaillée dans la section 2.1.

Dans un premier temps, on propose une étude basée sur la présence des différentes hauteurs au cours de la pièce. Seuls les trois premiers paramètres, relatifs au temps et à la hauteur de la note nous intéressent donc. L'unité des paramètres relatifs au temps (la date de début et la durée de la note) est la miliseconde. On paramètre une segmentation basée sur la durée d'une demi-mesure pour la constitution de la base. La durée d'une noire est ici égale à 500ms. La durée d'une demi-mesure a donc pour valeur 1000ms.

Les hauteurs de notes sont représentées par leur valeur midi. Le tableau suivant représente un extrait du tableau d'équivalences entre hauteurs de note et valeures midi :

55	56	57	58	59	60	61	62	63	64	65
$sol3$	$sol\sharp 3$	$la3$	$la\sharp 3$	$si3$	$do4$	$do\sharp 4$	$re4$	$re\sharp 4$	$mi4$	$fa4$

La première opération consiste à découper la séquence en segments successifs de 1000ms.

Une fois l'ensemble des segments déterminé, on procède à la construction de la base par fragmentation. On rapelle qu'on ne s'intéresse dans ce cas qu'à la présence des notes en tant que hauteur uniquement. La vélocité et le canal n'interviennent pas dans la construction de cette base. La segmentation ayant déjà pris compte de l'ordonnancement temporel des notes, seul l'information sur la hauteur de la note, *le pitch*, rentre ici en compte. On ramène donc par exemple les deux premiers segments de la comptine :

60	0	500	100	1	62	500	500	100	1

64	1000	500	100	1	60	1500	500	100	1

aux deux segments suivants :

60	62

64	60

A partir de ces segments, une fonction a été développée en MGS afin de générer la base. La fonction est appelée autant de fois qu'il y a de segments. Lorsqu'elle est appelée, la fonction prend le segment en question en argument, et génère la nouvelle base à partir de celle qui existait déjà. Evidemment, pour le premier segment considéré, la base est vide. La fonction se contente alors d'inclure le segment

tel quel dans la base. A partir du deuxième segment, il s'agit d'appliquer des fonctions de détection de sous-segments évoqués précédemment afin d'effectuer correctement la fragmentation lorsqu'elle s'avère nécéssaire. Voici la manière dont cela s'effectue pour les deux premiers segments :

1. Au départ la base est vide.
$$base = \emptyset$$
2. Le premier segment est intégré tel quel dans la base.
$$base = \boxed{60 \| 62}$$
3. La fonction considère le deuxième segment $\boxed{64 \| 60}$
4. L'élément $\boxed{60}$ est détecté comme sous-segment commun avec l'un des segments de la base.
5. Le segment de la base $\boxed{60 \| 62}$ est oté de la base car il comprend le sous segment $\boxed{60}$
6. La fragmentation divise ce segment en plusieurs nouveaux : le sous-segment $\boxed{60}$ lui même, et les éléments qui *l'entourent*. Dans notre cas il ne s'agit que de l'élément $\boxed{62}$. Ces éléments sont réinjectés dans la base.
$$base = \boxed{60}\ \boxed{62}$$
7. Le segment $\boxed{64 \| 60}$ est divisé de la même manière. Ses éléments sont injecté dans la base.
$$base = \boxed{60}\ \boxed{62}\ \boxed{64}\ \boxed{60}$$
8. La base étant implémentée sous la forme d'une collection topologique de type `set`, c'est à dire d'un ensemble, les doublons sont automatiquement supprimés. La base réelle à l'issu de la prise en compte des deux premiers segments est la suivante :
$$base = \boxed{60}\ \boxed{62}\ \boxed{64}$$

Notons que la fonction de fragmentation a été écrite de manière à laisser intacte les éléments entourant le sous-segment ayant engendré la fragmentation. Cependant, il est possible qu'une fragmentation en entraine une autre. Par exemple, la fragmentation de la séquence

55	59	60	64	65	67	65	60	64	55

par la séquence

60	64

aboutit à l'ensemble

55	59		60	64		65	67	65		55

Ce nouvel ensemble comprend des éléments possédant une sous-séquence commune :

55	59	55

La nécéssité d'une nouvelle fragmentation est alors automatiquement détectée. Il s'agit d'un calcul de point fixe. Il aboutit dans notre exemple à la base :

55	59	60	64	65	67	65

Ce processus de génération de base a été proposé afin conserver les segments caractéristiques de la pièce. Toutefois, d'autres possibilités de génération de base sont envisagables en fonction de la pièce étudiée.

Une fois tous les segments de la comptine pris en compte pour la constitution de la base, on obtient l'ensemble suivant :

$$base = \boxed{55}\ \boxed{60}\ \boxed{62}\ \boxed{64}\ \boxed{65}\ \boxed{67}\ \boxed{69}$$

On s'apercoit que le processus de fragmentation a ramené la base à l'ensemble des notes utilisée dans la comptine. Il s'agit d'un cas spécial, mais qui montre les limites de ce processus.

On établit alors la matice d'incidence par détection de chacun des éléments de la base dans chacune des mesures. On aboutit à la matrice suivante :

	Mesure1	Mesure3	Mesure5	Mesure7
55	0	0	0	1
60	1	0	1	1
62	1	0	0	0
64	1	1	1	0
65	0	1	1	0
67	0	1	1	0
69	0	0	1	0

La comptine Frère Jacques est constitué de 4 mesures répétées 2 fois. C'est la raison pour laquelle il est suffisant de ne représenter que les mesures impaires. Deux complexes simpliciaux résultent de cette matrice :

- le premier complexe associe des 0-simplexes aux segments de la base puis des n-simplexes aux mesures de la pièce.
- le second simplexe associe des 0-simplexes aux mesures de la pièce puis des n-simplexes aux segments de la base.

Notons qu'une matrice d'incidence symétrique aboutira à deux complexes duaux isomorphe.

Pour chacune des représentations, un problème de dimension est susceptible de nous empêcher de visualiser le résultat. En effet, dans le premier cas, le complexe sera de dimension superieure à 3 si au moins une mesure contient plus de quatre segments de la base. Le problème se posera pour le deuxième complexe si au moins un segment de la base est présent dans plus de quatre mesures. Ce problème n'est gênant que d'un point de vue humain car nous ne pouvons pas imaginer l'allure d'un complexe simplicial de dimension supérieure à 3. Un programme informatique, au contraire, est capable d'en extraire des propriétés.

Le premier complexe simplicial issu de l'étude de la comptine Frère Jacques est visibles sur la figure 4.5. Le complexe dual n'est pas représentable pour les raisons évoquées ci-dessus. En effet, on s'aperçoit que la mesure 5, comprenant 5 segments de la base, doit être associée à un 4-simplexe correspondant à un objet en 4 dimensions.

Il est difficile d'extraire des informations de tels résultats. En effet, les représentations simpliciales de la comptine pourraient présenter des aspects très différents selon les choix associés aux multiples critères qu'il est nécéssaire de paramétrer au départ de l'analyse. Toutefois, il est intéressant de constater que

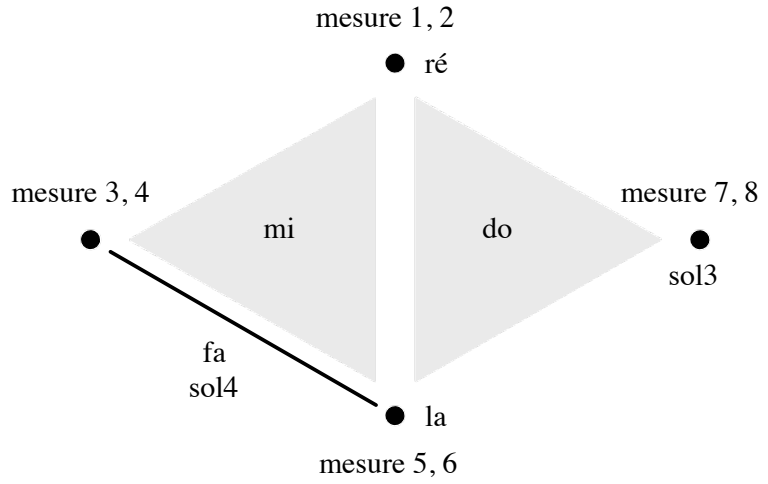


FIG. 4.5 – Complexe simplicial associé à la contine Frère Jacques

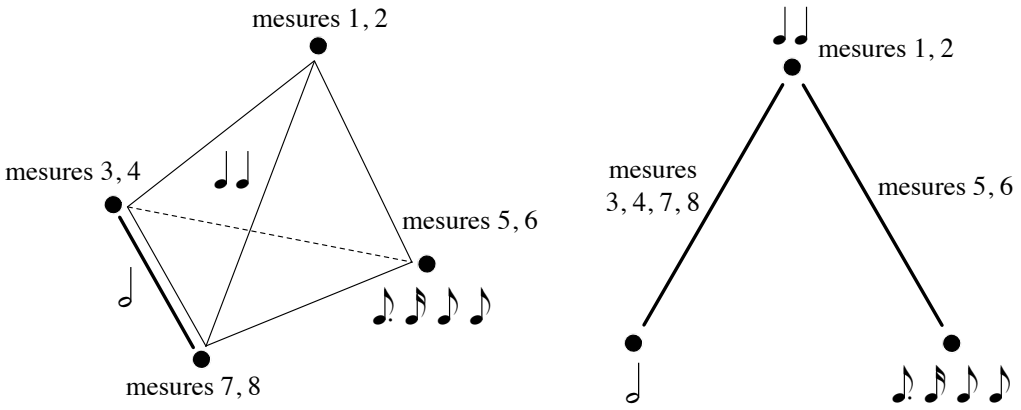


FIG. 4.6 – Complexex simpliciaux associés à l'étude rythmique de la comptine Frère Jacques

les deux simplexes de plus haute dimension au sein du complexe généré dans notre cas, sont associés aux hauteurs *do* et *mi*, révélant respectivement la tonalité et le mode de la comptine. Il s'agit en effet d'une pièce en *do* majeur.

Nous procédons à présent à la même analyse portée sur les valeurs des durées des notes de la séquence mélodique. Il s'agit donc d'une étude portant sur les propriétés rythmiques de la comptine. Nous obtenons la matrice d'incidence suivante, associée à deux complexes simpliciaux duales (figure 4.6).

	Mesure1	Mesure3	Mesure5	Mesure7
[500,500]	1	1	1	1
1000	0	1	0	1
[375,125,250,250]	0	0	1	0

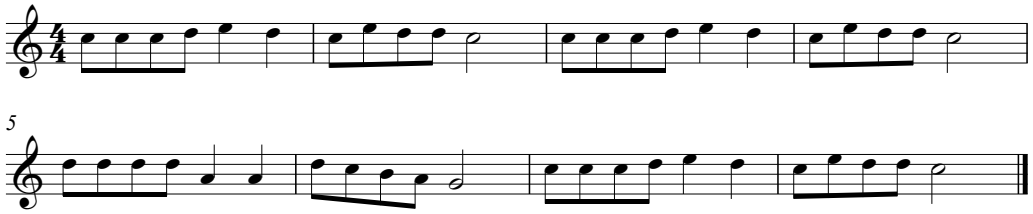
La multiplicité des représentations possibles pour une pièce nous pousse à réfléchir à la constitution d'une boucle de rétroaction, ou d'une selection automatique des complexes simpliciaux les plus informatifs. Pour cela, il est nécéssaire de comprendre quelles sont les propriétés topologiques ayant un sens

dans le domaine musical afin de privilégier les représentations les mettant en évidence. Ces questions se trouvent au centre des travaux qui se feront dans la continuité de ce stage.

4.3.2 Résultats pour d'autres comptines

Afin de ne pas nous heurter à des problèmes de dimensions de complexes évoqués plus haut, nous choisissons de travailler sur des extraits de comptines limitant la longueur des séquences analysées.

Au clair de la lune

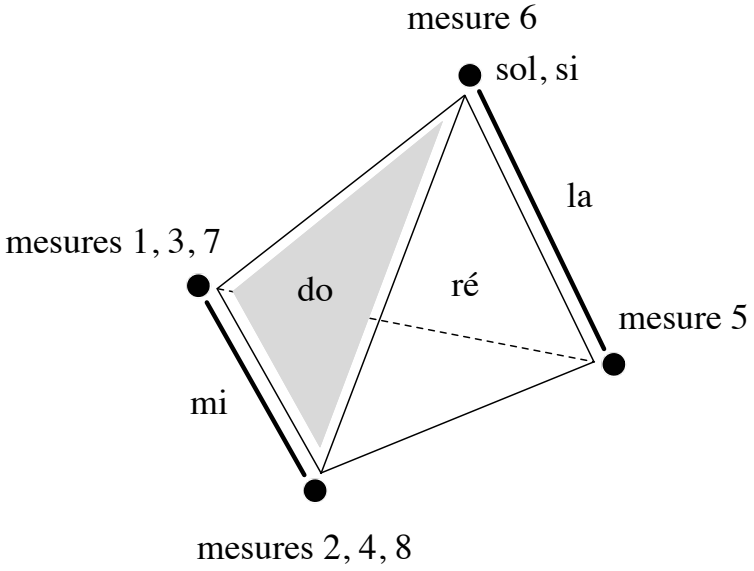


Les mesures 3, 7 et 4, 8 sont respectivement identiques aux mesures 1 et 2. On peut donc réduire l'analyse à l'étude des mesures 1, 2, 5 et 6.

Matrice d'incidence :

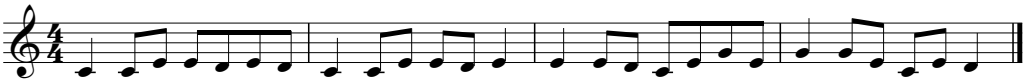
	Mesure1	Mesure2	Mesure5	Mesure6
55	0	0	0	1
57	0	0	1	1
59	0	0	0	1
60	1	1	0	1
62	1	1	1	1
64	1	1	0	0

Complexe simplicial :



On remarque que la sixième mesure utilise 5 des 6 segments de la base. Par conséquent, le complexe dual n'est pas représentable en trois dimensions.

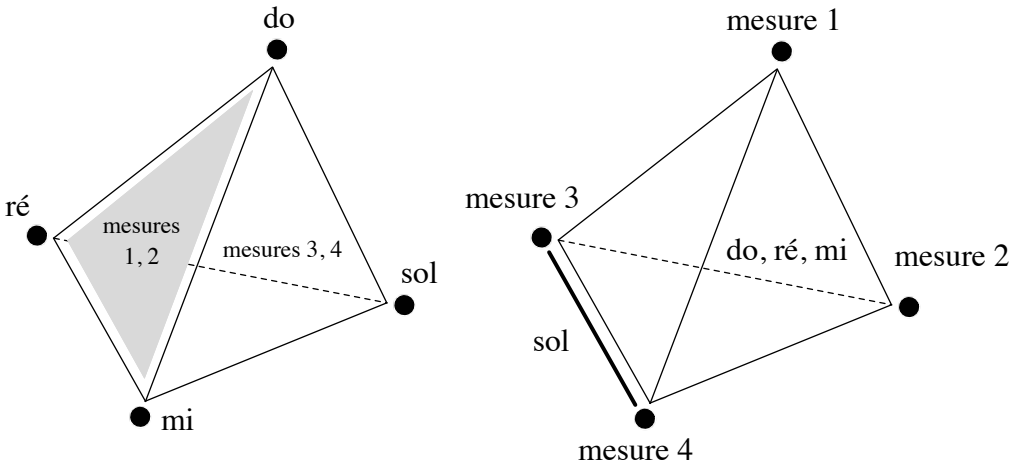
A la claire fontaine



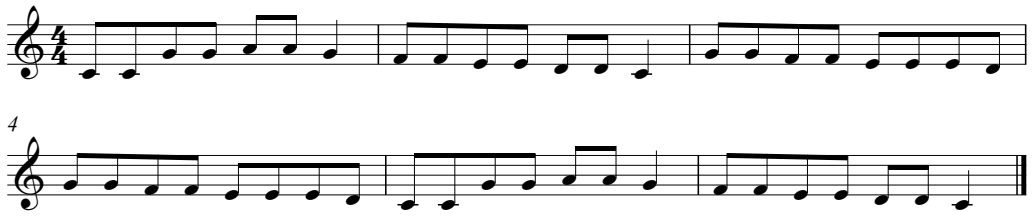
Matrice d'incidence :

	Mesure1	Mesure2	Mesure3	Mesure4
60	1	1	1	1
62	1	1	1	1
64	1	1	1	1
67	0	0	1	1

Complexes simpliciaux :



A vous dirais je maman

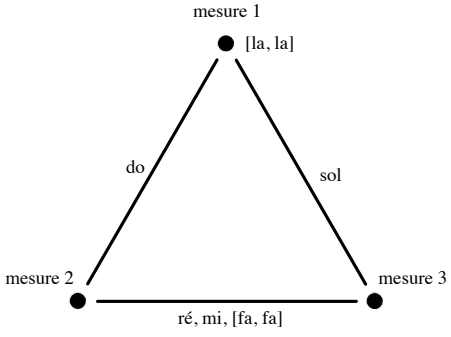


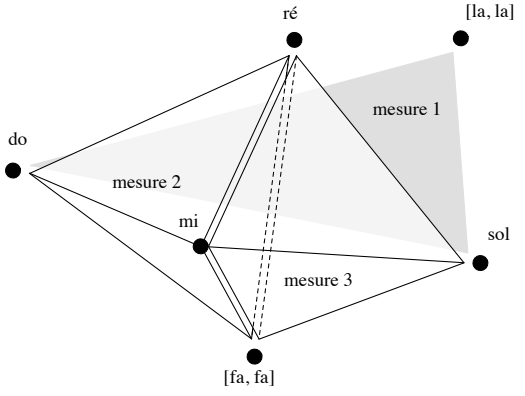
Matrice d'incidence :

	Mesure1	Mesure2	Mesure3
60	1	1	0
62	0	1	1
64	0	1	1
65,65	0	1	1
67	1	0	1
69,69	1	0	0

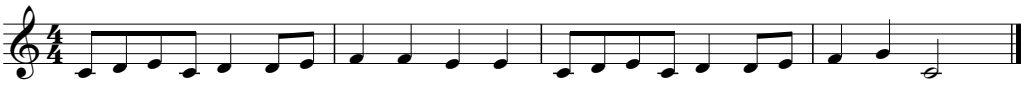
Il est intéressant de noter pour cet extrait que les segments | 65 | 65 | et | 69 | 69 | n'ont pas été décomposés par le processus de fragmentation. En effet, une propriété de cet estrait réside dans le fait que les notes fa et la n'apparaissent que sous la forme de doublons consécutifs. Notons que la note mi qui apparait tantôt sous la forme d'un doublet tantôt sous la forme d'un triplet, se retrouve par conséquent fragmenté à une note simple. Cette dernière constatation peut paraitre incohérente et fait partie des points nécéssitant réflexion pour l'amélioration de l'application.

Complexes simpliciaux :





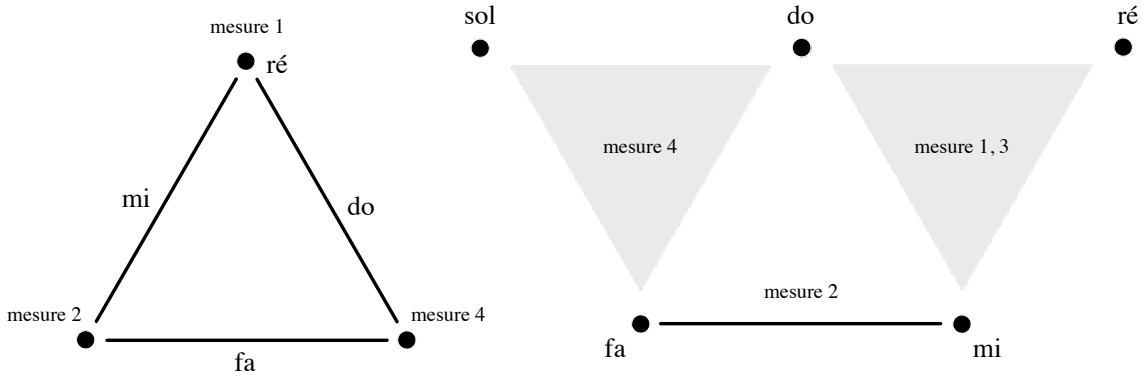
J'ai du bon tabac



Matrice d'incidence :

	Mesure1	Mesure2	Mesure4
60	1	0	1
62	1	0	0
64	1	1	0
65	0	1	1
67	0	0	1

Complexes simpliciaux :



Comme nous l'avons souligné, nous parvenons difficilement à extraire des informations des représentations topologiques associées aux comptines. Cependant, la question concernant les paramétrages de la segmentation et de l'extraction de la base gagneraient à être approfondis et pourraient nous mener à des résultats plus significatifs. D'autre part, la simplicité des comptines relativise considérablement l'intérêt des propriétés que l'on peut en tirer. Enfin, on se retrouve freiné pour la visualisation massive de complexes simpliciaux. Il n'existe pas de système générant automatiquement des visualisations de complexes et il nous est nécéssaire de les tracer à la main à partir de chacune des matrices d'incidence.

Chapitre 5

Conclusion

Ces travaux montrent qu'il est pertinent d'avoir recours à des outils spatiaux pour des problèmes d'analyse musicale. Initilament conçu pour tenter de fournir aux biologistes un outil de programmation dédié à la simulation, le langage MGS offre de nouvelles approches dans le domaine des représentations de la musique. On a particulièrement constaté l'originalité que propose ce paradigme lorsqu'il permet de projeter une oeuvre dans un espace sous forme de support avec les GBF. Les possibilités offertes par cet outil sont encore loin d'avoir été exploités à la hauteur de leur potentiel. Les différentes représentations possibles sont nombreuses grâce aux diverses combinaisons de générateurs qu'il est possible d'imaginer.

Notamment, comme on l'a constaté avec le Prélude de F. Chopin, il peut être possible d'aboutir à un espace pour la projection d'une oeuvre, à partir de propriétés résultantes d'un espace correspondant à l'oeuvre elle même.

Comme évoqué précédemment, ces études nous ramènent souvent à nous questionner sur la manière d'interpréter des propriétés topologiques dans le domaine musicale. Ces questions sont au centre des travaux qui se feront dans la continuation de ce stage.

Les résultats décrits dans le chapitre 4 sont les moins aboutis. Mais comme on l'a souligné, ce type d'analyse s'avèrerait probablement plus intéressante appliquée de manière plus approfondi et à des pièces plus complexes et donc plus riches en propriétés. La Q-Analyse ayant montré son intérêt dans l'analyse de comptes de fées, on pourrait imaginer l'appliquer à des analyses de pièce d'opéra permettant d'extraire des propriétés mettant en évidence des relations entre musique et intrigue.

D'autres nombreuses perspectives restent à explorer notamment dans le domaine de la composition musicale. Des problèmes relatifs à la polyphonie et au contrepoint pourraient par exemple trouver un sens en se placant dans certains espaces de contraintes. On pourrait aussi réfléchir à la notion de voisinage dans des collections topologiques regroupant de nouveaux paramètres, comme le timbre. A terme, il serait intéressant de pouvoir intégrer à la plate forme Open Music développée par l'équipe Représentations musicales de l'Ircam de nouvelles fonctionnalités offertes par la programmation spatiale.

Les résultats présentés dans les chapitres 3 et 4 constituent l'aboutissement de travaux menés au cours de ce stage. Les différentes approches présentées dans ce document proposent des démarches à des stades relativement exploratoires pour des problèmes qui peuvent sembler disparates. Cependant, les résultats obtenus sont encourageant et ont abouti à un projet de thèse ainsi qu'à la publication d'un article dans le cadre de la conférence Spatial Computing Workshop à Budapest en Septembre 2010 [BSM].

Bibliographie

[And03] Moreno Andreatta. *Méthodes algébriques en musique et musicologie du XXème siècle : aspects théoriques, analytiques et compositionnels*. PhD thesis, EHESS/IRCAM, 2003.

[Atk74] Ronald Atkin. *Mathematical Structure in Human Affairs*. Heinemann, 1974.

[BSM] Louis Bigo, Antoine Spicher, and Olivier Michel. Spatial programming for music representation analysis. Spatial Computing Workshop 2010.

[Cam98] Emilios Cambouropoulos. *Towards a General Computational Theory of Musical Structure*. PhD thesis, The University of Edinburgh, 1998.

[cas97] *Reality rules : II. Picturing the world with mathematics*. Wiley-Interscience, 1997.

[CC89] Norman Carey and David Clampitt. Aspects of well-formed scales. *Music Theory Spectrum*, 1989.

[Choa] Jean-Marc Chouvel. Analyser l'harmonie - aux frontières de la tonalité. `http://jeanmarc.chouvel.3.free.fr/textes/AnalyserHarmonie0.3.pdf`.

[Chob] Jean-Marc Chouvel. Au-delà du système tonal. `http://jeanmarc.chouvel.3.free.fr/textes/LeSystemeTonal0.2.pdf`.

[Coh97] Richard Cohn. Neo-riemannian operations, parsimonious trichords, and their "tonnetz" representations. *Journal of Music Theory*, 1997.

[Eim64] Herbet Eimert. *Grundlagen der musikalischen Reihentechnik*. Vienna : Universal Edition, 1964.

[Eul39] Leonhard Euler. *Tentamen novae theoriae musicae ex certissismis harmoniae principiis dilucide expositae*. Saint Petersburg Academy, 1739.

[Gia03] J.-L. Giavitto. Topological collections, transformations and their application to the modeling and the simulation of dynamical systems. In *Rewriting Technics and Applications (RTA'03)*, volume LNCS 2706 of *LNCS*, pages 208 – 233, Valencia, June 2003. Springer.

[GM01a] J.-L. Giavitto and O. Michel. Declarative definition of group indexed data structures and approximation of their domains. In *Proceedings of the 3nd International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming (PPDP-01)*. ACM Press, September 2001.

[GM01b] Jean-Louis Giavitto and Olivier Michel. Mgs : a rule-based programming language for complex objects and collections. In Mark van den Brand and Rakesh Verma, editors, *Electronic Notes in Theoretical Computer Science*, volume 59. Elsevier Science Publishers, 2001.

[GM02] J.-L. Giavitto and O. Michel. Data structure as topological spaces. In *Proceedings of the 3nd International Conference on Unconventional Models of Computation UMC02*, volume 2509, pages 137–150, Himeji, Japan, October 2002. Lecture Notes in Computer Science.

[GM03] Fabien Gouyon and Benoit Meudic. Towards rhythmic content processing of musical signals : Fostering complementary approaches. 2003.

[GS08] Jean-Louis Giavitto and Antoine Spicher. *Systems Self-Assembly : multidisciplinary snapshots*, chapter Simulation of self-assembly processes using abstract reduction systems, pages 199–223. Elsevier, 2008. doi :10.1016/S1571-0831(07)00009-3.

[GV] Jean-Louis Giavitto and Erika Valencia. Combinatorial algebraic topology for diagrammatic reasoning. Technical Report 1165, LRI, ura 410, april 1998.

[Lar03] Olivier Lartillot. Une analyse musicale automatique suivant une heuristique perceptive. 2003. EGC'2003, Atelier "Fouille de Données", Lyon.

[LS01] R.C. Lyndon and P.E. Schupp. *Combinatorial group theory*. Springer Verlag, 2001. reprint of vol. 89 of Ergebnisse der Mathematik und ihrer Grenzgebiete, Springer (1977).

[M+02] G. Mazzola et al. *The topos of music : geometric logic of concepts, theory, and performance*. Birkhäuser, 2002.

[mam09] Séminaire mamux, november 2009. `http://recherche.ircam.fr/equipes/repmus/mamux/ProgrNov09.html`.

[Mat02] Benoit Mathieu. Outils informatiques d'analyse musicale. Master's thesis, 2002. IRCAM.

[Maz07] Guerino Mazzola. *La vérité du beau dans la musique*. Delatour, 2007.

[Meu02] Benoit Meudic. Automatic meter extraction from midi files. 2002. Proc. Journées d'informatique musicale.

[Mun84] James Munkres. *Elements of Algebraic Topology*. Addison-Wesley, 1984.

[Rei85] David L. Reiner. Enumeration in music theory. *The American Mathematical Monthly*, 92(1) :51–54, January 1985.

[RM06] André Riotte and Marcel Mesnage. *Formalismes et modèles musicaux.1*. Delatour, 2006.

[Rol98] Pierre-Yves Rolland. *Découverte de régularités dans les séquences et application à l'analyse musicale*. PhD thesis, Université de Paris VI, 1998.

[Sch11] Arnold Schoenberg. *Harmonielehre*. 1911.

[Spi06] Antoine Spicher. *Transformation de collections topologiques de dimension arbitraire. Application à la modelisation de systèmes dynamiques*. PhD thesis, 2006.

[Val97] Erika Valencia. Un modèle topologique pour le raisonnement diagrammatique. Master's thesis, August 1997. Rapport pou le DEA Sciences Cognitives LIMSI.

[VGa] Erika Valencia and Jean-Louis Giavitto. Algebraic topology for knowledge representation in analogy solving. In ECAI, editor, *European Conference on Artificial Intelligence (ECAI 98)*, pages 88-92, Brighton, UK, 23-28 August 1998. Christian Rauscher.

[VGb] Erika Valencia and Jean-Louis Giavitto. Topologie algébrique pour la construction de représentations diagrammatiques. In Paris 8, editor, *7ème Colloque de l'Association pour la Recherche Cognitive*, Paris, France, dic 1998. ARC.

[VS99] Erika Valencia and Jean-Paul Sansonnet. Formalisme simplicial pour la représentation spatiale des connaissances et application au raisonnement par analogie. 1999. LIMSI ; Université Paris-Sud.

[Wil] Jon Wild. Pairwise well-formed scales and a bestiary of animals on the hexagonal lattice. Proceedings of the MCM Conference 2009, Yale, New Haven.