Université de Lille 1 SCIENCES ET TECHNOLOGIES

Projet n°85 : *Développement d'une interface Javascript dédiée à la visualisation harmonique de séquence musicales*

Etudiant : Pierre Caruyer

Encadrant : Louis Bigo - Laboratoire CRIStAL(équipe Algomus)

Table des matières

1.Introduction....3
2.Présentation du projet....6
2.1.L’équipe Algomus....6
2.2.Objectif global....6
3.Les applications existantes....7
3.1.Présentation de TonnetzViz....7
3.2.Présentation de mudcube.js....7
4.Objectifs du projet....8
4.1.Premier objectif....9
4.2.Second objectif....9
5.Enjeux du projet....10
6.Objectifs remplis....11
6.1.Premier objectif....11
....11
6.2.Second objectif....11
....11
6.3.Objectif secondaire....13
....13
7.Conclusion....14
8.Remerciements....15
9.Lexique....16
10.Annexe....17

1.Introduction

1.1.Une représentation usuelle de la musique

Habituellement, lorsque l'on représente un morceau de musique, on le représente sous la forme d'une suite de notes appelée partition.
Dans une partition, chaque note est représentée, dans un ordre chronologique, par son timbre (do, ré, mi, fa, etc ...), sa hauteur – grave ou aiguë - sa durée et son intensité. ce qui permet aux musiciens de reproduire le son comme le compositeur l'a prévu lorsqu'il a écrit son morceau.

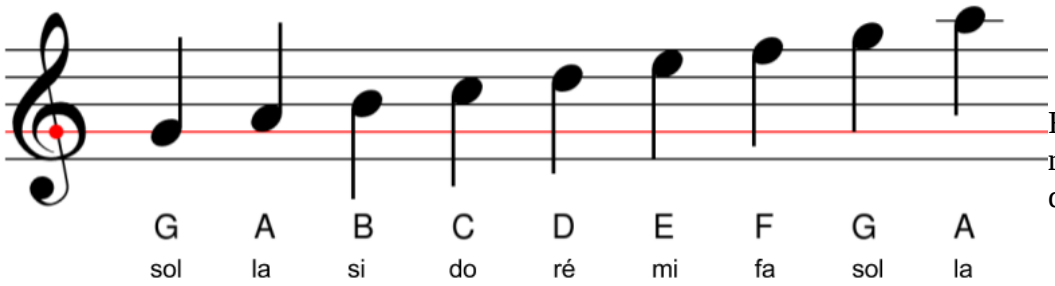
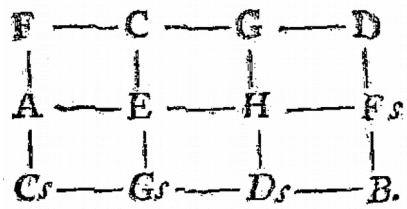


Fig.1.Note représentées dans une portée

1.2.Un peu de théorie musicale

Fig.2.Tonnetz d'Euler



Il existe d'autres façons de représenter la musique comme le Tonnetz qui a été inventé par Leonhard Euler en 1739.
Le Tonnetz correspond à une organisation des notes de musique et des accords dans l'espace, dans laquelle la distance entre les éléments reflète leur consonance sur le plan musical. La notion de proximité entre les notes permet d'attribuer un score (par exemple, le score sera plus ou moins bas en fonction du nombre d'accord dissonants que comporte la musique) à toute pièce de musique, en offrant ainsi différentes approches pour l'analyse et la classification stylistique.

Pour représenter la musique dans le Tonnetz, nous ne nous soucierons pas de l'octave, c'est-à-dire que les 88 touches piano seront réduites à 12 classes de hauteur : Do, Do#, Ré, Ré#, Mi, Fa, Fa#, Sol, Sol#, La, La#, Si.

1.3.La segmentation

Il est possible de faire le lien entre une partition et le Tonnetz car celui-ci permet de représenter un accord (ensemble de notes) à un moment précis de la musique. Une partition peut donc être décomposée en une suite d'accord, ce que l'on appelle la segmentation pour être représentée dans le Tonnetz.

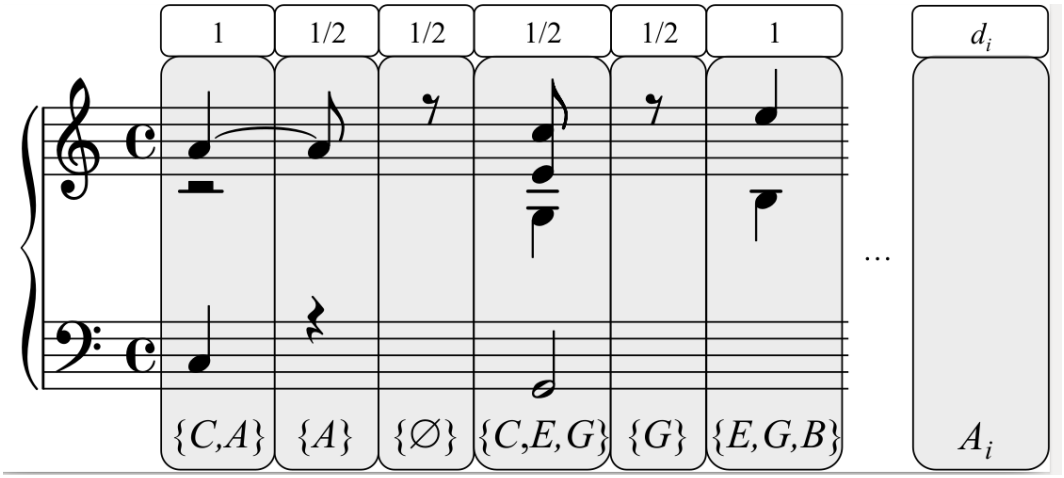


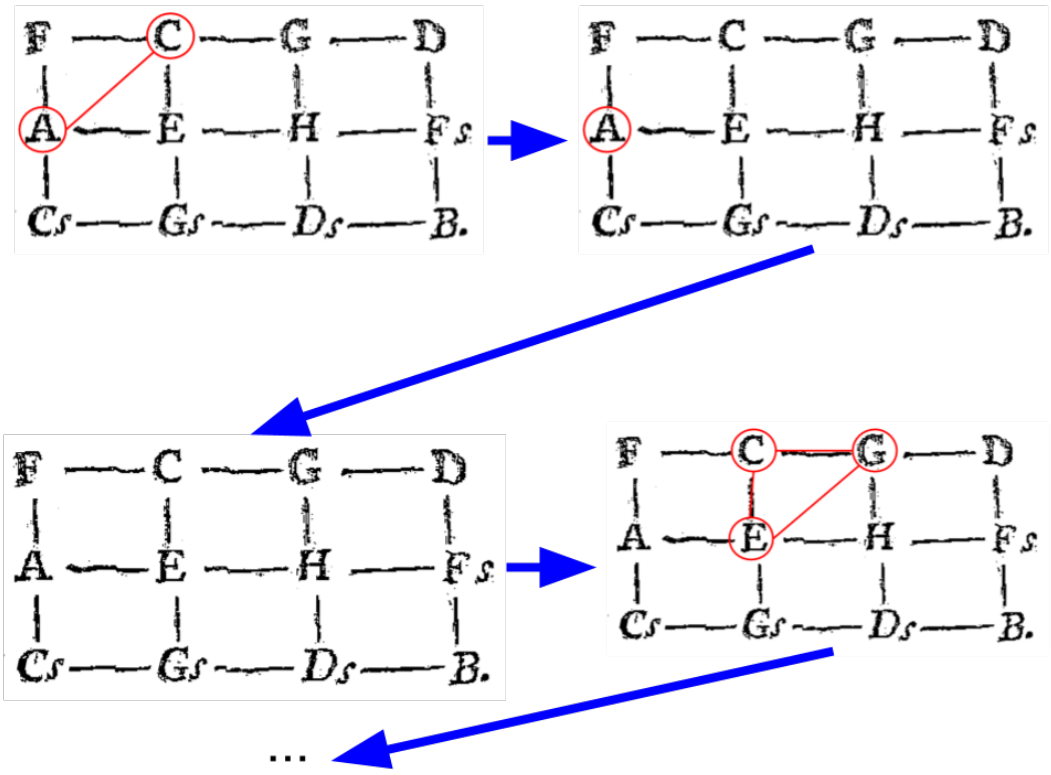
Fig.3.Segmentation d'un pièce musicale

Ici, on peut voir la représentation d'un cours morceau de musique joué par deux instruments à la fois. Chaque portée correspondant à un instrument. Sous cette portée, sont représentées les notes correspondantes jouées par les instruments dans le Tonnetz, que l'on appelle segmentation.

1.4.Représentation dans le Tonnetz

On peut ensuite utiliser cette segmentation pour représenter cette partition dans le Tonnetz. Chaque segment correspondant à un instant de la musique, on pourra désormais représenter la musique étudiée sous la forme de ces instants dans le Tonnetz comme le présente la figure suivante.

Fig.4.Représentation d'une pièce musicale dans le Tonnetz d'Euler



2.Présentation du projet

2.1.L’équipe Algomus

Ce projet est proposé par l’équipe « Algomus », spécialisée dans l’élaboration d’algorithmes dédiés à l’analyse musicale, aussi appelée musicologie assistée par ordinateur.
L’équipe Algomus développe des outils permettant aux musicologues d’étudier des pièces musicales de différents styles sur les plans mélodiques, harmoniques et rythmiques.

2.2.Objectif global

L’analyse et la représentation de progressions harmoniques (suite d’accords) y constitue un axe central. Le logiciel HexaChord développé en Java (site web en footnote) a été développé dans le but de visualiser et d’analyser ces progressions durant leur écoute à l’aide de représentations dérivées du *Tonnetz*.

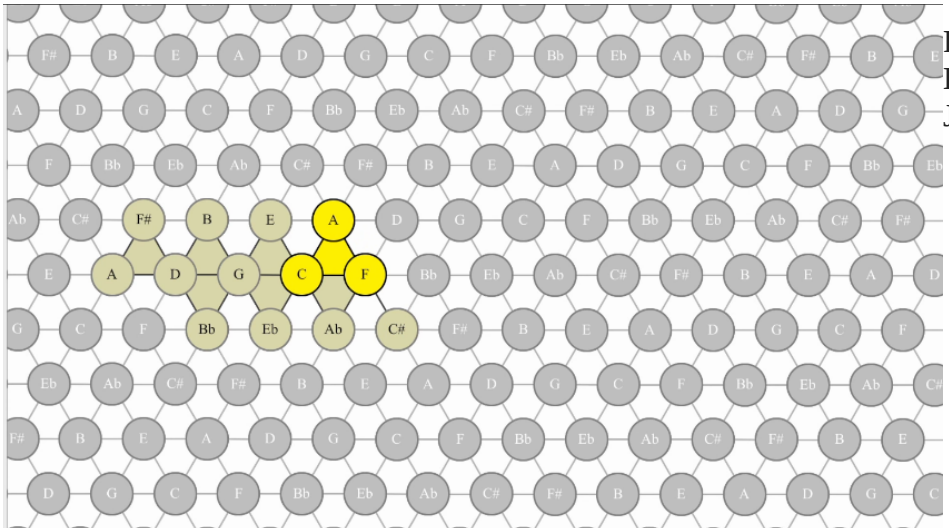


Fig.5 Le logiciel Hexachord développé en Java par l’équipe Algomus.

Le présent projet a pour but d’étudier les outils disponibles pour l’implémentation de fonctionnalités de bases de HexaChord en langage JavaScript, pour cela nous allons nous baser sur deux applications existantes, TonnetzViz et mudcube.js.

3.Les applications existantes

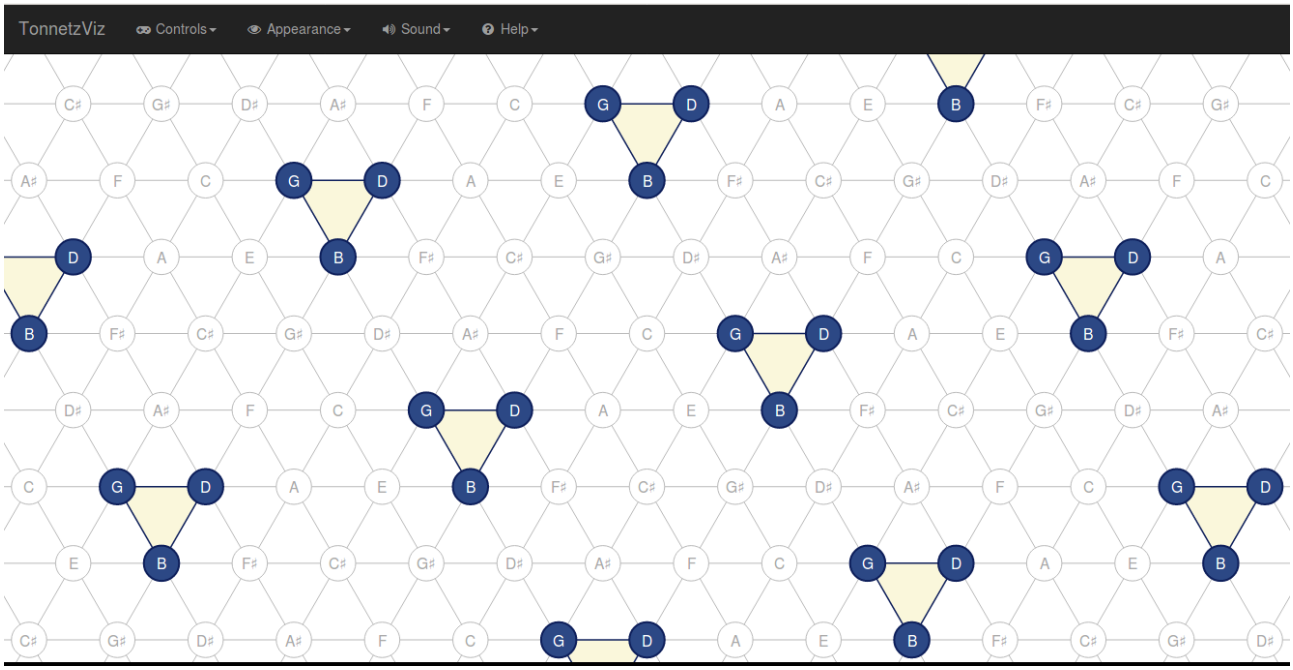
Deux applications existantes constituent le point de départ de ce projet.

3.1.Présentation de TonnetzViz

« TonnetzViz » est une application JavaScript en ligne permettant de jouer des notes en tapant au clavier et de voir leur représentation dans le Tonnetz en direct.

Ici, la représentation du Tonnetz dans l'application TonnetzViz :

Fig.6.TonnetzViz



3.2.Présentation de mudcube.js

L'application, « mudcube.js » permet de lire une pièce musicale sous la forme d'une chaîne de caractères (format base64), et de la jouer en générant les signaux MIDI correspondant, comme le ferait un lecteur multimédia classique.

Le format base64 est une façon d'encoder de l'information binaire dans une chaîne de caractères dans laquelle chaque caractère contient 6 bits d'information.

4.Objectifs du projet

L’objectif du projet proposé est de développer un prototype en JavaScript permettant la visualisation de fichiers au format MIDI dans le Tonnetz comme le fait Hexachord.

Le format midi est un format qui permet non seulement de lire une musique mais au aussi de connaître la date à laquelle jouer chaque note. Ce format rend ainsi l’analyse musicale assistée par ordinateur plus facile car il permet à la fois d’analyser le fichier dans son ensemble mais aussi accord par accord.

Le format midi possède également la particularité de fonctionner grâce à un mécanisme de ‘noteOn’ / ‘noteOff’ que l’on peut voir comme interrupteur qui fermerait ou ouvrirait un circuit. Lorsque le circuit est fermé, le courant passe et on joue la note (ce qui correspond à un évènement noteOn), et lorsque le circuit est ouvert et le courant arrête de passer, ce qui correspond à un évènement noteOff. Ce procédé permet de savoir à quel moment une note commence et à quel moment elle s’arrête.

Le format de fichiers midi est conçu pour pouvoir reproduire les sons de façon précise lors de la lecture une note au format midi contient donc sensiblement les mêmes informations sur une note qu’une partition :
-le type d’évènement (noteOn/noteOff)
-son timbre
-sa durée
-l’instrument associé à cette note

4.1.Premier objectif

Le premier objectif du projet consiste à élargir les fonctionnalités du visualiseur TonnetzViz afin de permettre à l'utilisateur de visualiser le déroulement de fichiers MIDI et non seulement de notes saisies de manière ponctuelle à l'aide du clavier. La lecture de fichiers MIDI se fera à l'aide de la bibliothèque midi.js employée par l'application « mudcube ».

4.2.Second objectif

La suite du projet consiste à ajouter à l'application un parseur de fichier MIDI permettant au visualiseur de se référer à une structure de données chargée en mémoire plutôt qu'à écouter les notes MIDI saisies à la volée au cours de la lecture du fichier MIDI. Cette fonctionnalité permettra notamment une navigation plus facile dans la pièce de musique (retour de x pas en arrière, en avant, etc.). La manipulation de la pièce musicale sous la forme d'une structure de données dans l'application ouvre par ailleurs d'intéressantes possibilités pour l'implémentation à venir de fonctionnalités permettant l'analyse automatique de la pièce musicale sans avoir à la lire à l'aide du lecteur.

4.3.Objectif secondaire

Un objectif supplémentaire consiste à développer un côté serveur de cette application pour pouvoir mettre en ligne l'application et y accéder de n'importe où et par tout le monde.

5.Enjeux du projet

Le fait de développer à nouveaux les fonctionnalités du logiciel HexaChord, pour une application, à l’aide de technologies plus modernes, notamment, le langage JavaScript, permet d’une part, de permettre à l’équipe de maintenir son application plus facilement en évitant de ressortir une nouvelle version de l’application sous forme d’un fichier au format (.jar) dès qu’une modification est apportée à l’application.

La simplicité d’accès (seul un navigateur est nécessaire) permettra d’autre part à l’application d’être utilisée plus facilement et de toucher ainsi un public plus large, notamment chez les musicologues et les étudiants en musique qui constituent le publique visé majoritairement.

Enfin, le développement de fonctionnalités côté serveur devrait permettre à terme l’équipe de récupérer un certain nombre de données utilisateurs qui pourront être utilisée dans un cadre d’apprentissage automatique.

6.Objectifs remplis

6.1.Premier objectif

Le premier objectif est rempli : les notes constituant une pièce musicale fournies sous la forme d'une chaîne de caractères écrite en dur dans un fichier de l'application (au format base64) sont affichées dans l'interface graphique de l'application TonnetzViz à chaque nouvel évènement.

6.2.Second objectif

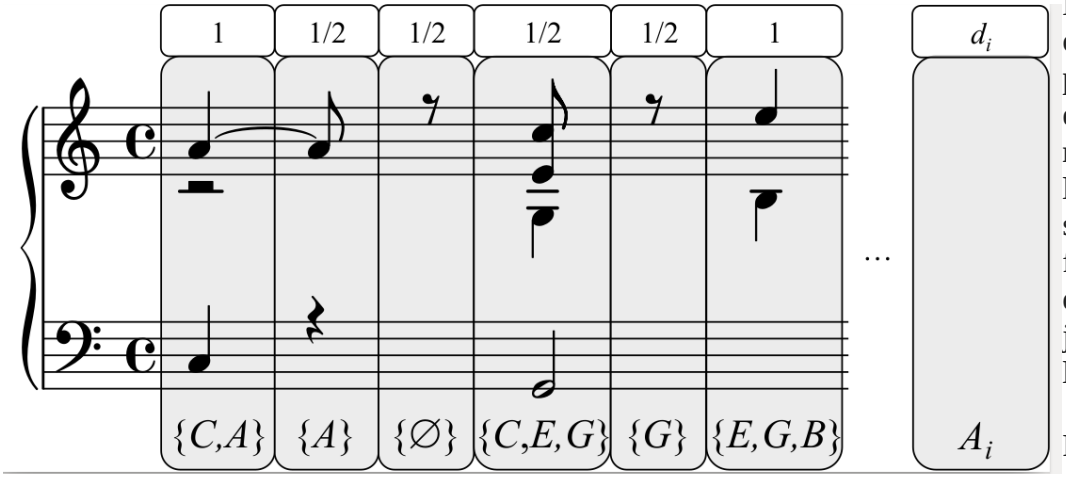


Fig.7

Le second objectif principal a également été rempli, l'utilisateur peut sélectionner un fichier midi qu'il souhaite jouer depuis l'application.

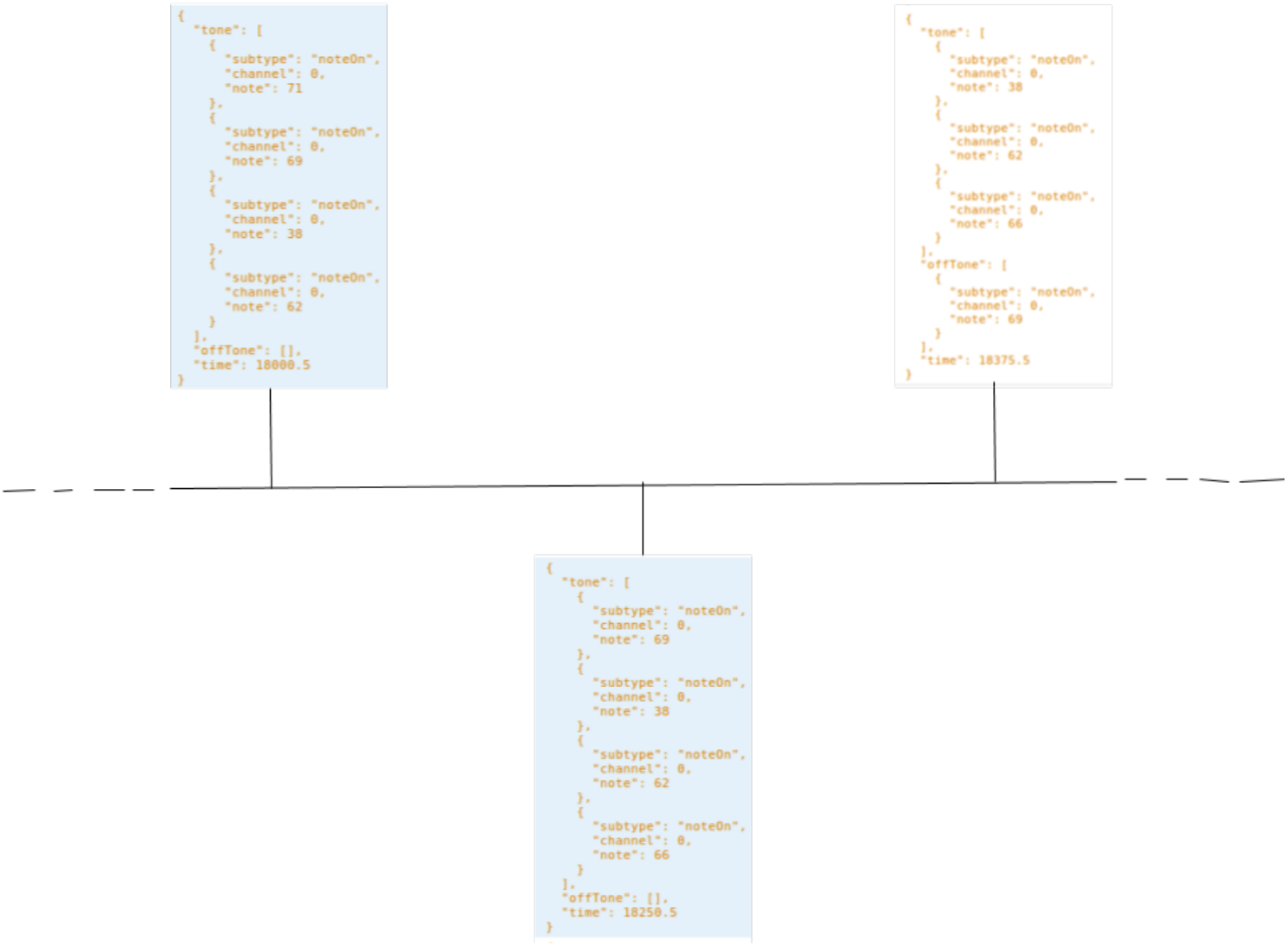
C'est ici qu'intervient la segmentation que nous avons introduite plus tôt. Cette segmentation sera effectuée non pas sur une partition mais sur un fichier midi qui est l'équivalent numérique de celle-ci.
Tout d'abord, le parseur s'occupera de récupérer les notes présentes dans le fichier puis un algorithme s'occupera de la segmentation de cette partition numérique.

Le résultat de la segmentation se présentera sous forme d'une structure de donnée dans laquelle chaque élément correspond à un accord, les éléments seront également rangés dans l'ordre chronologique pour faciliter leur accès lors de la lecture du fichier midi.

L'affichage se fait dorénavant grâce à cette structure de donnée correspondant à une fresque chronologique dans laquelle il est possible de naviguer.
Une fois cette fresque construite, il est possible d'interroger celle-ci pour afficher les notes dans le Tonnetz.

Cette fresque ressemble, une fois construite, à la figure suivante :

Fig.8.Représentation partielle de la segmentation d’une pièce musicale via l’application

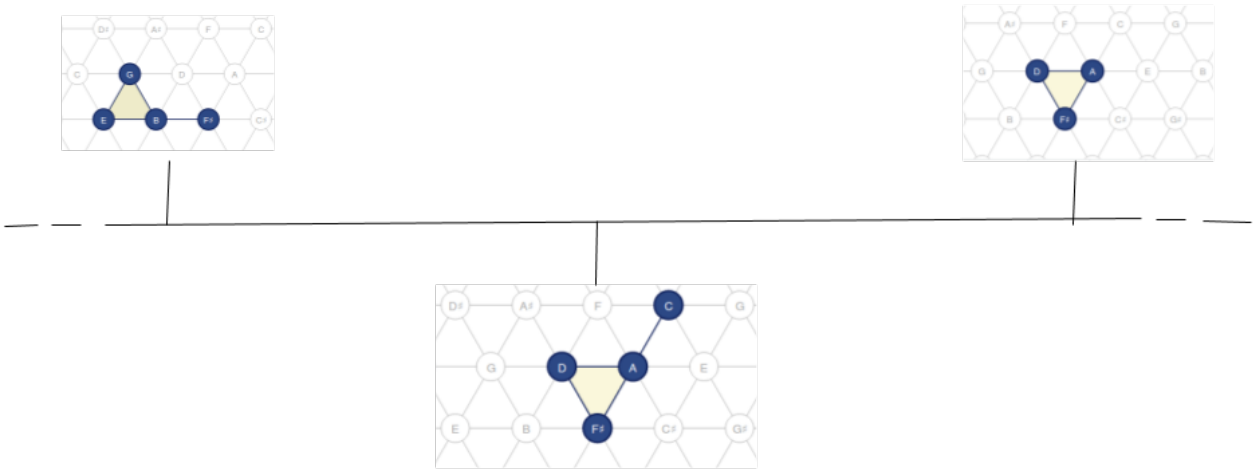


Ici on a représenté au format textuel, un exemple de trois accords se suivant tels qu’ils sont gardés en mémoire par l’application.
On a ici gardé, une version minimaliste de ce qui compose une note au format MIDI, c’est-à-dire, l’instrument par lequel elle est jouée, le type d’évènement (noteOn pour une note que l’on est en train de jouer et noteOff pour une note à laquelle on demande de s’interrompre).

A chaque fois que le lecteur de fichier midi de l’application ira lire une nouvelle note, on mettra à jour l’affichage grâce à cette structure.
La fonction qui a été implémentée pour lire un nouvel élément dans la fresque peut également servir à revenir en arrière et ré-afficher un accord que l’on vient d’entendre.

Comme nous l’avons vu précédemment, nous avons réalisé une segmentation de la pièce musicale (voir la figure 8) dans l’application qui nous a permis de décomposer la pièce musicale en une suite d’accords. Ceci nous permet maintenant d’afficher dans le Tonnetz chacun de ces accords.

Fig.9.Représentation de plusieurs accords dans le Tonnetz de l’application



6.3.Objectif secondaire

Nous avons également commencé à implémenter un serveur en Nodejs.
Pour l’instant nous avons seulement adapté l’interface de l’application.

7.Conclusion

Lors de ce projet, j’ai, dans un premier temps, eu l’occasion de pouvoir ré-utiliser deux applications applications, Mudcube.js et TonnetzViz, et faire coïncider les fonctionnalités intéressantes pour les musicologues de l’équipe Algomus. L’une est intéressante pour son lecteur de fichier midi et l’autre pour ce qu’elle apporte en terme d’analyse de la musique grâce à sa représentation de notes au format midi dans le Tonnetz.
Dans un second temps, j’ai pu implémenter un outil de segmentation de la pièce musicale au choix de l’utilisateur.

Cette application est pour le moment uniquement disponible en accès local, l’implémentation du serveur n’étant pas terminée.
Les prochaines étapes consisteront au développement d’un serveur, ce qui permettrait d’élargir encore les fonctionnalités de l’application et de rendre accessible cet outil à d’autres équipes de musicologues.
Cette solution permettrait également à l’équipe Algomus de pouvoir mettre à jour simplement l’application côté serveur sans avoir besoin de redistribuer l’application à chaque mise à jour.

Ce projet m’a permis de découvrir le langage JavaScript que j’avais envie d’apprendre à utiliser.
J’ai également commencé à utiliser Nodejs grâce à ce projet.
De plus, ce projet m’a permis d’élargir ma culture musicale, notamment sur des choses peu connues comme le Tonnetz, le format de fichier midi qui apporte beaucoup de possibilités, et sur la musicologie en générale.
Ce projet a donc été pour moi l’occasion de me familiariser avec des disciplines qui étaient nouvelles pour moi, ce qui m’a permis de vivre un projet enrichissant.

8.Remerciements

Je tiens à remercier M. Bigo pour avoir pris de son temps pour me conseiller tout au long de ce projet.

9.Lexique

Accord : ensemble de notes.

Consonance : combinaison de sons dite « agréable » à l’oreille.

Tonnetz : signifie littéralement « réseau tonnal » en allemand et permet de représenter la musique sous forme d’un graphe où chaque nœud est une note et les arêtes représentent les accords.
(cf. annexe)

MIDI : format de fichier musical dans lequel chaque note est représentée par une durée d’activité pendant laquelle la note sera jouée, une note qui permettra de savoir quelle note (fa, do, etc …) doit être jouée ainsi que le délai entre chaque note.

Hexachord : programme développé par l’équipe Algomus permettant de visualiser une musique de format MIDI dans le tonnetz.

10.Annexe

Fig. 1 : http://www.musictheory.net/lessons/10

Fig. 2 : https://fr.wikipedia.org/wiki/Tonnetz#/media/File:Speculum_musicae.png

Hexachord : https://www.youtube.com/watch?v=NQ7LkWCzKxI

Tonnetz : https://i.ytimg.com/vi/NQ7LkWCzKxI/hqdefault.jpg

TonnetzViz : https://cifkao.github.io/tonnetz-viz/

MIDI.js : https://galactic.ink/midi-js/