Exploration de partitions musicales modélisées sous forme de graphe

Exploring Music Scores as Graph Data

Philippe Rigaux [1], Virginie Thion [2]

[1]Conservatoire National des Arts et Métiers, CEDRIC Laboratory, France, Paris, France, Philippe.Rigaux@cnam.fr

[2]Univ. Rennes, CNRS, IRISA, Lannion, France, Virginie.Thion@irisa.fr

RÉSUMÉ. Depuis plusieurs siècles, la diffusion de la musique occidentale est assurée par la transcription des œuvres musicales sur des documents appelés *partitions musicales*. Pendant longtemps transmises sur un support physique, les partitions musicales peuvent aujourd'hui être encodées dans des formats numériques offrant une représentation fine de leur contenu. Ce nouveau média de représentation ouvre la voie à de nombreuses fonctionnalités reposant sur l'exploitation automatisée du contenu musical encodé, telles que la recherche, l'analyse, la classification ou la recommandation guidées par le contenu des partitions. Plusieurs formats standards peuvent actuellement être utilisés pour l'encodage des partitions musicales. Ils reposent en partie sur la représentation de la notation musicale en elle-même, intrinsèquement complexe et imbriquant des aspects relatifs au contenu et à la mise en forme de ce contenu. Les données contenues dans les partitions, même numérisées, restent donc difficiles à manipuler. Dans l'objectif de proposer un paradigme de représentation du contenu d'une partition musicale à la fois expressif et intuitif, nous proposons un modèle de données orienté graphe permettant de représenter le contenu purement musical de partitions numérisées s'abstrayant des aléas liés aux choix d'encodage et à la surcharge qui découle de l'imbrication des informations de contenu et de mise en forme. Après avoir présenté ce modèle, nous discutons des fonctionnalités d'interrogation offertes par ce type de représentation. Nous fournissons également une réalisation concrète du cadre formel proposé, sous la forme d'une implantation dans le système Neo4j, avec interrogation des données *via* le langage de requête Cypher.

ABSTRACT. *Sheet music scores* have been the traditional way to preserve and disseminate Western classical music works for centuries. Nowadays, their content can be encoded in digital formats that yield a very detailed representation of music content expressed in the language of *music notation*. These *encoded (digital) scores* constitute an invaluable asset for digital library services such as search, analysis, clustering, and recommendations. In this paper, we propose a model of the musical content of digital score as graph data, which can be stored in a graph database management system. We then discuss the querying of such data through graph pattern queries. We also describe a proof-of-concept of the approach that allows uploading music scores in a Neo4j database, and expressing searches and analyses through graph pattern queries with the query language Cypher.

MOTS-CLÉS. Partitions musicales, Base de données graphe, Modèle de données, Interrogation à base de patrons.

KEYWORDS. Music scores, Graph databases, Data model, Pattern queries.

Introduction

La sauvegarde des œuvres musicales constitue un enjeu patrimonial important. Bien que les fichiers audio soient aujourd'hui le principal canal de diffusion de la musique, celle-ci a été, pendant des siècles, préservée et disséminée sous la forme de partitions musicales imprimées. Les partitions musicales ont été – et sont encore dans de nombreux cas – la façon la plus précise de restituer les intentions d'un compositeur, et de transmettre, aussi fidèlement que possible, ces intentions aux interprètes.

La notation musicale est le formalisme le plus élaboré qui ait été développé pour décrire la musique au niveau symbolique [Ape61]. Elle est traditionnellement utilisée pour transcrire des œuvres musicales sous la forme de partitions musicales [Gou11]. Une partition musicale imprimée est un objet sémiotique complexe qui, dans une unique et compacte représentation, combine un encodage symbolique de la musique à produire, avec une visualisation sophistiquée favorisant une lecture efficace. La figure **1** est un extrait de partition musicale issue de l'un des corpus de plateforme NEUMA [RAA+12, Neu22]. Il s'agit

d'une œuvre composée de quatre voix, dont le contenu est représenté dans le système de la *notation musicale occidentale contemporaine*, formalisme sur lequel nous nous focalisons dans la suite de cet article.

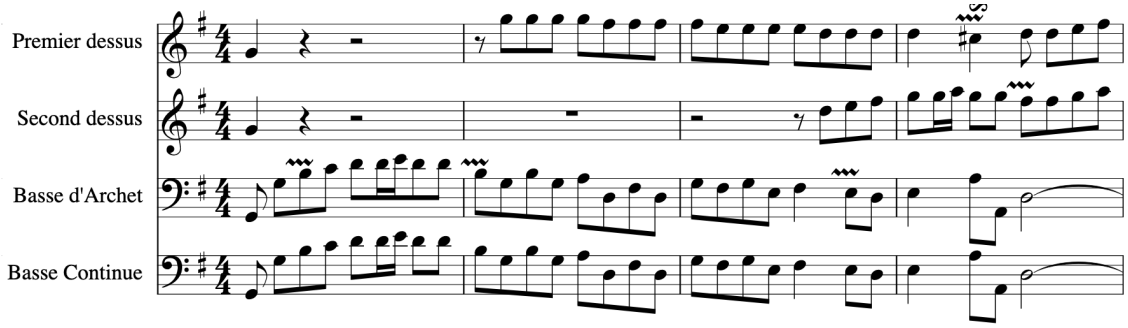


FIGURE 1.: Extrait de *La Françoise*, François Couperin (platforme NEUMA)

Pendant longtemps, les partitions musicales ont été transmises par copie ou impression sur un support physique comme le papier. Avec l'avènement de l'informatique, elles ont ensuite été numérisées sous forme d'images, facilitant ainsi leur diffusion par transmission numérique. Aujourd'hui, les partitions musicales peuvent être encodées dans des formats numériques, tels que le **kern format [Hur4] ou les formats semi-structurés fondés sur XML (MusicXML [Goo01], et ses successeurs MNX [MNX18], ou MEI [Rol02, MEI22]), offrant une représentation fine de leur contenu. Dans ces formats d'encodage, la notation musicale est vue comme un support permettant la structuration et le traitement de l'information musicale. Par exemple, **kern a été spécifiquement conçu pour encoder des partitions musicales en vue de leur traitement par les modules d'analyse musicale de l'outil Humdrum [Hur02].

De large collections de partitions numérisées sont aujourd'hui disponibles, non seulement grâce à un effort de long terme de numérisation et de capitalisation de partitions par des organismes institutionnels tels que des laboratoires de recherche [Sap05, RAA+12], mais aussi en raison de l'utilisation grandissante d'outils populaires d'édition de partitions musicales (par exemple MuseScore [Mus]).

Les bibliothèques numériques ont grandement évolué ces dernières décennies, et proposent des fonctionnalités allant au delà du simple stockage de données encodées. On attend maintenant d'une bibliothèque numérique qu'elle offre des services "intelligents" permettant une exploitation automatisée des données, au sein même de la bibliothèque. Les bibliothèques numériques de partitions musicales ont suivi cette évolution et visent à proposer des services exploitant le contenu musical lui-même, tels que la recherche par le contenu (par exemple, la recherche de patrons musicaux rythmiques, mélodiques ou harmoniques), la transformation du contenu musical (par exemple la transposition ou la conversion dans d'autres formats tels que le format MIDI), ou encore l'analyse du contenu (par exemple la recherche de patrons musicaux fréquents ou la découverte de défauts de qualité des données dans les partitions [FRT21]).

Ces nouveaux services reposent sur une représentation structurée de la *notation musicale* contenue dans les partitions encodées, sous forme d'objets qui peuvent être manipulés (sélectionnés, transformés) par des opérateurs de haut niveau. Les langages de manipulation de données traditionnels ne sont malheureusement pas bien adaptés au cas particulier de la manipulation de partitions musicales. Par exemple, les langages de manipulation de données dédiés aux modèles de données relationnels sont clairement trop peu expressifs, et les langages de manipulation de données dédiés aux modèles de don-

nées semi-structurés deviennent rapidement difficiles à utiliser dans le cadre des partitions musicales en raison de la complexité des opérations de navigation à mettre en œuvre pour parcourir les nombreuses données imbriquées contenues dans une partition. Ils sont aussi très dépendants de détails de choix d'encodage [FSRT16]. Une alternative est la manipulation de données par une approche fondée sur des paradigmes de programmation, telle que mise en œuvre dans la suite d'outils Music21 [CA10], qui permet la définition d'opérations très expressives, à la fois par leur richesse et leur finesse. La difficulté, à l'usage, de ce type d'approche est la nécessité pour l'utilisateur de posséder des compétences évoluées en programmation informatique, faisant défaut à la plupart des utilisateurs métier. Le problème de la manipulation de partitions musicales numérisées, c'est-à-dire la définition d'une structure de données et d'un langage permettant leur manipulation de façon à la fois expressive et intuitive, reste donc un problème ouvert.

En outre, un modèle de données attire particulièrement l'attention de nombreux chercheurs de la communauté des bases de données : celui de *modèle de données orienté graphe* [AG08, Ang12, WRE13, AAB+17, BFVY18]. La finalité première de ce type de modèle est de permettre une gestion efficace des données qui peuvent nativement être représentées sous forme de graphe, tels que tout type de réseau (social ou de transport), ou encore des données biologiques, cartographiques, topologiques, bibliographiques, *etc.* Aux modèles de données orientés graphe existants, sont associés des langages de manipulation de données, reposant sur la notion de *patron de graphe*. Dans un langage d'interrogation à base de patron de graphe, les données à retrouver sein des données sont décrites par la forme (la "silhouette") d'un sous-graphe à rechercher, ce qui constitue une vision assez intuitive de la notion de requête d'interrogation. Si l'on revient au sujet des partitions musicales, partant du principe que le contenu d'une partition musicale est composé d'informations diverses et complexes reliées entre elles (une note suit une autre note, une note appartient à une voix, et apparaît dans une mesure, une voix appartient à un instrument, ...), la représentation d'une partition musicale sous forme de graphe semble une approche pertinente en termes de modélisation, puis d'interrogation.

Dans la suite de cet article, nous proposons un modèle de données permettant de représenter le contenu musical d'une partition sous la forme d'un graphe comportant des propriétés. Cette modélisation permet le stockage de collections de partitions dans une base de données graphe, et la manipulation des partitions *via* un langage de requête à base de patron de graphe. Nous proposons également une implantation preuve de concept de l'approche, reposant sur un outil nommé MUSYPHER qui, à partir d'une partition musicale encodée dans un format standard, transforme la partition sous forme de graphe puis le charge dans une base de données graphe gérée par le système Neo4j [Neo22b]. Ce dernier fournit une interface permettant d'explorer et interroger les partitions à travers des requêtes à base de patron de graphe *via* le langage Cypher [Neo22a]. Les partitions utilisées pour illustrer la preuve de concept sont issues de la bibliothèque numérique NEUMA [RAA+12].

L'article est organisé comme suit. La section 1 présente la littérature traitant du problème de la modélisation du contenu musical d'une partition. La section 2 introduit le modèle de données orienté graphe que nous proposons. La section 3 aborde la notion d'interrogation, par des requêtes à base de patron de graphe, de données respectant le modèle proposé dans la section précédente. Une implantation illustrant le cadre théorique est présentée dans la section 4. Enfin, la section 5 conclut ces travaux et en dresse quelques perspectives.

1. Travaux connexes : modélisation du contenu musical d'une partition

La littérature propose trois types d'approche pour la modélisation du contenu musical d'une partition. La première, adoptée par les formats d'échanges standards, consiste à modéliser le contenu de la partition sous la forme d'un arbre, décomposant la partition musicale de façon "verticale". La seconde approche consiste à modéliser le contenu musical sous la forme d'un ensemble synchronisé de séries temporelles d'évènements musicaux se produisant sur une période de temps. Ce type de décomposition de la partition musicale peut être qualifié d'"horizontal". Enfin, une troisième approche proposée dans la littérature vise à représenter, sous la forme d'un graphe, les éléments rythmiques apparaissant dans la restitution visuelle d'une partition musicale imprimée, dans l'objectif d'améliorer les processus de reconnaissance optique de musique [1]. Nous présentons succinctement ces approches ci-dessous.

Découpage "vertical" – Modélisation sous forme d'un arbre. Les premiers formats standardisés d'encodage du contenu d'une partition musicale sont apparus sous l'impulsion de travaux académiques (MuseData and Humdrum), afin de répondre au besoin de l'interopérabilité des systèmes et outils dédiés à la manipulation des partitions, tels que les outils d'édition, les outils de reconnaissance optique de musique, ou les outils séquenceurs de musique. Les formats les plus répandus à ce jour sont MusicXML [Goo01, Mus22] et MEI [Rol02, MEI22]. Dans ces formats, le contenu d'une partition est encodé, incluant le contenu musical lui-même et les informations de restitution graphique. En respectant un dialecte XML, les données de la partition (notes, paroles) sont structurées sous la forme d'un arbre n-aire ordonné découpant le contenu musical par mesure, comme illustré dans la figure **2**. Bien que les

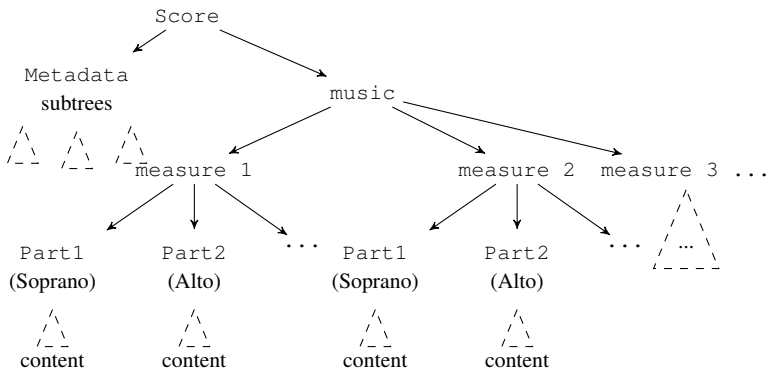


FIGURE 2.: Découpage typique d'une partition au format MEI

partitions encodées dans ces formats aient principalement vocation à être échangées entre outils, il est envisageable de chercher à les exploiter de façon *ad hoc* en utilisant des langages de requête génériques dédiés aux données semi-structurées tels que XPath ou XQuery [GSD08, FSRT18]. Mais les partitions encodées sous cette forme s'avèrent malheureusement complexes à appréhender pour un utilisateur en raison de la verbosité inhérente au formaliste XML, de l'indéterminisme de leur représentation (certaines informations peut être représentées par différentes variantes syntaxiques d'une partition encodée à une autre, voire même au sein d'une même partition), et parce qu'elles mêlent les informations du contenu musical lui-même avec des informations de restitution graphique. Cette complexité entrave la définition de langages de requête robustes dédiés à l'exploitation du contenu musical modélisé sous cette forme.

1. en anglais *Optical Music Recognition* (ou OMR).

Découpage "horizontal" – Modélisation sous forme de séries temporelles synchronisées. Un autre point de vue consiste à envisager le contenu musical comme étant un ensemble de *voix* (la notion de *voix* est le terme établi pour désigner une séquence d'événements musicaux non superposés). Chaque voix est représentée par une série temporelle d'évènements, où chaque évènement se produit sur un intervalle de temps. Les voix d'une partition sont synchronisées entre elles relativement à l'échelle de temps, commune à toutes les voix de la partition. L'algèbre temporelle *ScoreAlg* définie dans [FSRT18] est fondée sur cette approche. Une voix y est définie comme étant une fonction dont le domaine est un intervalle de temps, et le co-domaine est l'ensemble des évènements musicaux. La figure **3** est un exemple d'une telle fonction. Elle modélise le commencement de la voix *Basse continue* de la figure **1**. Les voix synchronisées sur l'échelle de temps commune forment une partition complexe. Dans [FSRT18], *ScoreAlg* fournit, sur la base de cette modélisation, une algèbre d'interrogation (un langage clos) fondée sur des opérateurs de manipulation des partitions. Ce type de langage est très expressif mais nécessite une abstraction formelle complexe, peu intuitive pour la plupart des utilisateurs.

$$v_{basse\ cont.}(t) = \begin{cases} G2_0^4, & t \in [0,4[\\ G3_4^8, & t \in [4,8[\\ B3_8^{12}, & t \in [8,12[\\ C4_{12}^{16}, & t \in [12,16[\\ D4_{16}^{20}, & t \in [16,20[\\ D4_{20}^{22}, & t \in [20,22[\\ E4_{22}^{24}, & t \in [22,24[\end{cases}$$

FIGURE 3.: Voix modélisée sous la forme d'un ensemble de séries temporelles d'évènements musicaux

Modélisation d'élément rythmiques sous forme de graphe. Dans l'objectif d'améliorer les processus de reconnaissance optique de musique, les auteurs de [JR15] ont proposé une approche visant à modéliser les éléments qui constituent les marqueurs rythmiques d'une partition imprimée. Le rythme est représenté sous la forme d'un graphe dans lequel les nœuds représentent des occurrences de symboles graphiques pertinents d'un point de vue rythmique (note, silence, barre de mesure). Chaque symbole a une durée associée (par exemple, une croche pointée a une durée de *3/16* et une barre de mesure a une durée de *0*). Les nœuds sont connectés, ou bien selon leur ordre d'occurrence au sein d'une voix (un symbole suit un autre symbole), ou bien par des relations de concomitance (des symboles de voix différentes alignés temporellement dans la restitution de la partition). Ce modèle de données se concentre sur l'alignement des symboles graphiques dans l'objectif de raffiner le processus de reconnaissance optique des symboles musicaux.

Le modèle de données orienté graphe que nous proposons dans la suite de cet article s'inscrit dans la veine des travaux présentés ci-dessus. On y vise à abstraire le contenu musical seul des partitions musicales (contenu des partitions exempt des méta-données et des informations de restitution) afin d'offrir une représentation à la fois robuste, expressive et intuitive du contenu des partitions, sous forme de graphe. Les principaux objectifs de cette représentation sont (i) de proposer un modèle de données expressif qui concilie et étend, au sein d'une même représentation, les deux points de vue de découpage "vertical" et "horizontal" du contenu d'une partition et (ii) d'offrir à un utilisateur la possibilité de définir des requêtes *ad hoc* dans un formalisme intuitif reposant sur la notion de patron de graphe.

2. Modélisation du contenu musical sous forme de graphe

Dans cette section, nous présentons le modèle de données que nous proposons, permettant de modéliser le contenu musical d'une partition musicale. Le modèle repose sur la notion de *graphe attribué* et intègre, dans un même modèle de données, les deux points de vue de la littérature consistant à voir le contenu musical comme structuré selon le rythme (découpage "horizontal" de la partition), ou comme une série temporelle d'évènements musicaux (découpage "vertical" de la partition). Avant de décrire le modèle en lui-même, il convient de présenter la notion de modèle de *graphe attribué*.

2.1. *Notion préliminaire : modèle de* graphe attribué

Dans un système de gestion de bases de données graphe, aussi bien la structure du schéma que les instances sont modélisées par des graphes, et la manipulation des données se fait au moyen d'opérations orientées graphe et de constructeurs de type [AG08, Ang12, WRE13, AAB$^+$17]. Il existe différents modèles de bases de données graphe (voir [AG08] pour une vue d'ensemble), incluant celui de *graphe attribué* [AAB$^+$17, BFVY18]. Dans ce modèle, les données sont représentées par un réseau d'entités, sous la forme d'un graphe dans lequel les nœuds représentent les entités et les arcs représentent des relations entre ces entités. Chaque nœud ou arc peut encapsuler des *propriétés* (aussi appelées *attributs*) sous la forme de l'ensemble d'éléments de type clef-valeur.

Passons à la définition formelle de *graphe attribué*. En termes de vocabulaire, on suppose l'existence des ensembles suivants disjoints : un ensemble $\mathcal{V}$ de *nœuds*, et un ensemble $\mathcal{E}$ d'arcs. On considère également l'existence d'un ensemble *Lab* d'étiquettes, d'un ensemble *Prop* de clefs de propriété, et d'un ensemble *Val* de valeurs de propriétés.

Definition 2..1 (Graphe attribué)**.** *Un graphe attribué $\mathcal{G}$ est un n-uplet $(V, E, \rho, \lambda, \sigma)$ où*

- *$V \in \mathcal{V}$ est un ensemble fini de nœuds ;*
- *$E \in \mathcal{E}$ est un ensemble fini d'arcs ;*
- *$\rho : E \to (V \times V)$ est une fonction totale qui attribue un couple de nœuds (origine et destination) à chaque arc (où $\rho(e) = (n_1, n_2)$ indique que e est un arc allant de n_1 à n_2) ;*
- *$\lambda_V : V \to \mathcal{P}(Lab)$ est une fonction partielle qui attribue un ensemble d'étiquettes à chaque nœuds de V ; et $\lambda_E : V \to Lab$ est une fonction partielle qui attribue une étiquette à chaque arc de E ; (sans ambiguïté, λ renvoie soit à λ_V soit à λ_E selon le domaine de l'élément auquel la fonction s'applique).*
- *$\sigma : (V \cup E) \times Prop \to Val$ est une fonction partielle qui attribue des couples clef-valeur aux élément de V et E (où $\sigma(n, p) = v$ (resp. $\sigma(e, p) = v$) indique que le nœud n (resp. l'arc e) a la propriété p de valeur v).*

Dans la suite, on considère un graphe attribué $\mathcal{G} = (V, E, \rho, \lambda, \sigma)$.

La notion de *chemin* s'applique au sein d'un graphe attribué comme elle s'applique classiquement au sein d'un graphe.

Definition 2..2 (Chemin)**.** *Un chemin p dans $\mathcal{G}$ est une séquence $n_1\, l_1\, n_2\, l_2\, n_3 \ldots n_{k-1}\, l_{k-1}, n_k$ avec $k \geq 1$, où $\{n_1, ..., n_k\} \subseteq V$ et $\{l_1, ..., l_{k-1}\} \subseteq Lab$ et chaque (n_i, l_i, n_{i+1}) t.q. $i \in \{1, ..., k-1\}$ est un arc de*

$\mathcal{G}$.[2] *Un tel chemin p relie le nœud n_1 au nœud n_k, et sa* longueur *est $|p| = k - 1$. Toute sous-séquence non-vide de p est un* sous-chemin *de p. Le chemin p est un cycle ssi $k \geq 2$ et $n_1 = n_k$. Le chemin p est cyclique ssi l'un de ses sous-chemins est un cycle (sinon p est acyclique).*

Definition 2..3 (Paths)**.** *Soit n_1 et n_2 deux nœuds de V. On note $Paths(n_1, n_2)$ l'ensemble des chemins acycliques qui relient n_1 à n_2.*

Les concepts généraux posés, nous passons passe maintenant à la définition du modèle de données permettant de représenter le contenu d'une partition musicale. Ce modèle de données orienté graphe intègre la modélisation de structure rythmique hiérarchique d'une partition proposée dans [ZFSRT22]. Intuitivement, on modélise l'information musicale comme mettant en correspondance un domaine temporel structuré avec un ensemble de *faits musicaux*. Le domaine temporel prend la forme d'une structure hiérarchique, appelée *arbre rythmique*, qui découpe un intervalle fini de temps en intervalles de temps disjoints. Chaque intervalle correspond à une feuille de l'arbre rythmique, mis en correspondance avec un *fait musical*. Les intervalles et les faits musicaux constituent des évènements musicaux, c'est-à-dire des faits musicaux ayant lieu pendant un intervalle de temps donné. Dans la suite de cette section, nous détaillons cette représentation, en commençant par définir la notion de fait musical.

2.2. *Domaine des faits musicaux*

Afin d'illustrer les concepts introduits, on s'appuie sur l'extrait de partition musicale donné en figure **4**, issue de l'hymne allemand *Das Lied der Deutschen*, composé par *Joseph Haydn* en 1797 [Hay97].

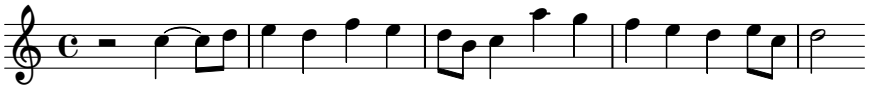


FIGURE 4.: Premières notes de *Das Lied der Deutschen*, de *Joseph Haydn* (1797).

Commençons par nous intéresser à la notion de fait musical (atomique).

Différents domaines de faits peuvent être considérés. Nous nous focalisons ici sur les sons simples. Un son peut être caractérisé par de nombreuses propriétés telles que son intensité, son timbre, sa fréquence. Dans le langage de la notation musicale, on utilise un ensemble fini de fréquences, aussi appelées hauteurs, pour qualifier l'ensemble des sons qui peuvent être utilisés dans une pièce de musique. Nous nous appuyons sur la norme [Int75] instituée par l'Organisation Internationale de Normalisation (ISO) pour énumérer les hauteurs de sons. La fréquence d'un son y est désignée par un nom (ou *classe de fréquence*) P (une lettre A, B, C, D, E, F, ou G), un indice désignant une octave I parmi $[1, 7]$, et une éventuelle altération a parmi $\{\sharp, \flat\}$. Un son est ainsi modélisé par un triplet $\{class : P, octave : I, accidental : a\}$, que l'on peut représenter sous la forme d'un symbole $P[a]I$. La modélisation d'autres propriétés pertinentes (timbre, intensité) peut être effectuée de façon triviale par extension de cette structure.

Graphiquement, dans les partitions, les niveaux de fréquence des sons sont matérialisées sur des lignes horizontales (des portées). Un son est représenté par un symbole graphique de *note* comportant une tête

2. (n_i, l_i, n_{i+1}) est un arc de $\mathcal{G}$ signifie qu'il existe $e \in E$ tel que $\rho(e) = (n_i, n_{i+1})$ et $\lambda(e) = l_i$.

(noir ou blanche), dont le positionnement vertical (la *hauteur*) sur la portée désigne le nom (la classe de fréquence) et l'octave du son. En guise d'illustration, le premier son dans la partition de la figure **4** est un son $C5$, suivi par un son $D5$, lui même suivi par un son $E5$, etc.

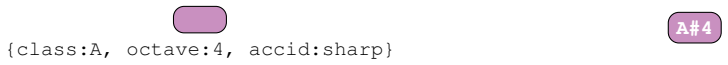
La musique comporte également des silences. Par exemple, la partition de la figure **4** commence par un silence, représenté graphiquement par un petit rectangle noir posé en appui sur la troisième ligne de la portée. Le domaine de faits musicaux atomiques comporte donc aussi le symbole de *silence* [3] noté r.

Dans la notation musicale conventionnelle, deux sons adjacents peuvent être "liés", composant un seul son dont la durée est égale à la somme des durées des sons liés. Une *liaison* est graphiquement représenté par un arc reliant les têtes des notes concernées. Dans la figure **4**, les deux premières notes sont liées. De façon à pouvoir modéliser la continuation de son, le domaine des faits musicaux comporte un *symbole de continuation*, noté _.

Definition 2..4 (Domaine des faits musicaux)**.** *Le domaine $\mathcal{F}_{\mathcal{M}}$ des* faits musicaux *est composé de : (1) l'ensemble des faits de type son, chaque fait de ce type étant représenté par un triplet $P[a]I$, où P est un nom de note (*"pitch class"*), a une altération optionnelle, et $I \in [1, 7]$ est une octave ; (2) le fait de type silence `r`; (3) le fait de continuité, noté _, utilisé pour modéliser des liaisons ou des continuations de son (c.-à-d. un son unique dont la représentation nécessite plusieurs symboles).*

Un son de hauteur $A\sharp 4$, c'est-à-dire un son *La* de l'octave *4* avec une altération *dièse*, est un exemple de fait musical.

Dans la modélisation sous forme de graphe que nous proposons, un fait est représenté par un nœud portant un ensemble de *propriétés* (ou *attributs*) de type clef-valeur. La figure **5**.(a) est un nœud de fait musical qui représente le fait $A\sharp 4$. Le nœud a trois propriétés : une propriété nommée `class` de valeur `A`), une propriété nommée `octave` de valeur `4`), et une propriété nommée `accid` de valeur `sharp`, qui modélise l'altération dièse ♯). Pour des raisons de lisibilité, un nœud de fait pourra être représenté par sa représentation compacte faisant apparaître les valeurs de ses propriétés fondamentales au sein du nœud lui-même, tel qu'illustré dans la figure **5**.(b)).



(a) nœud et ses propriétés (b) representation compacte

FIGURE 5.: Nœud représentant un fait musical

2.3. *Organisation temporelle des faits*

Une pièce de musique est une organisation temporelle de sons se produisant sur une plage de temps bornée. Les faits musicaux surviennent sur des ensembles de positions temporelles qui résultent d'une décomposition récursive d'intervalles temporels, dont découle une organisation rythmique intrinsèquement hiérarchique.

Dans la notation musicale occidentale, une pièce de musique est divisée temporellement en *mesures* (graphiquement, les mesures sont délimitées par des barres verticales apparaissant sur les portées, comme

3. *rest* (en).

représenté sur la figure **4** qui comporte cinq mesures), et une mesure se déroule sur un ou plusieurs *temps*. Les temps peuvent eux-mêmes être subdivisés en fragments de tailles égales, qui peuvent encore être subdivisés, etc. Ces divisions récursives du temps amènent à générer une organisation hiérarchique du temps qui prend la forme d'une structure arborescente. La *signature rythmique* (ou *chiffrage de mesure*), définie par un nombre rationnel (dans notre exemple, $4/4$ dénoté par $\mathbf{c}$) indique le type et la durée des temps dans les mesures qui suivent, déterminant la décomposition privilégiée. Une mesure de type $4/4$ consiste en 4 temps, la durée de chaque temps étant le quart de la mesure complète (graphiquement, une note noire ♩). Ainsi, dans le contexte d'une signature rythmique en $4/4$, la décomposition privilégiée est de 4 sous-intervalles,[4] les temps sont divisés en deux intervalles de 1/8e de mesure (graphiquement, une note croche ♪), elles-mêmes chacune divisée en deux intervalles de 1/16e de mesure (graphiquement, une note double-croche ♬), etc. Les autres signatures rythmiques (e.g., $3/4$, $6/8$), respectent des patrons de décompositions différents mais, dans tous les cas, leur découpage rythmique peut être représenté sous la forme d'une structure arborescente. Cette structure arborescente est dite *arbre rythmique*.

Le découpage récursif du temps structuré en *arbre rythmique* [ZFSRT22] est fondé sur les principes suivants : i) la plage de temps d'une pièce est divisée en mesures de taille égales, ii) chaque mesure est récursivement divisée selon des règles métriques, et iii) chaque feuille de l'arbre généré correspond à un fait musical qui survient pendant le sous-intervalle de temps défini par la position de la feuille.

Considérons de nouveau l'extrait de partition musicale donnée en figure **4**. La figure **6** illustre la structure rythmique associée au début du contenu musical de la figure **4**. Les quatre premiers niveaux de l'arbre (en bleu) composent l'arbre rythmique. La racine (s) correspond à la pièce de musique complète. Le niveau suivant est composé de nœuds découpant la pièce au niveau des mesures. Chaque mesure est elle-même découpée en temps, demi-temps, quart de temps, ou encore plus finement selon la finesse de décomposition temporelle nécessaire pour porter les faits musicaux. La première me-

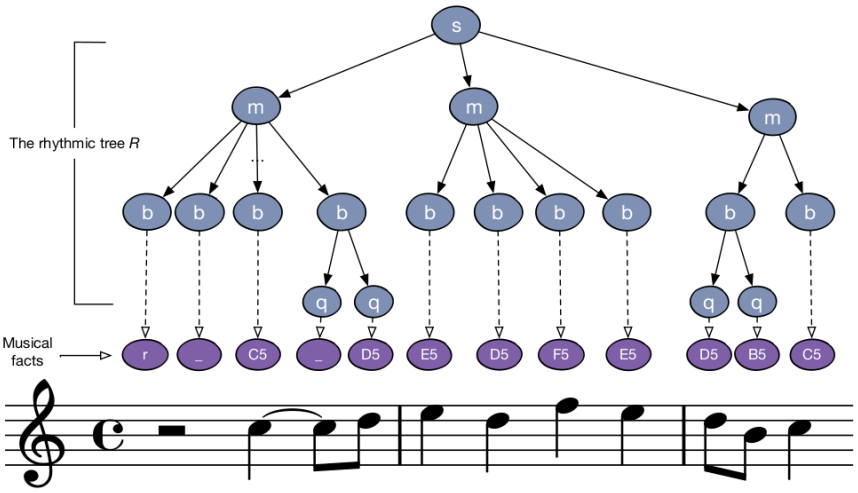


FIGURE 6.: Modélisation des premières notes de l'hymne allemand, avec son arbre rythmique et ses faits associés

sure, par exemple, est décomposée en quatre temps, le dernier temps étant lui-même décomposé en deux demi-temps. Les durée et position temporelle de chaque nœud peuvent facilement être déterminées en fonction de l'arbre rythmique. En effet, si N est un nœud couvrant un intervalle $I = [\alpha, \beta[$,

4. bien que moins susceptibles d'être envisagés, d'autres découpages sont possibles.

avec les sous-arbres $(N_1, \cdots, N_n)$, alors l'intervalle $itv_I(N_i)$ couvert par chacun des sous-arbres N_i est $[\alpha_N + (i-1) \times s, \alpha_N + i \times s[$ où $s = \frac{\beta_N - \alpha_N}{n}$ est la taille de chaque sous-arbre (voir [ZFSRT22] pour les détails et définitions formelles).

Dans la suite, on adopte la convention suivante pour représenter les valeurs temporelles : la durée d'une mesure est de 1, s'étendant sur un intervalle de $[0, 1[$, et la plage de temps d'une pièce de musique est $I = [0, n[$, où n est le nombre de mesures qui composent la pièce. La durée et l'intervalle d'un nœud découlent de la structuration de l'arbre rythmique de la pièce. Si le chiffrage de mesure est $4/4$, alors la durée d'une note blanche est $\frac{1}{2}$, la durée d'un temps est $\frac{1}{4}$, etc.

Cette structure constitue une partie de la modélisation du contenu d'une seule *voix* d'une partition sous forme d'un graphe.

Definition 2..5 (Voix)**.** *Une* voix *est un couple $(R, \mathcal{M})$, où R est un arbre rythmique et $\mathcal{M}$ met en correspondance les éléments de R avec des faits musicaux $\mathcal{F}_\mathcal{M}$.*

2.4. *Les évènements musicaux*

Chaque feuille de l'arbre rythmique est liée à un fait musical (feuilles violettes de l'arbre dans la figure **6**). Le premier temps de la première mesure est un fait de type silence, le deuxième temps est un fait de continuité (il poursuit le silence précédent), le troisième temps est un fait de type son ($C5$). Le quatrième temps est divisé en deux faits : une continuité du fait $C5$, puis un fait $D5$. Comme chaque feuille l de l'arbre rythmique couvre un intervalle $itv(l)$, on obtient des évènements musicaux sous la forme de couples $(itv(l), fact(l))$ qui associent un fait musical à un intervalle de temps.

Definition 2..6 (Évènement musical)**.** *Soit I un intervalle temporel, et $V = (R, \mathcal{M})$ une voix. Si l est une feuille de R, alors le couple $(itv_I(l), \mathcal{M}(l))$ est un* événement musical.

Revenons à la figure **6**. Le premier évènement musical est (I_1, r), avec $I_1 = [0, \frac{1}{4}[$ (premier temps de la première mesure). Le deuxième évènement est $(I_2, \_)$, avec $I_2 = [\frac{1}{4}, \frac{2}{4}[$ (continuité du silence précédent). Le troisième évènement est $([\frac{2}{4}, \frac{3}{4}[, C5)$, le quatrième évènement est $([\frac{3}{4}, \frac{7}{8}[, \_)$, etc.

Toute partie à jouer d'un instrument apparaissant dans une partition peut être modélisé sous forme d'un ensemble de *voix* (au sens large), chaque voix peut être structurée sous la forme d'un arbre rythmique.

2.5. *Représentation d'une voix sous forme de graphe*

Intuitivement, la modélisation sous forme de graphe consiste à représenter, au sein d'un même modèle de données, les deux points de vue de découpage vertical et horizontal de la partition. Cette représentation est obtenue de la façon suivante :

1. Les nœuds et arcs de l'arbre rythmique peuvent trivialement être représentés sous la forme d'un graphe. Les feuilles de l'arbre rythmique portent chacune deux propriétés `-start` (début) et `end` (fin)- représentant l'intervalle temporel du nœud de l'arbre. Les arcs issus de la modélisation de l'arbre rythmique n'ont pas nécessité à être orientés.
2. Si n_1 et n_2 sont deux frères adjacents dans l'arbre rythmique (par exemple deux nœuds de mesure frère) alors n_1 et n_2 sont reliés par un arc (orienté) allant de n_1 à n_2.

3. Si l_1 et l_2 sont deux feuilles adjacentes dans l'arbre rythmique alors l_1 et l_2 sont reliées par un arc r (orienté) allant de l_1 à l_2, et r porte une propriété `duration` (durée) dont la valeur est $itv_I(l_1)$, à savoir la durée de l_1 (voir page 10). [5]

Le point 1 modélise la vision consistant à représenter une voix par sa structure rythmique (découpage vertical de la partition). Les points 2 et 3 modélisent la vision consistant à représenter une voix comme étant une série temporelle d'événements musicaux (découpage horizontal de la partition).

4. Le contenu d'une voix peut être atteint, à partir d'un nœud "point d'entrée de la voix" , ou bien par l'arbre rythmique (*rhythmic tree perspective*), ou bien par un accès au premier évènement musical qui compose la voix (*time series perspective*). Toutes les informations pertinentes qualifiant une voix en tant que telle (par exemple l'instrument auquel est associée la voix) peuvent être attachées au nœud de point d'entrée de la voix sous la forme de propriétés.

Dans la suite, les arbres rythmiques se limiteront à la représentation des trois niveaux racine, mesure et évènement.

À partir de la voix de la figure **6**, en se limitant à la représentation de l'arbre rythmique ne comportant que le découpage rythmique au niveau des mesures, on obtient le graphe de la figure **7**.

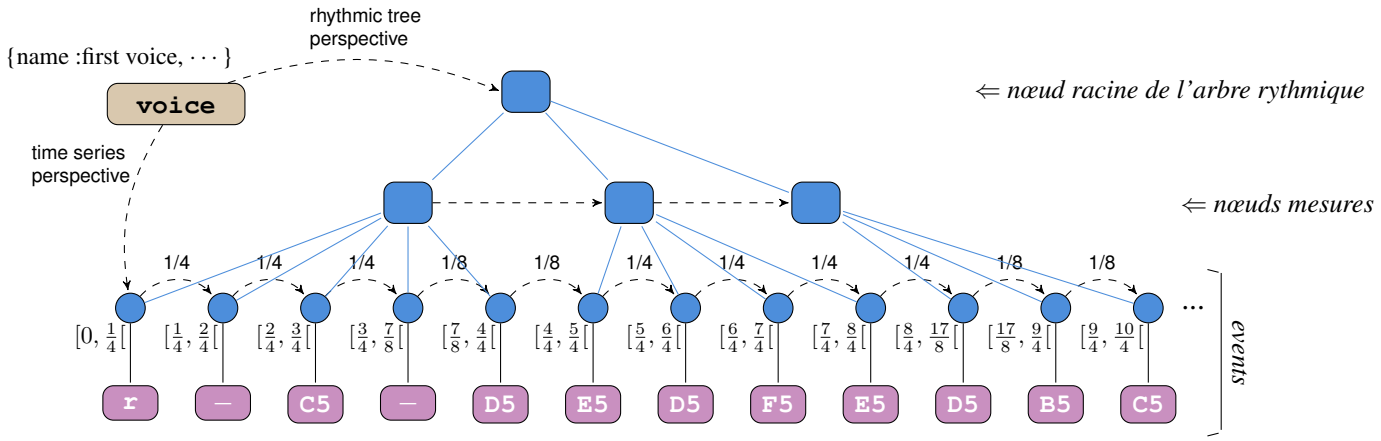


FIGURE 7.: Représentation d'une voix sous forme de graphe

Il est possible, pour faciliter l'interrogation des données, d'étiqueter les nœuds et les arcs (ensemble Lab associé à la fonction λ dans la définition 1). En particulier, afin de faciliter leur identification, il est possible d'étiqueter les nœuds modélisant des mesures par une étiquette `Measure`, étiqueter les nœuds points d'entrée des voix par une étiquette `Voice`, et étiqueter les nœuds de faits musicaux par une étiquette `Fact`. Il est également possible d'ajouter des étiquettes permettant d'identifier les natures des faits musicaux, par exemple `Note` ou `Rest` (note ou silence). Sur le même principe, les arcs peuvent être étiquetés pour représenter la nature de la relation exprimée.

2.6. *Modalités de représentation de la musique polyphonique*

Une pièce de musique dite *polyphonique* consiste en un ensemble de voix $\{V_1, V_2, \cdots, V_n\}$. La modélisation de ce type de pièce repose sur la modélisation de chacune de ses voix, rattachées à un nœud

5. Toute série comporte un évènement supplémentaire "spécial" de fin de série, permettant de modéliser une dernier arc porteur de la durée du dernier évènement d'une voix.

global qui représente la pièce de musique dans son ensemble (un exemple de ce type de modélisation sera donné dans la section suivante). Toutes les voix ont nécessairement le même découpage d'arbre rythmique de la racine au niveau des mesures inclus. La modélisation de la structure rythmique assure donc la synchronisation des différentes voix. Les arbres rythmiques des différentes voix diffèrent ensuite en fonction du contenu de chaque voix.

En termes de modélisation, deux choix sont possibles pour représenter un contenu musical d'une pièce composée de plusieurs voix.

Première option - Duplication des arbres rythmiques. On peut choisir de dupliquer la partie commune des arbres rythmiques des voix, avec la contrainte suivante : les sous-graphes correspondant aux découpages d'arbres rythmiques jusqu'au niveau des mesures inclus est le même pour toutes les voix d'une même pièce. Cette contrainte assure la synchronisation des voix au niveau des arbres rythmiques.

Seconde option - Factorisation des parties communes des arbres rythmiques. On peut aussi choisir de factoriser les parties communes des arbres rythmiques, en représentant le découpage commun de l'arbre rythmique jusqu'au niveau des mesures inclus (les voix sont synchronisées de façon inhérente au niveau de leur arbre rythmique puisqu'elles partagent une partie commune d'arbre rythmique jusqu'au niveau des mesures).

Dans la suite, nous avons choisi de factoriser les parties communes des arbres rythmiques (seconde option). Le graphe **9** (à venir, page 13) est un exemple de ce type de représentation modélisant deux voix qui partagent le même arbre rythmique pour la partie de l'arbre allant de la racine aux mesures.

La modélisation du contenu des partitions musicales sous la forme du modèle de données proposé dans la présente section permet leur encodage numérique respectant un format expressif et intuitif, dont l'un des principaux intérêt est d'ouvrir la voix à l'interrogation du contenu par un langage à base de patron de graphe, par exploration des données selon l'un ou l'autre des points de vue de découpage de la partition, ou alliant les deux points de vue à la fois. Ce sujet est abordé dans la section suivante.

3. Interroger des partitions musicales encodées sous forme de graphe

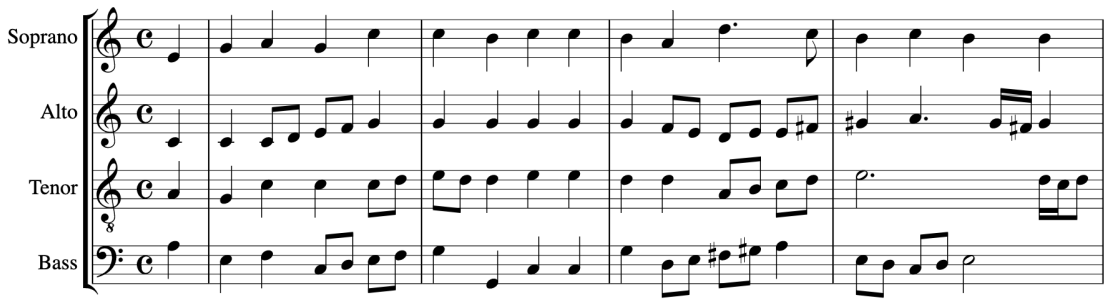


FIGURE 8.: Cantate BWV111 (J.S. Bach), 6e mouvement

La modélisation de contenu d'une partition musicale sous la forme d'un graphe permet l'exploration de ce contenu par le paradigme dit de l'interrogation à base de *patron de graphe*. De façon à illustrer cette notion, nous considérons l'extrait de partition de musique (polyphonique) de la figure **8**, nommée

BWV111 dans la suite. Cette partition contient quatre voix : *Soprano*, *Alto*, *Tenor* et *Bass*. Cinq mesures apparaissent sur la figure 8. Elles seront numérotées de *0* à *4*.

Le graphe de la figure 9, nommé $\mathcal{G}_{\text{BWV111}}$ dans la suite, est la représentation des quatre premières mesures des voix *Soprano* et *Alto* de BWV111 sous la forme d'un graphe respectant la modélisation introduite dans la section 2. La partie du modèle représentant l'arbre rythmique apparaît en couleur bleue. Les nœuds représentant des faits musicaux apparaissent en couleur rose. Les nœuds structurant les voix `Soprano` et `Alto`, points d'entrée vers la représentation du contenu des voix sous forme d'une suite (série) d'évènements musicaux apparaissent à la gauche de la figure (couleur marron).

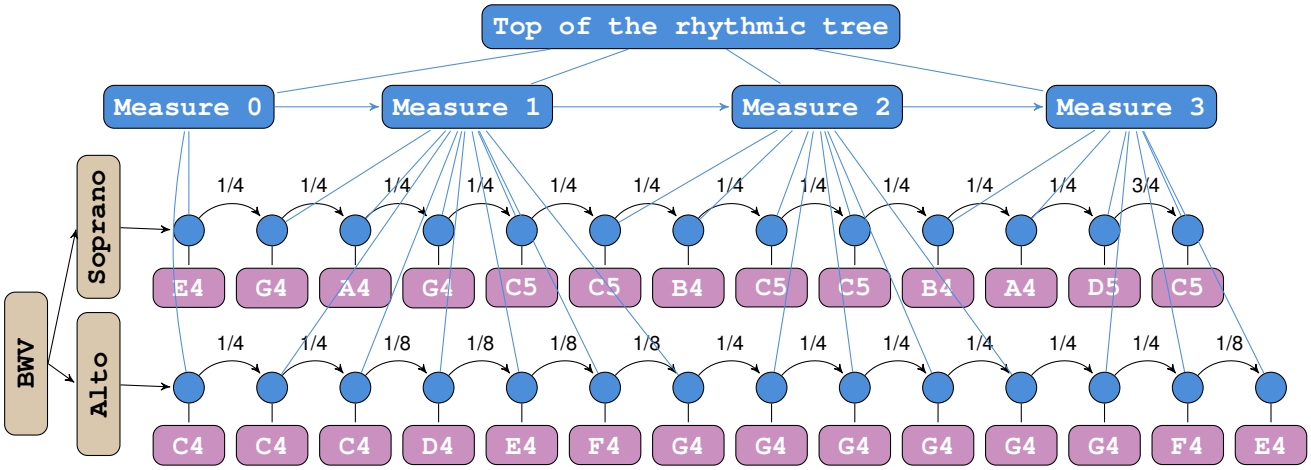


FIGURE 9.: ($\mathcal{G}_{\text{BWV111}}$) Représentation de BWV111 sous la forme d'un graphe respectant le modèle de données introduit dans la section 2.

On s'intéresse maintenant à l'exploration d'une partition musicale représentée sous cette forme, à l'aide de *requêtes à base de patron de graphe*.

Formellement, un patron de graphe est un graphe dans lequel des variables et des conditions peuvent apparaître [Bar13, AAB+17].

Definition 3..1 (Patron de graphe – syntaxe)**.** *Soit Var_{nodes} et Var_{lab} deux ensembles distincts de variables de nœud et d'arc respectivement. Une* requête à base de patron de graphe *est un graphe attribué de la forme $(V, E, \rho, \lambda, \sigma, Cond_n, Cond_e)$ dans lequel des variables de nœud et d'arc peuvent apparaître* [6]*, et $Cond_n$ (resp. $Cond_e$) sont des conditions Booléennes portant sur les valeurs des propriétés des éléments de V (resp. E).* [7]

Intuitivement, un patron de graphe définit la forme d'un sous-graphe à retrouver dans les données. Dans sa forme la plus simple un patron de graphe définit un chemin. Les figures 10, 11 et 12 sont des exemples de tels patrons. Dans ces patrons, les nœuds et les arcs sont des variables, et des conditions sont ajoutées aux éléments composant le patron.

Le patron de la figure 10 vise à rapatrier les occurrences de deux évènements suivant une note $C4$. Dans ce patron, `Fact1` et `Fact2` sont des variables de nœud. Des conditions sont associées aux nœuds :

6. formellement, tel que $V \in \mathcal{V} \cup Var_{nodes}$ and $\lambda : (V \cup E) \rightarrow Lab \cup Var_{lab}$.
7. p.e., si f_1 est une variable de nœud alors une condition peut être $f_1.class = E \wedge f_1.octave = 4$, qui indiquerait dans notre contexte que la variable f_1 doit être un son $E4$ (Mi à l'octave 4).

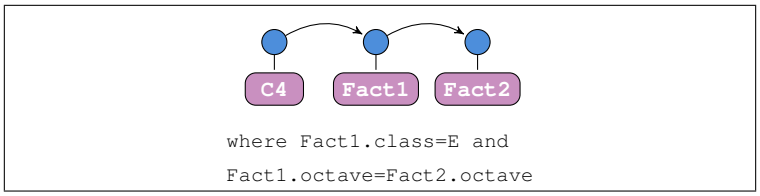


FIGURE 10.: Deux évènements suivant une note $C4$

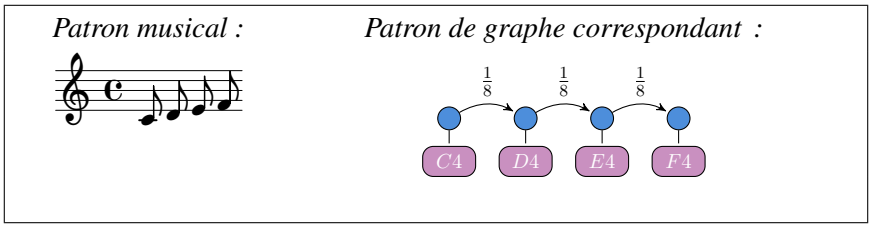


FIGURE 11.: Séquence de notes C4-D4-E4-F4

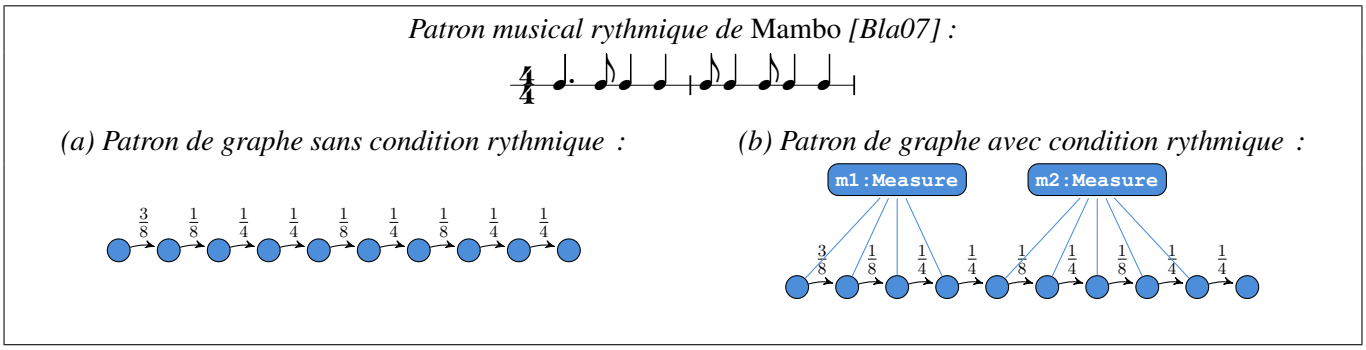


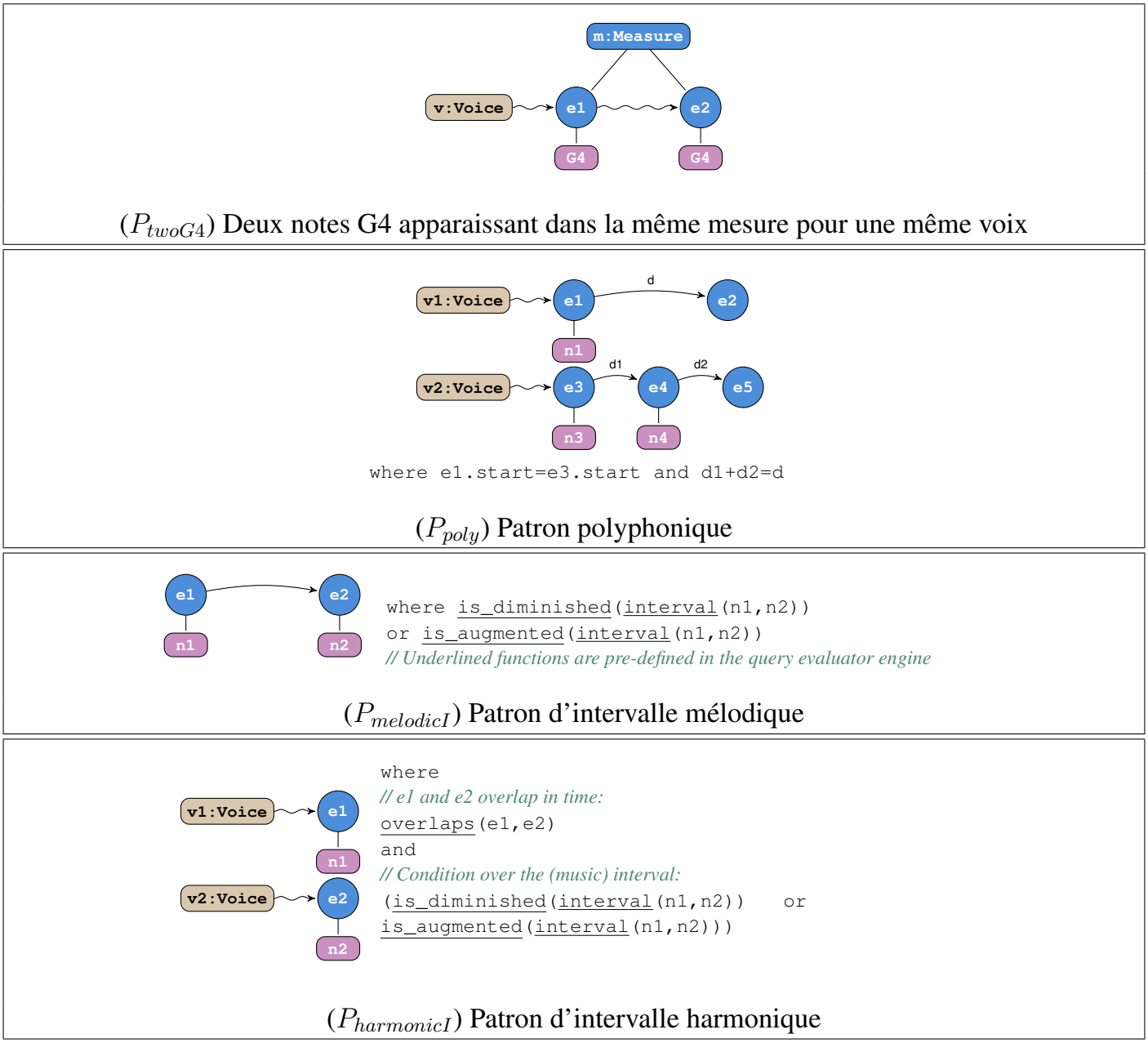
FIGURE 12.: Patron rythmique de *Mambo*

(`Fact1` doit être une note Mi (E), et `Fact1` et `Fact2` doivent être de la même octave). Aucune autre condition n'est imposée sur les évènements (en particulier, aucune condition n'est à satisfaire concernant le rythme des évènements).

La patron de la figure **11** vise à retrouver les séquences de croches $C4$-$D4$-$E4$-$F4$ qui apparaissent dans les données. Le patron contient des conditions qui contraignent à la fois la hauteur des notes et des propriétés rythmiques (durée des évènements).

Les patrons des figures **10** et **11** se fondent sur le point de vue consistant à voir une voix comme étant une série d'évènements musicaux, indépendamment du découpage rythmique plus global de la partition modélisé sous forme d'arbre rythmique. Aborder une voix comme une série d'évènements musicaux est un point de vue généralement plus naturel que celui consistant à aborder une partition par son arbre rythmique. Les deux points de vue ne sont cependant pas inconciliables. La figure **12** est un patron dont l'objectif est de rapatrier les séquences d'évènements musicaux correspondant à un patron rythmique donné, ici un rythme de Mambo. Deux interprétations en termes de patron de graphe peuvent être envisagées. Le patron (a) est composé d'une suite d'évènements musicaux (variables de nœud et variables d'arc) pour lesquels seuls les éléments rythmiques de la séquence d'évènements sont imposés (valeurs des propriétés *duration* des arcs de la séquence, sans considération des sons). La version (a) adopte exclusivement le point de vue pour lequel une voix est une séquence d'évènements musicaux, et ne prend pas en compte du découpage vertical de la partition. La version (b), elle, combine les deux points de vue en considérant la répartition dans l'arbre rythmique (au niveau des mesures) des évènements musicaux vus sous la forme d'une série d'évènements.

Les patrons de graphe peuvent prendre une forme plus complexe, considérant plusieurs voix, et comportant éventuellement des conditions exprimées dans une forme flexible (expressions régulières pour définir un chemin reliant deux nœuds). Dans la figure **13**, le patron P_{twoG4} a vocation à rapatrier les sous-graphes comportant deux notes $G4$ qui appartiennent à la même mesure, dans la même voix (les deux notes ne sont pas nécessairement adjacentes). La relation spécifiée sous la forme d'un arc ondulé entre deux nœuds définit un chemin de taille arbitraire reliant les deux nœuds et dont la forme n'est pas contrainte.



(P_{twoG4}) Deux notes G4 apparaissant dans la même mesure pour une même voix

(P_{poly}) Patron polyphonique

($P_{melodicI}$) Patron d'intervalle mélodique

($P_{harmonicI}$) Patron d'intervalle harmonique

FIGURE 13.: Exemples de patrons de graphe complexe

La plupart des langages concrets d'interrogation de bases de données graphe par des requêtes à base de patron de graphe, tels que le langage Cypher présenté en section 4, offrent la possibilité de manipuler plus finement les objets d'une requête en effectuant des calculs sur les valeurs de leurs propriétés (transformation et agrégation de valeurs), enrichissant ainsi le langage d'interrogation. Ces langages offrent également la possibilité de pré-définir des fonctions spécialisées qui implantent des calculs d'intérêt pour les utilisateurs. Par exemple, il est possible de définir une fonction *interval()* qui implante la notion d'intervalle (diatonique) entre deux hauteurs de note, ou d'autres fonctions permettent de vérifier les

propriétés d'un intervalle, par exemple les fonctions Booléennes *is_diminished* ou *is_augmented*. Il est également possible de définir une fonction *overlaps* qui permet, étant donnés deux évènements, de dire si les évènements se chevauchent dans le temps. Le patron de graphe polyphonique P_{poly} utilise ce type de fonction. Il vise à rapatrier les occurrences de deux notes (`n3` et `n4`), jouées sur le laps de temps d'une autre note (`n1`) dans une autre voix. Les patrons $P_{melodicI}$ et $P_{harmonicI}$ quant à eux visent à identifier les intervalles (respectivement, mélodiques ou harmoniques), diminués ou augmentés.

L'interprétation (évaluation) $[\![P]\!]_{\mathcal{G}}$ d'un patron de graphe P sur une base de données graphe $\mathcal{G}$ consiste à trouver tous les sous-graphes de $\mathcal{G}$ correspondant à la forme (incluant les conditions associées) définie par P.

Definition 3..2 (Patron de graphe – interprétation)**.** *La réponse $[\![P]\!]_{\mathcal{G}}$ à une requête à base du patron P, appliquée au graphe $\mathcal{G}$, est l'ensemble des sous-graphes $\{g \in \mathcal{P}(\mathcal{G}) \mid g$ "correspond à" $P\}$. Un graphe g "correspond à" P ssi il existe un homomorphisme h allant des variables de nœud et d'arc de P dans $\mathcal{G}$ et tel que chaque nœud (resp. arc) $h(v)$ satisfait les conditions $Cond_n$ (resp. $Cond_e$) qui lui sont associées.*

En guise d'illustration, reprenons le patron P_{twoG4} de la figure **13** et considérons son évaluation sur le graphe $\mathcal{G}_{\text{BWV111}}$. Plusieurs sous-graphes de $\mathcal{G}_{\text{BWV111}}$ correspondent au patron P_{twoG4}. Parmi ces sous-graphes, on peut retrouver les deux sous-graphes suivants (voir la figure **14**) :

— le sous-graphe contenant la première et la troisième note de la mesure 1 pour la voix *Soprano*, et
— le sous-graphe contenant la première et la seconde note de la mesure 2 pour la voix *Alto*.

Ces deux sous-graphes appartiennent à l'ensemble $[\![P_{twoG4}]\!]_{\mathcal{G}_{\text{BWV111}}}$ des réponses de l'évaluation du patron P_{twoG4} sur le graphe $\mathcal{G}_{\text{BWV111}}$ (parmi d'autres réponses possibles).

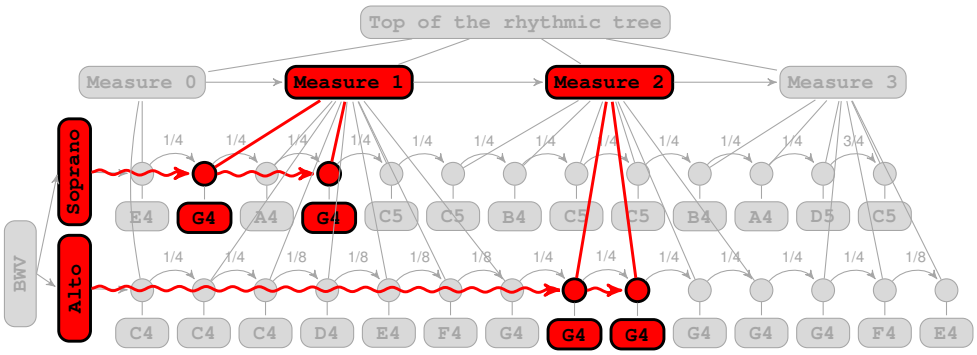


FIGURE 14.: ($\mathcal{G}_{\text{BWV111}}$) Deux réponses de $[\![P_{twoG4}]\!]_{\mathcal{G}_{\text{BWV111}}}$

4. Preuve de concept

On s'intéresse maintenant à une preuve de concept du cadre théorique présenté dans les sections précédentes. Elle repose sur un outil prototype nommé MUSYPHER, qui permet d'alimenter une base de données graphe à partir de données de contenu musical de partitions musicales. MUSYPHER [8] est une application Java en charge de transformer une partition, encodée au format XML MEI, en sa représentation sous forme d'un graphe de données respectant le modèle de données proposé dans la section 2, puis d'injecter ce graphe dans une base de données Neo4J [Neo22b]. Les partitions injectées dans la

8. La version actuelle de MUSYPHER est disponible à l'adresse `https://www-shaman.irisa.fr/musypher/`.

base de données peuvent ensuite être explorées à l'aide d'outils d'interrogation de données, par exemple l'outil console fourni par Neo4J permettant l'interrogation des données par le langage d'interrogation Cypher [Neo22a], un langage à base de patron de graphe.

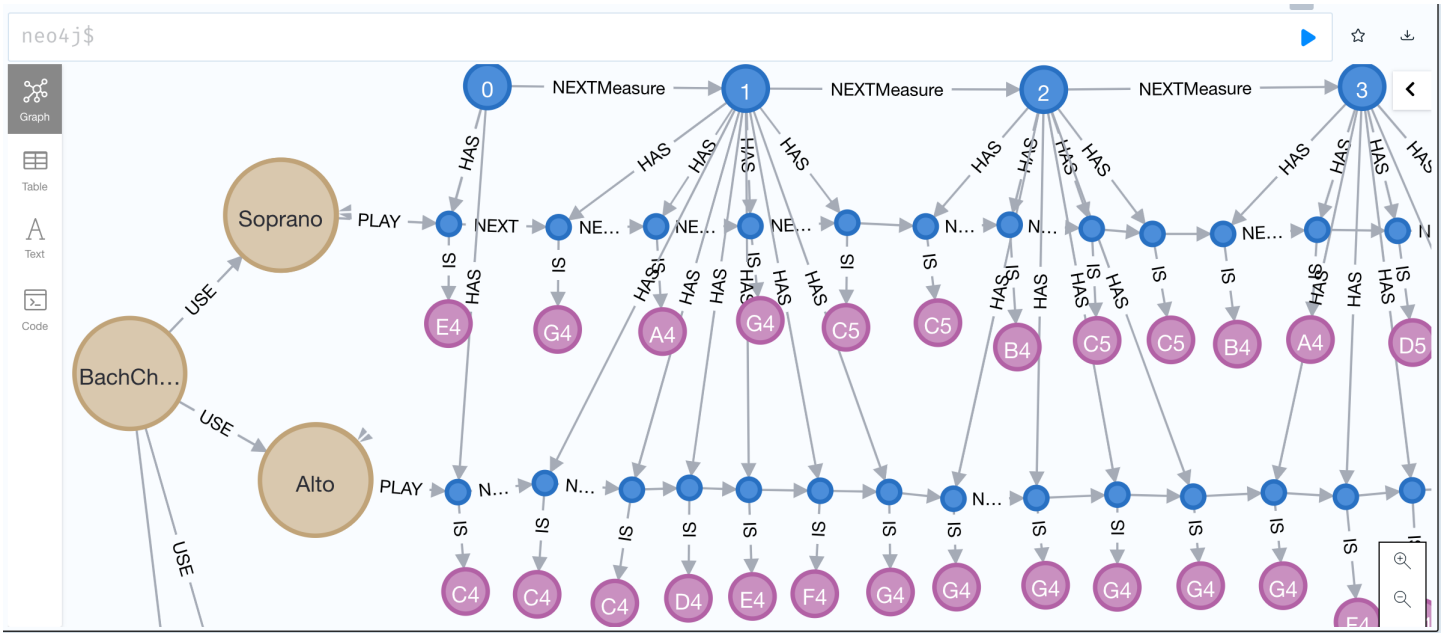


FIGURE 15.: Capture écran *Neo4j Desktop* :BWV111

Considérons de nouveau la partition musicale BWV111 de la figure **8** (p. 12) dont la représentation sous forme de graphe a été injectée par MUSYPHER dans une base de données Neo4J (à partir du fichier MEI initial disponible sur la plateforme NEUMA [Neu22]). La figure **15** est une capture d'écran de l'outil de visualisation *Neo4j Desktop* connecté à la base de données contenant la partition. [9] On reconnaît aisément la forme du graphe $\mathcal{G}_{\text{BWV111}}$ de la figure **9** (p. 13).

Illustrons maintenant l'interrogation des données de $\mathcal{G}_{\text{BWV111}}$ *via* des requêtes Cypher à base de patron de graphe. Dans le langage Cypher, le patron P_{twoG4} (figure **13**) est exprimable par la requête Q_{twoG4} de la figure **16**. La forme du patron de graphe est déclarée dans la clause MATCH (lignes 1 à 6). Dans le formalisme Cypher, le symbole graphique `()` représente un nœud, et peut contenir des informations de la forme `variable:Type`. Le symbole `–[expr]–>` exprime une connexion entre deux nœuds. `expr` peut être une simple étiquette représentant un arc, ou une expression régulière plus complexe représentant la forme d'un chemin. Les expressions de la forme `{attr1:value1, attr2:value2, ...}` sont des conditions supplémentaires portant sur les propriétés des nœuds ou des arcs. La clause RETURN (lignes 7 à 9) définit les éléments d'intérêt à renvoyer en tant que résultat de la requête, parmi les sous-graphes réponse. Dans la requête Q_{twoG4}, on récupère pour chaque sous-graphe réponse : le numéro de la mesure, le nom de la voix, et l'identifiant MEI initial des éléments. [10]

Une partie du résultat de l'évaluation, par l'évaluateur de requête de Neo4J, de Q_{twoG4} appliquée à $\mathcal{G}_{\text{BWV111}}$, est donnée est figure **16** (capture d'écran dans de l'outil *Neo4j Desktop*, dans la partie inférieure de la figure). Parmi les réponses fournies, les première et sixième lignes correspondent aux sous-graphes réponses de l'évaluation de P_{twoG4} sur $\mathcal{G}_{\text{BWV111}}$ que nous avions précédemment identifiés (voir

9. Il est bien entendu possible d'intégrer plusieurs partitions à une même base de données.
10. Dans la pratique, les identifiants MEI sont considérés comme des données d'intérêt dans le cadre d'un post-traitement des données rappatriées. Ces identifiants sont transférés par MUSYPHER par le biais de propriétés associées aux objets.

la figure **14** et la discussion associée p. 16).

```
MATCH
 (v:Voice)–[*]–>(e1:Event)–[*]–>(e2:Event), // e1 belongs to a voice. e2 belongs to the same voice and
      appears "somewhere" after e1.
 (m:Measure)––(e1), // e1 belongs to a measure (m)
 (m)––(e2), // e2 belongs to the same measure (m)
 (e1)––({class:'g',octave:4}), // e1 is a G4
 (e2)––({class:'g',octave:4}) // e2 is a G4
RETURN
 m.number AS m, v.name AS v,
 e1.id AS src_id1, e2.id AS src_id2
```

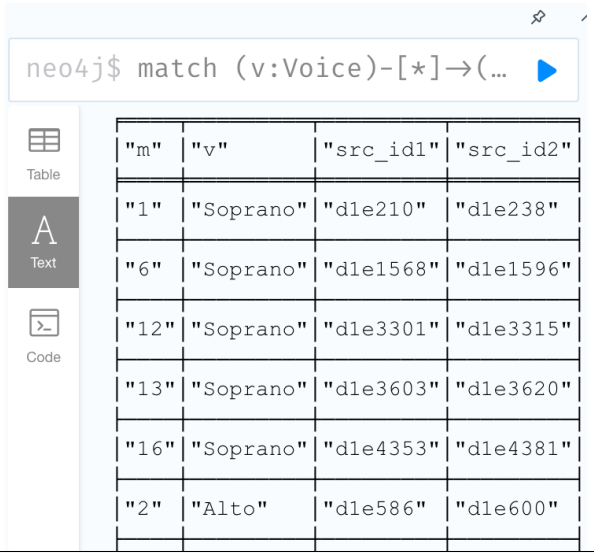


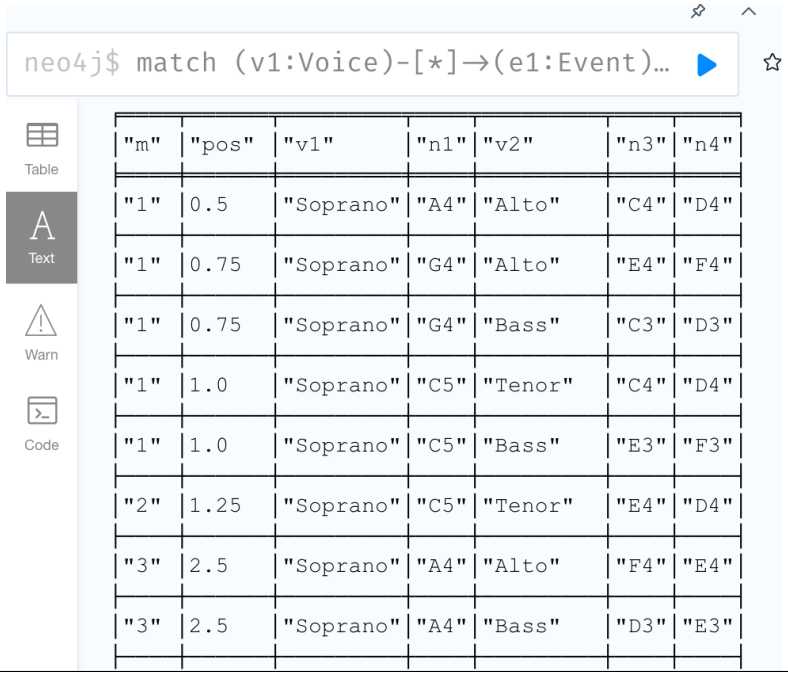
FIGURE 16.: Requête Q_{twoG4} et résultat de son évaluation par Neo4j sur $\mathcal{G}_{\text{BWV111}}$

Une autre illustration est la requête Q_{poly} de la figure **17** (page 19), exprimant dans le langage Cypher le patron de graphe P_{poly} de la figure **13** (page. 15). La première ligne du résultat montre qu'au temps 0.5 $(2/4)$ de la musique, la voix *Soprano* joue une note $A4$ (deuxième note de la mesure 1 de cette voix) pendant que la voix *Alto* joue une note $C4$ suivie d'une note $D4$ (deuxième et troisième notes de la mesure 1 de cette voix).

5. Conclusion and perspectives

Dans cet article, nous avons proposé un modèle de données orienté graphe permettant de modéliser le contenu musical d'une partition musicale. Le modèle proposé intègre à la fois le point de vue du découpage "vertical" des données de la partition visant à représenter le contenu musical selon la structure rythmique de la partition, et le point de vue du découpage "horizontal" des données visant à représenter les données d'une partition comme un ensemble de séries temporelles d'évènements musicaux synchronisées. Nous avons ensuite exploré le potentiel offert par ce type de modélisation, en termes d'interrogation des données par des requêtes à base de patron de graphe. Nous avons également présenté une preuve de concept de cette approche théorique, fondée sur un outil appelé MUSYPHER, capable de transformer une partition encodée dans un format standard XML en sa représentation sous forme du modèle de données proposé, puis de l'injecter dans une base de données Neo4J en vue d'une interrogation *via* le langage de requête Cypher.

```
MATCH
(v1:Voice)–[*]–>(e1:Event)–[r:NEXT]–>(e2:Event),
(v2:Voice)–[*]–>(e3:Event)–[r1:NEXT]–>(e4:Event),
(e4)–[r2:NEXT]–>(e5:Event), (m:Measure)––(e1),
(e1)––(n1:Fact{type:'note'}),
(e3)––(n3:Fact{type:'note'}),
(e4)––(n4:Fact{type:'note'})
WHERE
 e1.start=e3.start AND // start of time intervals
 r1.duration+r2.duration = r.duration // durations
RETURN
 m.number AS m, e1.start AS pos,
 v1.name AS v1, n1.name AS n1, v2.name AS v2,
 n3.name AS n3, n4.name AS n4
```



"m"	"pos"	"v1"	"n1"	"v2"	"n3"	"n4"
"1"	0.5	"Soprano"	"A4"	"Alto"	"C4"	"D4"
"1"	0.75	"Soprano"	"G4"	"Alto"	"E4"	"F4"
"1"	0.75	"Soprano"	"G4"	"Bass"	"C3"	"D3"
"1"	1.0	"Soprano"	"C5"	"Tenor"	"C4"	"D4"
"1"	1.0	"Soprano"	"C5"	"Bass"	"E3"	"F3"
"2"	1.25	"Soprano"	"C5"	"Tenor"	"E4"	"D4"
"3"	2.5	"Soprano"	"A4"	"Alto"	"F4"	"E4"
"3"	2.5	"Soprano"	"A4"	"Bass"	"D3"	"E3"

FIGURE 17.: Requête Q_{poly} et résultat de son évaluation

Ces travaux ouvrent la voie à de nombreuses perspectives. Certaines d'entre elles concernent l'*extension du modèle de données*. Dans l'état actuel des travaux, le modèle ne considère que la modélisation des faits de type sons, mais d'autres domaines peuvent être ajoutés pour enrichir la description de l'information musicale (syllabes, timbre, intensité), en respectant le principe consistant à mettre en correspondance les feuilles de l'arbre rythmique avec des faits musicaux.

En ce qui concerne l'interrogation, une étude théorique plus détaillée doit être menée concernant l'expressivité et le coût de l'interrogation des données respectant le modèle. Le problème de l'interrogation de données graphe a été intensément étudié dans la littérature [Woo12, Bar13, LMV13, BLR14, AAB+17], on peut s'appuyer sur ces travaux pour amorcer une première réflexion sur ce sujet. Concernant l'*expressivité du langage d'interrogation*, un arbre étant un cas particulier de graphe, les requêtes de type navigationnelles (à la XPath) peuvent être déclinées sur des donnés représentées respectant une structure de graphe [LMV13], capturant ainsi l'expressivité offerte par l'interrogation de données modélisées sous forme d'arbre dans les formalismes semi-structurés actuels. Les requêtes à base de patron de graphe, elles, sont un formalisme très expressif, et donc coûteux en termes de coût d'évaluation. Le *coût de l'évaluation d'une requête à base de patron* dépend de la forme de la requête, allant de $O(|\mathcal{G}|.|P|)$, où where $|\mathcal{G}|$ est la taille du graphe de données et $|P|$ est la taille du patron de graphe, pour un patron

prenant la forme la plus simple d'un chemin avec expression régulière (un chemin, reliant deux nœuds, dont la forme est définie par une expression régulière) à un problème NP-complet pour le cas le plus général de patron contenant un cycle.

Un dernier ensemble de perspectives concerne l'aspect applicatif de ces travaux. Nous désirons adapter et intégrer l'application existante en tant que module d'exploration au sein de bibliothèques de partitions musicales (dont la plateforme NEUMA), afin d'offrir la possibilité de l'interrogation des données sous forme de graphe. Une autre amélioration concerne la restitution des réponses que nous planifions de rendre plus conviviale, par exemple par la mise en relief des réponses à une requête sur les partitions restituées graphiquement (utilisation du framework Verovio [PZR14]).

Remerciements. Les auteurs remercient Clément Van Straaten pour son travail mené sur le développement de la seconde version de l'outil MUSYPHER.

Bibliographie

Renzo Angles, Marcelo Arenas, Pablo Barceló, Aidan Hogan, Juan L. Reutter, and Domagoj Vrgoc. Foundations of modern query languages for graph databases. *ACM Comput. Surv.*, 50(5) :68 :1–68 :40, 2017.

Renzo Angles and Claudio Gutierrez. Survey of graph database models. *ACM Computing Surveys (CSUR)*, 40(1) :1–39, 2008.

Renzo Angles. A comparison of current graph database models. In *Proc. of the Intl. Conf. on Data Engineering (ICDE) Workshops*, pages 171–177, 2012.

Willi Apel. *The Notation Of Polyphonic Music 900-1600*. The Mediaeval Academy Of America, 1961.

Pablo Barceló. Querying graph databases. In *Proc. of the ACM Symposium on Principles of Database Systems (PODS)*, pages 175–188, 2013.

Angela Bonifati, George H. L. Fletcher, Hannes Voigt, and Nikolay Yakovets. *Querying Graphs*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2018.

Alfred Blatter. *Revisiting Music Theory : A Guide to the Practice.* London and New York : Routledge, 2007.

Pablo Barceló, Leonid Libkin, and Juan L. Reutter. Querying regular graph patterns. *J. ACM*, 61(1) :8 :1–8 :54, 2014.

Michael Scott Cuthbert and Christopher Ariza. music21 : A toolkit for computer-aided musicology and symbolic music data. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2010.

Francesco Foscarin, Philippe Rigaux, and Virginie Thion. Data Quality Assessment in Digital Score Libraries. The GioQoso Project. *Intl. Journal on Digital Libraries*, 22(2) :159–173, 2021.

Raphaël Fournier-S'niehotta, Philippe Rigaux, and Nicolas Travers. Querying XML Score Databases : XQuery is not Enough ! In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2016.

Raphaël Fournier-S'niehotta, Philippe Rigaux, and Nicolas Travers. Modeling Music as Synchronized Time Series : Application to Music Score Collections. *Information Systems*, 73 :35–49, 2018.

Michael Good. *The Virtual Score : Representation, Retrieval, Restoration*, chapter MusicXML for Notation and Analysis, pages 113–124. W. B. Hewlett and E. Selfridge-Field, MIT Press, 2001.

Elaine Gould. *Behind Bars*. Faber Music, 2011.

Joachim Ganseman, Paul Scheunders, and Wim D'haes. Using XQuery on MusicXML databases for musicological analysis. In *Proc. of the Intl. Conf. on Music Information Retrieval, (ISMIR)*, pages 433–438, 2008.

Joseph Haydn. Das lied der deutschen, 1797. Lyrics by August Heinrich Hoffmann von Fallersleben.

David Huron. Music Information Processing Using the Humdrum Toolkit : Concepts, Examples, and Lessons. *Computer Music Journal*, 26(2) :11–26, 2002.

David Huron. The humdrum toolkit : Software for music research, 1994-.

International Organization for Standardization. ISO 16 :1975 Acoustics – Standard tuning frequency (Standard musical pitch), 1975.

Rong Jin and Christopher Raphael. Graph-based rhythm interpretation. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 343–349, 2015.

Leonid Libkin, Wim Martens, and Domagoj Vrgoč. Querying Graph Databases with XPath. In *Proceedings of the Intl. Conf. on Database Theory ICDT*, pages 129–140, 2013.

Music Encoding Initiative. `http://www.music-encoding.org`, 2022. Accessed Feb. 2022.

Music Notation Community Group. `https://www.w3.org/community/music-notation/`, 2018. Accessed April 2019.

MuseScore. Web site. https ://musescore.org/.

MusicXML. `http://www.musicxml.org`, 2022. Accessed Feb. 2022.

Neo Technology. The Neo4j Manual. https ://neo4j.com/developer, 2022. Accessed Feb. 2022.

Neo4j web site. `www.neo4j.org`, 2022. Accessed Feb. 2022.

NEUMA. `http://neuma.huma-num.fr`, 2022. Accessed Feb. 2022.

Laurent Pugin, Rodolfo Zitellini, and Perry Roland. Verovio : A library for Engraving MEI Music Notation into SVG. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 107–112, 2014.

Philippe Rigaux, Lylia Abrouk, H. Audéon, Nadine Cullot, C. Davy-Rigaux, Zoé Faget, E. Gavignet, David Gross-Amblard, A. Tacaille, and Virginie Thion-Goasdoué. The design and implementation of neuma, a collaborative digital scores library - requirements, architecture, and models. *Intl. Journal on Digital Libraries*, 12(2-3) :73–88, 2012.

Perry Rolland. The Music Encoding Initiative (MEI), 2002.

Craig Stuart Sapp. Online Database of Scores in the Humdrum File Format . In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2005.

Peter T. Wood. Query languages for graph databases. *SIGMOD Rec.*, 41(1) :50–60, 2012.

Jim Webber, Ian Robinson, and Emil Eifrem. *Graph Databases*. O'Reilly Media, 2013.

Tiange Zhu, Raphaël Fournier-S'niehotta, Philippe Rigaux, and Nicolas Travers. A Framework for Content-based Search in Large Music Collections. *Big Data and Cognitive Computing*, 23(6), 2022.

