ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA
SCUOLA DI SCIENZE

CORSO DI LAUREA IN INGEGNERIA E SCIENZE INFORMATICHE

Progettazione di un sistema per la modellazione di melodie musicali come reti complesse: conversione da formato MIDI a MusicXML

Relazione finale in
SISTEMI MULTIMEDIALI

Relatore
Prof. STEFANO FERRETTI

Presentata da
SOFIA ROSETTI

Sessione II
Anno Accademico 2015/2016

Indice

Capitolo 1

Introduzione

Il progetto che verrà presentato consiste in uno strumento software che effettua una conversione di file, dal formato MIDI al formato MusicXML.

L'idea nasce come proseguimento di un'attività di tirocinio basata sull'analisi di reti complesse e sullo sviluppo di strumenti per la modellazione di spartiti musicali.

Ad oggi le reti complesse rivestono un argomento di ricerca multidisciplinare, in cui spesso si cerca di rappresentare sistemi reali come un insieme di nodi interconnessi da un insieme di archi.

La teoria delle reti viene impiegata in diversi campi, dalle scienze fisiche e naturali a quelle sociali e umanistiche, e può essere vista come uno strumento matematico in grado di connettere il mondo reale con ricerche teoriche.

Allo stesso modo con cui sistemi di tipo economico, tecnologico e biologico possono essere modellati come reti complesse, così anche i brani musicali possono essere analizzati. Melodie, assoli ed in generale tracce musicali sono rappresentabili come reti complesse, in cui i nodi identificano le note del brano e gli archi mostrano le sequenze con cui queste vengono suonate.

L'argomento è stato trattato in "*Guitar Solos As Networks*" (Stefano Ferretti, 2016), dove viene presentato il problema della rappresentazione

di melodie tramite reti complesse, in aggiunta all'analisi di alcuni assoli di chitarra.

L'attività di tirocinio si è basata su un tool fornito dal Docente in grado di analizzare diversi assoli musicali di alcuni chitarristi in modo da identificare le caratteristiche principali dell'assolo e di estrarre quindi lo stile del musicista. Infatti, trasformando una traccia musicale in una rete complessa di unità musicali, quali note e pause, e delle loro relazioni, è auspicabile ottenere le proprietà derivanti dall'interazione di questi elementi. Ciò è possibile grazie al complesso sistema ed al potente quantitativo di misure che, applicate al contesto, sono in grado di catturare l'essenza della complessità di una rete.

Utilizzando la teoria delle reti è possibile estrarre alcune metriche principali, tipiche delle reti, che caratterizzano il brano in questione. Questa analisi potrebbe avere un impatto concreto su alcune applicazioni multimediali, come la generazione automatica di melodie e musica digitale, la classificazione o l'identificazione di brani musicali. Questi applicativi possono essere ampiamente sfruttati all'interno di scenari didattici o di intrattenimento multimediale.

Oggigiorno l'audio riveste un'importanza non irrilevante all'interno di numerosi contesti, e negli ultimi anni è stata sempre più affrontata la tematica della digitalizzazione dell'audio, insieme a ciò che concerne sintetizzazione, rappresentazione e riproduzione di suoni in generale. Gli studi al riguardo hanno posto maggiormente l'attenzione su come trasmettere, classificare o indicizzare le tracce musicali, mentre non si sono focalizzati sull'idea di estrarre alcune caratteristiche generali di una melodia.

La base di questi studi può essere formata sia da dati simbolici, come le partiture musicali, sia da registrazioni audio. In molti contesti applicativi rappresentazioni grafiche di tracce musicali possono facilitarne notevolmente l'analisi. Ad esempio, identificare le diverse note di una melodia all'interno di un file audio è spesso un compito difficile e non immediato, mentre una rappresentazione grafica dello stesso consente di ottenere immediatamente una lista di figure e quindi di utilizzare le note come un punto di partenza nell'analisi.

Ottenere una sequenza di note direttamente da un file audio, prescindendo da una partitura grafica, è il problema di base che è stato affrontato nello sviluppo del progetto.

La modellazione di brani musicali come reti complesse si avvale di rappresentazioni delle tracce secondo lo standard MusicXML, nel quale è chiaramente presentata la sequenza di note caratterizzanti. File in formato MusicXML sono ancora poco presenti e possono ad oggi essere ricavati utilizzando software musicali professionali.

Uno dei software che verranno presentati nel corso dei capitoli e di cui si è ampiamente fatto uso è Finale Print Music. Tale strumento è in grado di importare un file in formato MIDI, creare lo spartito grafico corrispondente ed esportare tale rappresentazione in un file in formato MusicXML.

L'obiettivo di questo progetto è effettuare una conversione tra i due formati senza l'intermediazione di una rappresentazione grafica, e che consenta di ottenere un set di partiture più ampio su cui effettuare i diversi tipi di analisi. É importante sottolineare che, sebbene siano equivalenti, non sempre tutti gli spartiti sono uguali, ed a seconda di come sono stati generati possono interpretare la traccia e mostrarla in modo leggermente diverso.

Il lavoro che è stato svolto si basa sull'analisi iniziale di un file in formato MIDI, dal quale è possibile ricavare gli elementi fondamentali per la rappresentazione di una melodia come una rete.

É stata per cui studiata con attenzione e poi utilizzata la struttura di diversi file MIDI, producendo un tool in grado di estrarre le informazioni di interesse da un file MIDI a scelta dell'utente. Il software realizzato estrae le suddette informazioni e le rielabora con l'obiettivo ultimo di creare un file in formato MusicXML che rispecchia il più possibile la struttura originaria del brano.

I file MusicXML ottenuti sono stati poi utilizzati per la creazione delle reti modellanti i relativi brani. Su queste ultime sono state infine applicate le principali metriche di analisi di rete, effettuando poi le dovute valutazioni.

Il lavoro è stato articolato analizzando inizialmente un file MIDI realizzato appositamente per questo progetto. La produzione autonoma di un file di questo tipo si è resa necessaria al fine di consentire una più agevole analisi della struttura di base del formato MIDI. Successivamente, sono stati analizzati diversi brani musicali di repertorio, i cui file MIDI sono stati recuperati su Web.

La struttura dei capitoli successivi è la seguente.

Nel capitolo 2 verranno esposti i concetti fondamentali per comprendere la tesi. Analizzando il problema, verranno descritti analiticamente i due formati di partenza e destinazione, evidenziandone le strutture e le caratteristiche più rilevanti all'interno del progetto.

Nel capitolo 3 verrà presentato il modo in cui il problema è stato affrontato e come è stato risolto, indicando le maggiori criticità.

Nel capitolo 4 verranno mostrati i casi presi in esame, descrivendo le differenze tra i risultati attesi ed i risultati ottenuti.

Nel capitolo 5 verranno tratte le conclusioni sul lavoro svolto, accennando ai possibili contesti d'utilizzo ed agli eventuali sviluppi futuri.

Capitolo 2

Descrizione del problema e stato dell'arte

L'utilizzo di partiture musicali codificate in formato MusicXML per ottenerne le rispettive rappresentazioni di reti ed effettuarne le dovute analisi ha portato alla necessità di avere un set di partiture più esteso, in quanto, ad oggi, file di questo tipo sono difficilmente reperibili a causa dei diversi tipi di strumenti e delle differenti strutture musicali delle tracce.

L'oggetto del problema è quindi costituito dalla conversione di due differenti tipologie di file: un file di partenza in formato MIDI ed un file di destinazione in formato MusicXML.

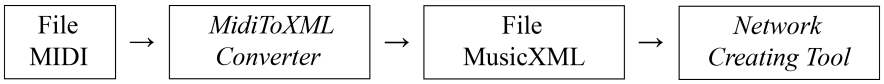


Figura 2.1: Schema riassuntivo del problema

Il processo di conversione richiede che dal formato di partenza vengano estratte le informazioni necessarie a costituire gli elementi del formato di destinazione. Tali corrispondenze, senza l'ausilio di una rappresentazione

grafica del brano, non sempre sono dirette, per cui è stato necessario effettuare diversi calcoli e approssimazioni descritti nel capitolo successivo.

In seguito verrà presentata l'idea di rappresentazione della musica tramite rete complessa, introducendo poi alcuni termini di uso frequente nel corso del testo, con l'obiettivo di rendere chiaro tutto lo sviluppo del lavoro e di evitare possibili ambiguità. L'analisi proseguirà presentando le strutture dei due formati oggetto di conversione.

2.1 Reti complesse, modelli e proprietà

La topologia di una rete è il modello geometrico (tendenzialmente un grafo) che rappresenta le connessioni tra gli elementi costitutivi della rete stessa, ovvero i nodi.

Una rete complessa si può definire come una rete che presenta una struttura topologica non banale, con caratteristiche non intuitive e costituita anche da milioni di unità comunicanti fra loro. Le reti complesse costituiscono un ambito giovane ed in continua crescita della ricerca scientifica, in quanto possono essere efficacemente utilizzate per modellare graficamente sistemi reali.

2.1.1 Modelli di rete

Quando si cerca di modellare reti complesse si fa riferimento, in ordine cronologico, a tre modelli principali:

- Rete di Erdös-Rényi
- Reti Small world
- Reti Scale-free

Rete di Erdös-Rényi

Una rete di Erdös-Rényi è una tipologia di rete introdotta a fine anni '50, definita anche come rete Random.

L'idea di base consiste nel considerare un grafo semplice e connesso, in cui, dato un numero fisso n di vertici, gli archi vengono creati in modo casuale, secondo una certa distribuzione di probabilità. Il grafo di partenza è un grafo semplice, in cui non sono presenti cicli e dove può esservi al massimo un arco tra due vertici distinti.

Considerando un intero n ed una probabilità p, la distribuzione G(n:p) produce un grafo random sugli n vertici, scegliendo ogni possibile arco da aggiungere nel grafo con probabilità p.

Il numero atteso di archi in un grafo derivante da tale distribuzione sarà quindi

$$\binom{n}{2}p$$

e il degree atteso di ogni nodo è

$$(n-1)p.$$

Si definisce componente di un grafo $G = (V,E)$ un sottografo i cui nodi $S \subset V$ e gli archi E_S connettono i nodi S e non connettono i nodi S con nessun nodo non in S.

Caratteristiche principali di una rete di Erdös-Rényi sono:

- Una distanza media fra i nodi bassa.
- Un coefficiente di clustering pari alla probabilità p con cui ogni coppia di nodi distinti sia connessa da un arco.
- Una distribuzione del grado dei nodi che segue una distribuzione binomiale.
- C'è una relazione tra la probabilità p che due vertici siano connessi e la dimensione delle componenti del grafo.
- Aumentando la probabilità p aumenta anche la densità di rete del grafo e la maggior parte dei nodi fa parte della stessa componente, detta componente gigante. Più precisamente, la probabilità p è direttamente proporzionale alla dimensione della componente gigante.

Un esempio di rete di Erdös-Rényi è mostrato in Figura 2.2.

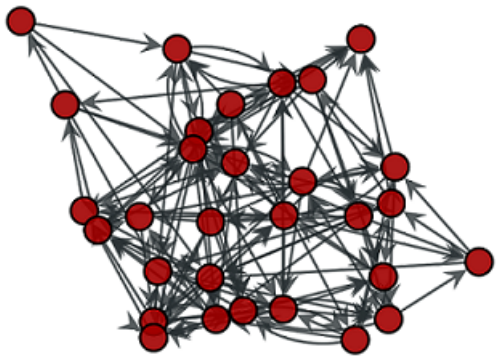


Figura 2.2: Esempio di Grafo Random che rappresenta una rete di Erdös-Rényi.

Reti Small world

Il modello Small world nasce dagli studi di uno psicologo di Harvard, Stanley Milgran, interessato a scoprire quale fosse la distanza media, in una rete sociale, di una qualsiasi coppia di persone scelte a caso. Gli esperimenti portarono a scoprire che la distanza media fra due persone scelte casualmente era molto bassa, nello specifico 5,5, da cui poi derivò la nota frase "sei gradi di separazione".

Formalmente, si definisce una rete small-world come un grafo dove la distanza media fra una qualsiasi coppia di nodi scelta casualmente cresce in modo proporzionale al logaritmo del numero di nodi nella rete.

Perciò la maggior parte dei nodi non è necessariamente adiacente, ma per ogni coppia di nodi esiste un cammino relativamente breve che li collega.

É possibile osservare che in molte reti reali è presente questa proprietà, ovvero la distanza tra due nodi qualsiasi della rete è relativamente piccola rispetto alle dimensioni della rete.

Il primo modello di rete small-world fu introdotto nel 1998 da Watts e Strogatz, i quali osservarono che molte reti reali mostrano un alto coefficiente di clustering mantenendo però le caratteristiche delle reti di Erdös-Rényi. Inoltre venne dimostrato che la maggior parte dei vertici tende a raggrupparsi fra loro e che una buona parte di essi possiede anche un arco verso un vertice più distante. Questi tipi di archi sono definiti come "archi deboli" e rivestono un ruolo importante all'interno della rete.

L'algoritmo di Watts e Strogatz tuttavia non riesce, al crescere della dimensione della rete, a mantenere una bassa distanza fra i nodi, caratteristica principale delle reti small-world.

Un esempio di rete Small world è mostrato in Figura 2.3.

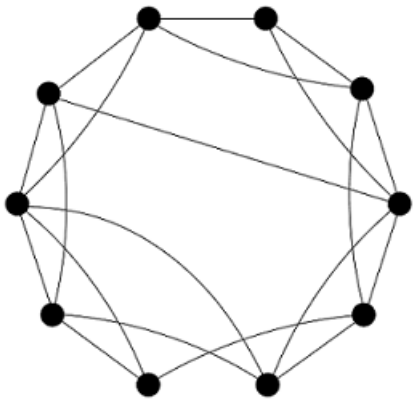


Figura 2.3: Esempio di rete Small World

Reti Scale-free

Il modello di Watts e Strogatz non riesce a catturare alcune proprietà di molte reti reali. Nel 1999 Barabàsi e Albert scoprirono che molte reti

reali possiedono una proprietà fondamentale: la distribuzione del grado segue una legge di potenza (power law), secondo la quale in un grafo, la probabilità che un generico nodo abbia grado k è proporzionale a $(\frac{1}{k})^{\alpha}$ con $\alpha > 1$, detto esponente di scala.

Nelle reti scale-free vi sono pochi vertici, detti hub, che hanno un grado elevato, mentre la maggior parte possiede un grado basso.

Caratteristiche delle reti scale-free che favoriscono il formarsi di nodi con grado elevato sono:

- Una crescita dinamica della rete: il numero di nodi cresce con il tempo ed i nodi più "vecchi" hanno maggior opportunità di acquisire nuovi collegamenti.
- Preferential Attachment: i collegamenti non vengono generati in modo random poiché alcuni nodi vengono scelti più frequentemente. In particolare, più è alto il grado di un nodo "vecchio", maggiore è la probabilità che un nuovo nodo stabilisca un collegamento con esso.

Uno dei vantaggi delle reti scale-free è la robustezza rispetto ai guasti. Se si rimuove in modo casuale (assumendo una distribuzione uniforme) un nodo da una rete scale-free, con alta probabilità tale vertice sarà un vertice di grado basso, la cui rimozione non costituisce un evento grave.

Uno svantaggio tuttavia è la vulnerabilità verso gli attacchi. Un attacco può colpire i vertici di grado più alto e la rimozione di un numero limitato di hubs può provocare il frazionamento della rete. Se la rete viene frammentata in tante componenti di piccola dimensione, la gravità dell'evento è decisamente maggiore.

Un esempio di rete Scale Free è mostrato in Figura 2.4, in cui sono evidenziati con colore rosso i vertici con grado più elevato.

2.1.2 Proprietà delle reti

All'interno di una rete, un nodo si definisce "importante" se occupa una posizione strategica al suo interno.

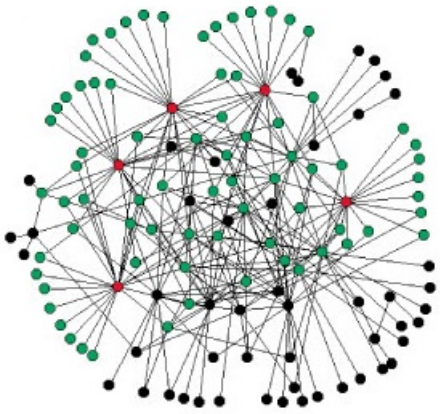


Figura 2.4: Esempio di rete Scale Free

La centralità misura l'importanza relativa di un nodo all'interno di un grafo $G = (V,E)$. Esistono diverse misure di centralità. In questo capitolo verranno presentate alcune metriche generali per l'analisi di grafi: Degree centrality, Coefficiente di clustering, Closeness centrality, Betweenness centrality, Eigenvector centrality.

Nello studio delle reti complesse la nozione di centralità può essere importante per:

- identificare le aree di una rete
- giudicare l'importanza e la criticità di nodi o aree della rete
- identificare il grado di coesione di un'area della rete
- attribuire una misura di distanza fra nodi o aree della rete

Degree centrality

Questa prima metrica è una delle più semplici utilizzate per descrivere una rete, e fornisce una stima della capacità dei nodi di avere una relazione

diretta con gli altri nodi. In particolare fornisce una stima generale sulla struttura del grafo, basandosi unicamente sul numero di archi incidenti in ogni nodo.

In un grafo $G = (V,E)$, la Degree centrality $DC(v)$ di un nodo $v \in V$ fornisce una stima dell'importanza del nodo v, basandosi sul numero di nodi con cui è in relazione diretta. Formalmente:

$$DC(v) = \sum_{k=1}^{V} e(v_i, v_k)(n-1)^{-1} \tag{2.1}$$

dove l'arco e_{ik} è contato pari a 1 quando è presente un collegamento tra il nodo v_i e un nodo v_k e 0 altrimenti.

La prima parte della formula, ovvero:

$$\sum_{k=1}^{V} e(v_i, v_k)$$

rappresenta il grado di un nodo $v \in V$, e nel caso di un grafo orientato è necessario distinguere fra il numero di archi orientati verso il nodo (archi entranti) ed il numero di archi orientati verso i nodi adiacenti (archi uscenti). Per cui si hanno un grado "entrante" definito come IN-Degree, e un grado "uscente" definito come OUT-Degree.

Per calcolare la Degree centrality in un grafo orientato si considera quindi il grado del nodo v come la somma di IN-Degree e OUT-Degree.

Si definisce Degree distribution (distribuzione del grado) la relazione tra i valori k assunti dal grado ed il numero di nodi che mostrano tale grado.

Coefficiente di clustering

Questa seconda metrica stima quanto i nodi adiacenti ad un nodo siano anch'essi in relazione fra loro. Quindi il coefficiente di clustering è un indice che mostra quanto i vicini di un generico nodo v siano a loro volta

adiacenti. Dato un nodo v, si indica con *N(v)* l'insieme dei nodi adiacenti a v. Definendo rispettivamente n_v e m_v come il numero di nodi in *N(v)* ed il numero di archi presenti nel sottografo indotto da *N(v)*, il coefficiente di clustering *CC(v)* è il rapporto tra il numero di collegamenti fra i membri di *N(v)* e il numero di collegamenti potenziali fra loro:

$$CC(v) = \frac{m_v}{\binom{n_v}{2}} = \frac{2m_v}{n_v(n_v - 1)} \tag{2.2}$$

se il grado di v è maggiore di 1, altrimenti non è definito.

Nel caso di un grafo orientato bisogna distinguere tra archi in entrata e archi in uscita. Il numero massimo di archi presenti nel sottografo indotto da *N(v)* diventa:

$$2\binom{n_v}{2} = n_v(n_v - 1)$$

da cui:

$$CC(v) = \frac{m_v}{2\binom{n_v}{2}} = \frac{m_v}{n_v(n_v - 1)} \tag{2.3}$$

Le definizioni precedenti si riferiscono ai singoli nodi. Il coefficiente di clustering *CC(G)* di un grafo *G = (V,E)* si può definire come la media dei coefficienti per ogni nodo, ovvero:

$$CC(G) = \frac{1}{|V^*|} \sum_{v \in V^*} CC(v) \tag{2.4}$$

dove V^* è la Degree distribution con grado maggiore di 1.

Closeness centrality

La Closeness centrality è una metrica che rappresenta una misura della distanza di un nodo da tutti gli altri nodi. Diversamente dalla Degree

centrality questa metrica non è limitata alla visione locale dei singoli nodi perché necessita di una visione più globale della rete.

Stimando il grado di vicinanza di un nodo dal resto dei nodi del grafo, la Closeness centrality è una metrica utile nell'indicare quali sono i punti della rete che minimizzano la distanza media fra i nodi.

Formalmente, si definisce come:

$$CN(v) = \frac{n-1}{\sum_{u \in V \setminus v} d(u,v)} \tag{2.5}$$

dove *d(u,v)* è la distanza fra i vertici *u,v* di un grafo fortemente connesso e pesato.

Betweenness centrality

Se un nodo fa parte di diversi cammini minimi, allora questo è un nodo che riveste una posizione importante nel dominio rappresentato dalla rete in questione. La Betweenness centrality si basa su tale osservazione: i nodi con un alto valore, se eliminati, possono causare una disconnessione della rete, per cui possono essere visti come punti deboli della rete.

Formalmente, la Betweenness centrality di un nodo si definisce come:

$$BC(v) = \sum_{s \neq v \neq t \in V} \delta_{st}(v) \tag{2.6}$$

dove $\delta_{st}(v)$ rappresenta la dipendenza di coppia, ovvero la frazione dei cammini minimi fra s e t che includono v.

La dipendenza di coppia consente di capire l'importanza del nodo v nelle comunicazioni da s a t. Per cui, la Betweenness centrality di un nodo v consiste nel calcolo della dipendenza di coppia per tutte le possibili coppie di vertici nel grafo.

Eigenvector centrality

Questa metrica, estensione naturale della Degree centrality, misura la centralità di un nodo in base alle sue interazioni con la rete, basandosi sul principio per il quale "un nodo è importante se è collegato a nodi importanti".

La Eigenvector centrality differisce dalla Degree centrality, poiché un nodo che riceve molti collegamenti non ha necessariamente una Eigenvector centrality alta. Inoltre, un nodo con un'alta Eigenvector centrality non ha necessariamente molti collegamenti, in quanto potrebbe averne pochi, ma importanti.

Formalmente, supponendo che $G = (V,E)$ sia un grafo diretto con una matrice di adiacenze indicata con

$$A = \{a_{vt}\}_{\forall v,t \in V}$$

si ha:

$$x_v = \frac{1}{\lambda} \sum_{t \in v} a_{vt} x_t \tag{2.7}$$

dove x_v è la Eigenvector centrality del nodo v, mentre λ è una costante.

2.2 Melodie, tracce e brani musicali

In generale, per canzone o **brano musicale**, si intende uno schema composto da più suoni che possono procedere come "singola successione", definita melodia, o "suoni simultanei", chiamati comunemente accordi, che costituiscono l'aspetto armonico del mondo sonoro.

Gli **accordi** si suddividono in bicordi (formati da due suoni sovrapposti), triadi (formati da tre suoni sovrapposti), quadriadi (formati da quattro suoni sovrapposti) e così via; possono essere riprodotti solo da strumenti in grado, appunto, di eseguire contemporaneamente diverse note, come pianoforte, chitarra, organo, arpa, celesta ecc...

Occorre precisare che gli strumenti prettamente solistici, come quelli a fiato, (es: flauto traverso, clarinetto, fagotto, oboe, tromba, saxofoni ecc...), potendo eseguire solamente un suono alla volta, vengono definiti melodici. I suoni della melodia possono essere riprodotti da strumenti musicali e da voci umane. Ogni parte di un brano suonato da un singolo strumento, o cantato da una singola voce, viene identificata come **traccia**.

Nello schema MIDI di un sequencer, (es: Cubase, Sonar, Digital performer e Logic su piattaforma Apple, ecc...) è possibile affidare ad ogni traccia un singolo strumento o una singola voce umana. Nello specifico, per strumenti ritmici come batteria, percussioni (senza suoni identificati nell'altezza) si affidano tali strumenti solo ed esclusivamente nell'ambito del canale 10.

La sovraincisione è il risultato di più tracce sovrapposte che possono procedere metricamente allo stesso modo, ovvero rispettando la medesima figura musicale, oppure diversificando l'andamento, attraverso figure musicali differenti, creando effetto contrappuntistico.

I suoni di cui si compongono i brani, vengono definiti **note**, le cui qualità sono altezza, durata, intensità, timbro.

La **durata**, che distingue i suoni in lunghi e brevi, determina la velocità di un brano (es: largo, adagio, andante, allegro, moderato, presto ecc...). La durata della notazione attuale, si avvale dei seguenti simboli e relativi valori:

- semibreve, $\frac{4}{4}$
- minima, $\frac{2}{4}$
- semiminima, $\frac{1}{4}$
- croma, $\frac{1}{8}$
- semicroma, $\frac{1}{16}$
- biscroma, $\frac{1}{32}$
- semibiscroma, $\frac{1}{64}$

Anticamente erano presenti anche la fusa ($\frac{1}{128}$) e la breve ($\frac{8}{4}$).

Ai suddetti simboli corrispondono silenzi musicali definiti **pause**.

I valori prestabiliti delle figure musicali possono subire varianti dovute all'utilizzo di simboli aggiunti come uno o più punti, scritti dopo la nota. Il valore della figura viene così modificato:

- un punto ($\cdot$) prolunga il suono di metà del suo valore;
- un secondo punto ($\cdot\cdot$) aggiunge metà del valore del primo punto;
- un terzo punto ($\cdot\cdot\cdot$), meno utilizzato, aumenta metà del valore del secondo punto.

Altro simbolo di prolungamento del suono è la "legatura di valore", ovvero una linea curva che unisce due note aventi la stessa altezza, creando un valore derivato dalla somma delle singole durate.

L'**intensità**, invece, distingue i suoni in forti e deboli con varie diciture tecniche: piano, forte, fortissimo ecc...

Il **timbro** è la caratteristica che permette di distinguere la "personalità" vocale o strumentale di un suono.

L'**altezza** è la caratteristica musicale che distingue i suoni in acuti e gravi, attraverso la posizione delle note sul pentagramma. Le note Do, Re, Mi, Fa, Sol, La, Si, che in notazione anglosassone vengono identificate con C, D, E, F, G, A, B, possono subire modificazione di altezza attraverso ottavazioni diverse e utilizzo di alterazioni musicali (diesis o bemolle). Tali **alterazioni**, o accidenti musicali, producono variazioni di semitono ascendente o discendente, che possono ritornare al loro stato naturale con l'uso del bequadro.

In musica, il "**pitch**" è una variazione di frequenza della nota musicale, ridotta o aumentata di una certa quantità. Intuitivamente, l'effetto può essere banalmente riprodotto con un elastico teso fra le mani e pizzicato. Se durante la vibrazione l'elastico viene ulteriormente allungato, il suono prodotto diverrà più acuto; se esso viene rilasciato, il suono diverrà più grave.

Ad esempio, con una chitarra, si può ottenere un pitch shift verso il basso stirando la corda pizzicata, grazie all'utilizzo del tremolo, oppure verso l'alto, scorrendo il dito che ha pizzicato la corda sulla tastiera, ortogonale alla corda stessa. Con una tastiera elettronica per gli effetti di pitch shift è possibile usare una leva chiamata bender, o accedere alle specifiche funzioni di programmazione dello strumento.

Altre tecniche sono comunemente adottate per numerosi altri strumenti musicali. Le variazioni più tipiche del pitch sono di un semitono o di un tono, ma possono essere applicate variazioni anche superiori.

La **tonalità** risulta essere l'insieme di note, scale, accordi, arpeggi ecc... strutturati in un determinato ambito sonoro prestabilito, sulla base di una certa altezza.

Ogni brano viene impostato, ad esigenza del compositore, con riferimento a voci o strumenti, su una determinata tonalità che può variare per esigenze strumentali, ma soprattutto vocali. Tale altezza può subire modificazioni di intonazione ascendente o discendente, minimo $\frac{1}{2}$ tono.

Un brano musicale è composto da tante note: per posizionare questi simboli con ordine si divide il pentagramma con delle stanghette verticali che formano tanti piccoli "cassetti", chiamati **battute**. Ogni battuta contiene un numero definito di pulsazioni, precisato dal tempo musicale. Il tempo, infatti, indica la somma della durata di tutte le note e le pause che ci sono in ciascuna battuta. L'indicazione del tempo musicale compare all'inizio del brano, subito dopo la chiave.

Alcuni esempi di tempo più comunemente utilizzati possono essere $\frac{4}{4}$, $\frac{3}{4}$, $\frac{2}{4}$ (tempi semplici), oppure $\frac{3}{8}$, $\frac{6}{8}$, $\frac{9}{8}$, $\frac{12}{8}$ (tempi composti).

In uno spartito possono poi essere presenti gruppi di note irregolari, quali duina e quartina nei tempi composti, terzina e sestina nei tempi semplici.

Nell'ultima battuta di un brano ed a volte anche all'interno dello stesso, si può trovare il simbolo del ritornello (due punti sovrapposti seguiti da una doppia stanghetta che indica la ripetizione del brano). Quando il compositore esige di ripetere più di una volta la melodia, scrive il numero delle ripetizioni a fianco del segno di ritornello.

2.3 Metodologie per rappresentare la musica in formato digitale

Tra i modi per rappresentare la musica in formato digitale è necessario inserire lo standard MIDI, che costituirà l'analisi di partenza per affrontare il problema della conversione tra i formati MIDI e MusicXML.

L'altra metodologia utilizzata nel contesto corrente per rappresentare digitalmente la musica è lo standard MusicXML, formato di destinazione del processo di conversione che si è proposto.

In seguito verranno presentati e descritti analiticamente i suddetti standard ed i protocolli che essi utilizzano.

2.3.1 Lo standard MIDI

L'acronimo MIDI (*Musical Instrument Digital Interface* ovvero Interfaccia Digitale per Strumenti Musicali) serve ad indicare sia l'interfaccia hardware necessaria per la comunicazione tra strumenti, sia un protocollo di comunicazione, quindi un insieme di regole necessarie all'interpretazione univoca di tutti i messaggi.

Il protocollo MIDI nasce all'inizio degli anni '80 dalla necessità di rendere i nuovi strumenti digitali in grado di comunicare e di sincronizzarsi tra loro. L'obiettivo originario era riuscire a collegare due o più tastiere e pilotare tutto il sistema da una sola, ottenendo però il suono da entrambe. I primi sintetizzatori elettronici, infatti, generavano delle sonorità alquanto povere, ed il tentativo era quello di sommare più suoni tra loro in modo da ottenere un suono risultante più "pieno", dal contenuto armonico più complesso.

Presentato per la prima volta nel 1981 da Dave Smith e Chet Wood, due progettisti della Sequential Circuit, il prototipo dell'interfaccia MIDI ottiene una risonanza immediata a livello mondiale ed il lavoro di cinque delle più importanti aziende del settore musicale (Roland, Yamaha, Sequential, Korg e Kawai) porta alla nascita delle prime specifiche tecniche ufficiali del MIDI (1982). Da questa data, la maggioranza dei sintetizza-

tori e degli strumenti musicali elettronici inizia ad incorporare le porte MIDI.

Successivamente la IMA (*International Midi Association*) innalza il MIDI al rango di standard, pubblicando le specifiche 1.0 riguardanti sia l'hardware necessario che il software corrispondente. Tali specifiche ad oggi sono state più volte modificate, ampliando notevolmente il tipo di messaggi trasmessi e riconosciuti.

Rappresentare i dati tramite protocollo MIDI è diventata quindi un'alternativa attraente sia per musicisti e compositori, sia per quelle applicazioni musicali che devono produrre suoni, come possono essere le presentazioni multimediali ed i giochi per computer.

Tra i vantaggi dell'utilizzo di file in formato MIDI bisogna senz'altro inserire la dimensione estremamente piccola rispetto ai classici file audio. Esemplificando, file contenenti una grande quantità di audio campionato in modalità stereo richiedono circa 10 MBytes di dati per ogni minuto di suono, mentre una tipica sequenza MIDI occupa meno di 10 KBytes di dati per ogni minuto di suono. Questo si spiega in quanto i file MIDI non contengono i dati relativi all'audio campionato, ma solamente le istruzioni necessarie affinché un sintetizzatore sia in grado di replicare i suoni. Queste istruzioni vengono date sotto forma di messaggi MIDI che istruiscono il sintetizzatore su quali suoni utilizzare, quali sono le note da suonare e la relativa potenza.

Tra gli altri vantaggi dell'utilizzo di MIDI per generare suoni vi è senz'altro la possibilità di editare facilmente la musica, modificandone la velocità di riproduzione o la tonalità. In applicazioni come può essere una struttura di karaoke, quest'ultima possibilità risulta di rilevante importanza, in quanto tonalità e tempo metronomico della canzone possono essere selezionati facilmente dall'utente.

Ogni file MIDI o, più formalmente, ogni *Standard Midi File* (SMF), può essere definito come un archivio di dati ed informazioni codificate tramite uno o più byte. Inoltre esistono tre varianti di file SMF, ognuna delle quali corrisponde ad un formato preciso:

- **Formato 0**: il blocco d'intestazione è seguito da una sola traccia,

contenente tutte le informazioni relative agli eventi di tutte le tracce originarie del brano.

- **Formato 1**: il blocco d'intestazione è seguito da una o più tracce che vengono memorizzate singolarmente condividendo però gli stessi valori di tempo e metrica, in quanto la velocità del brano viene inserita nella blocco d'intestazione. Questo è il formato più usato, poiché consente una gestione multitraccia di un brano.
  Le singole tracce vengono suonate simultaneamente e, nel caso in cui sia presente più di una traccia MTrk, la prima successiva a quella d'intestazione è conosciuta come "traccia del tempo". Essa contiene solitamente i Dati Meta indicanti la suddivisione della misura, del tempo metronomico e della tonalità della scala. Le tracce successive contengono tutti gli altri eventi MIDI e, all'occorrenza, eventi Meta in caso di modifica dei valori dei Dati Meta iniziali.

- **Formato 2**: il blocco d'intestazione è seguito da una o più tracce che vengono gestite indipendentemente, anche per quanto riguarda metrica e tempo.

Il protocollo MIDI prevede di rappresentare le note numericamente, come mostrato nella Tabella 2.1.

In seguito verrà descritta la struttura di un file MIDI, indicandone gli elementi fondamentali e soffermandosi sugli aspetti più rilevanti ed analizzati durante lo sviluppo del progetto. I dati che seguono vengono presentati con una codifica esadecimale.

Le strutture

Ogni file MIDI si compone di complesse strutture che tecnicamente vengono definite come *track chunk*, o blocco-traccia.

La prima struttura, presente unicamente all'inizio del file MIDI, è formata sempre da 14 byte e si definisce come *Midi Track header chunk* (MThd), ossia *Blocco d'intestazione*. Questo blocco presenta una prima

Tabella 2.1: Codifica delle note secondo il protocollo MIDI

Octave	**Note Numbers**											
	C	*C#*	*D*	*D#*	*E*	*F*	*F#*	*G*	*G#*	*A*	*A#*	*B*
−1	0	1	2	3	4	5	6	7	8	9	10	11
0	12	13	14	15	16	17	18	19	20	21	22	23
1	24	25	26	27	28	29	30	31	32	33	34	35
2	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59
4	60	61	62	63	64	65	66	67	68	69	70	71
5	72	73	74	75	76	77	78	79	80	81	82	83
6	84	85	86	87	88	89	90	91	92	93	94	95
7	96	97	98	99	100	101	102	103	104	105	106	107
8	108	109	110	111	112	113	114	115	116	117	118	119
9	120	121	122	123	124	125	126	127				

parte di byte invariabili, seguiti da altri byte che dipendono dalle informazioni che vengono trasmesse. L'intestazione specifica infatti alcune informazioni di base relative ai dati presenti nel file.

Attraverso un editor esadecimale è possibile visualizzare il blocco d'intestazione, che risulterà nel modo seguente:

4D 54 68 64 00 00 00 06 00 0n 00 nn 0n nn

dove:

- **4D 54 68 64** definisce la struttura corrente come blocco d'intestazione ed è quindi invariabile.
- **00 00 00 06** indica la lunghezza in byte della parte restante del blocco d'intestazione.
- **00 0n** indica il tipo di Standard Midi File (per il formato 0 si ha n = 0, per il formato 1 si ha n = 1, per il formato 2 si ha n = 2).

- **00 nn** indica quante tracce di tipo MTkr sono presenti nel file MIDI.
- **0n nn** indica il numero di impulsi (risoluzione) per nota da $\frac{1}{4}$. Ogni impulso si definisce come "tick", o istante, mentre la risoluzione viene espressa tramite PPQN, ovvero *Pulse Per Quarter Note*.

Le strutture successive alla prima, definite come *Midi Track* (MTrk), contengono gli eventi MIDI. Ciascuna è composta da un'intestazione, formata sempre da 4 byte invariabili, seguita poi da altri 4 byte identificativi della traccia:

4D 54 72 6B 00 00 00 nn

dove:

- **4D 54 72 6B** definisce il blocco di traccia come tale.
- **00 00 00 nn** indica quanti byte successivi sono presenti sino al termine della traccia. Dopo i descritti 4 byte è sempre presente un *Tempo Delta*, espresso con uno o più byte, e tutti gli eventi costitutivi della traccia.

Gli eventi presenti all'interno delle tracce MTrk sono sempre separati dal Tempo Delta, dato temporale che esprime in byte il tempo che intercorre tra due singoli eventi e quindi rappresenta la durata in PPQN tra un evento e quello che lo segue.

Il formato delle tracce contenenti gli eventi MIDI è esattamente lo stesso indipendentemente dal formato del file (0, 1 o 2).

Tutti i messaggi presenti all'interno di ogni traccia sono trasmessi sotto forma di eventi e presentano la medesima struttura di base: la prima parte è costituita da un *byte di stato*, identificabile dal bit più significativo del numero binario che viene sempre posto a 1, consentendo un range di possibili valori compreso tra 80 e FF (corrispondenti a 128 e 255 in codifica decimale); la seconda parte del messaggio è costituita invece da uno o più *byte di dati*, identificabili dal bit più significativo del numero binario che viene sempre posto a 0 e che pertanto consente un range di possibili valori compreso tra 0 e 7F (0 e 127 in codifica decimale).

I dati Meta

I dati Meta sono contenuti all'interno dei cosiddetti Meta Eventi, e costituiscono informazioni aggiuntive di vario genere.

Ogni evento Meta è riconoscibile dal primo byte sempre uguale a FF. In seguito la scritta "len" si riferisce alla lunghezza dell'evento in questione, ovvero quanti byte di dati seguono. Le notazioni "text" e "data" si riferiscono ai byte di dati (testuali o meno) specificati dalla lunghezza.

I principali Meta Eventi sono:

- **FF 01 len text** - *Text Event*: contiene informazioni testuali di qualunque dimensione. É buona norma inserire un evento testuale all'inizio di ogni traccia, indicando il nome della traccia, una descrizione dell'orchestrazione ed ogni altra informazione che si ritiene rilevante. Gli eventi testuali possono occorrere anche in altri momenti durante l'esecuzione della traccia, ad esempio per trasferire il testo di una canzone o altre descrizioni.

- **FF 03 len text** - *Track Name*: indica il nome della traccia come informazione testuale.

- **FF 04 len text** - *Instrument Name*: descrive il tipo di strumentazione utilizzata nella traccia e può essere utilizzato insieme al messaggio di MIDI Channel Prefix per specificare a quale canale applicare la descrizione. In alternativa il canale può anche essere specificato testualmente nell'evento stesso.

- **FF 05 len text** - *Lyric*: indica il testo che deve essere cantato, considerando ogni sillaba come un evento separato che inizia al tempo corrispondente all'evento.

- **FF 20 01 cc** - *MIDI Channel Prefix*: indica il canale MIDI (da 0 a 15) al quale bisogna associare tutti gli eventi che seguono. Con "cc" si imposta quindi il canale da ritenere valido fino al prossimo

evento MIDI contenente un canale oppure fino al prossimo evento di MIDI Channel Prefix.

- **FF 2F 00** - *End of Track*: indica la fine di una traccia. É un evento obbligatorio che viene inserito per specificare il momento esatto in cui una traccia termina, in modo che questa ottenga una durata precisa, indispensabile per tracce ripetute oppure concatenate.

- **FF 51 03 tttttt** - *Set Tempo*: indica un cambiamento di tempo metronomico (velocità del brano), ovvero quanti microsecondi (tttttt) devono essere presenti per nota da $\frac{1}{4}$.

- **FF 58 04 nn dd cc bb** - *Time Signature*: indica il metro della misura, esprimendolo attraverso 4 numeri (nn dd cc bb).
  I valori "nn" e "dd" rappresentano rispettivamente numeratore e denominatore della misura così come sarebbe scritta in uno spartito. Il denominatore è espresso come potenza di 2, per cui un valore pari a 2 corrisponde a un denominatore pari a 4, un valore pari a 3 corrisponde a un denominatore pari a 8 e così via.
  Il parametro "cc" esprime il numero di clocks MIDI in uno scatto del metronomo. Infine, il parametro "bb" esprime il numero delle note in 32esimi che sono segnate, rispetto a ciò che il MIDI considera per note da $\frac{1}{4}$ (24 MIDI Clocks).

- **FF 59 02 sf mi** - *Key Signature*: indica la chiave di tonalità della scala, in cui "sf" equivale al numero di accidenti in chiave e "mi" definisce la tonalità (maggiore con mi = 00 oppure minore con mi = 01).

Gli eventi MIDI

Gli eventi MIDI possono essere considerati come le informazioni principali e costitutive di ogni file MIDI. Essendo il MIDI stato concepito inizialmente soprattutto per gli strumenti musicali a tastiera, i messaggi MIDI si riferiscono ad un'azione musicale applicata alla tastiera.

Se nel file sono presenti più tracce di tipo MTrk, gli eventi MIDI saranno presenti a partire da quella successiva alla traccia del tempo, poiché in quest'ultima saranno presenti i dati Meta.

I due principali eventi MIDI riguardano il momento in cui una nota inizia a suonare ed il momento in cui termina. In particolare:

1. **9n hh vv** - *Note ON*: viene generato ogni volta che si preme un tasto di nota ed è sempre seguito da un evento di Note OFF.

2. **8n hh vv** - *Note OFF*: viene generato nel momento in cui una nota suonata viene rilasciata, per cui segue sempre un evento di Note ON.

Per entrambi i messaggi:

- "n" sta ad indicare il numero del canale al quale applicare l'azione di Note ON o di Note OFF;

- "hh" definisce rispettivamente il numero della nota da far suonare oppure da far cessare;

- "vv" (velocity, intesa come forza applicata sullo strumento) indica la velocità di tocco nel caso di Note ON e la velocità di rilascio nel caso di Note OFF.

2.3.2 Lo standard MusicXML

In questo sottoparagrafo sarà presentato lo standard MusicXML, descrivendone la struttura ed i suoi elementi fondamentali.

A tale scopo è necessario effettuare una breve introduzione riguardante il meta-linguaggio XML.

Il linguaggio XML

Per XML (*Extensible Markup Language*) si intende un meta-linguaggio di markup, ovvero un linguaggio che permette di definire ulteriori linguaggi di markup. Nello specifico, XML si costituisce di un insieme di standard di regole sintattiche per modellare la struttura di dati e documenti.

Le specifiche ufficiali, definite dal *World Wide Web Consortium* (W3C), indicano la modalità secondo la quale è possibile definire un proprio linguaggio di markup.

Il linguaggio XML nasce dalla necessità di avere un linguaggio di markup che offra una maggiore libertà nella definizione dei tag, rimanendo tuttavia nell'ambito del rispetto di uno standard. Nato inizialmente con l'idea di rispondere ad una necessità evidenziata in un contesto Web, l'XML è stato poi adottato in numerosi altri contesti, come possono essere la definizione della struttura di documenti, la rappresentazione di immagini, la definizione di formati di dati o lo scambio di informazioni tra sistemi diversi.

Concretamente, un documento XML è un file di testo contenente una serie di tag, attributi e testo secondo alcune regole sintattiche ben definite. La struttura intrinseca di ogni documento XML è di tipo gerarchico ed è caratterizzata da un insieme di componenti denominati elementi, ciascuno dei quali rappresenta un componente logico del documento e, oltre che del testo, può contenere altri sottoelementi. Le proprietà degli elementi sono memorizzate sotto forma di attributi.

L'elemento principale (*root element*) contiene l'insieme degli altri elementi del documento. La struttura logica di ogni XML viene tradotta in una corrispondente struttura fisica che si compone di elementi sintattici denominati *tag* e tale struttura può essere facilmente implementata attraverso un file di testo creato con un qualunque editor.

Il formato MusicXML

MusicXML è un sistema di codifica XML ed un formato standard libero che nasce con l'obiettivo di facilitare lo scambio di partiture musicali.

Essendo stato progettato per la condivisione di spartiti musicali tra applicazioni, oltre che per l'archiviazione di questi per un uso futuro, il formato MusicXML vanta una grande leggibilità ed usabilità da parte di un'ampia gamma di applicazioni di notazione musicale.

La versione corrente del formato è la 3.0, rilasciata nell'Agosto 2011, e attualmente più di 180 applicazioni includono il supporto per MusicXML.

Derivando dal sistema di codifica XML, esso offre tutte le potenzialità di questo strumento. Una volta codificato, uno spartito in formato MusicXML può essere interrogato e rielaborato, ed è in grado di fornire facilmente le informazioni di interesse.

Gli spartiti musicali scritti attraverso la notazione occidentale possono essere interpretati sia in senso verticale, se si considera come fattore di riferimento le battute, sia in senso orizzontale, dove il fattore di riferimento è costituito dalle parti (ovvero le diverse linee melodiche).

MusicXML mette a disposizione due differenti approcci, in modo da consentire una lettura dello spartito in entrambe le modalità. Essi si distinguono dall'elemento radice, il quale può essere:

- <**score-partwise**>, nel caso di lettura in senso orizzontale, dove le battute sono contenute all'interno delle parti
- <**score-timewise**>, nel caso di lettura in senso verticale, dove le parti sono contenute all'interno delle battute

Le due codifiche sono tuttavia convertibili l'una nell'altra attraverso due fogli di stile XSLT forniti da MusicXML. In questo modo un'applicazione in grado di interpretare file in formato MusicXML può scegliere quale modalità utilizzare e, nel caso l'elemento radice del documento non corrisponda, può operare una trasformazione XSLT prima di procedere all'elaborazione del documento.

La maggioranza dei software attuali che gestiscono il formato MusicXML utilizzano la codifica *<score-partwise>*. Tale codifica sarà anche quella presa in esame durante tutto lo sviluppo del progetto.

Struttura dei file MusicXML

In seguito verrà analizzata la struttura generale di un file MusicXML, soffermandosi in dettaglio sugli elementi più utilizzati e di rilievo per lo scopo del progetto.

La struttura gerarchica dei file prevede che uno dei figli diretti dell'elemento radice sia il tag <**identification**>. Questo elemento viene in genere inserito come primo figlio della radice e contiene i metadata di base della partitura.

L'elemento figlio principale di <identification> è <**encoding**>: questo ha lo scopo di mantenere informazioni come chi ha effettuato la codifica digitale, con quale software ed in quale data. Queste informazioni sono memorizzate rispettivamente come <**encoder**>, <**software**> e <**encoding-date**>.

In seguito al tag <identification> è possibile trovare l'elemento <**part-list**>: esso contiene una lista delle tracce presenti successivamente, per le quali vengono definiti alcuni elementi di base. In ogni file MusicXML deve essere presente almeno un elemento <**score-part**>, il quale corrisponde esattamente ad una traccia identificabile in un file MIDI in Formato 1.

Per ogni traccia, il tool realizzato specifica un codice, attraverso l'attributo "id", ed un nome, utilizzando un elemento <**part-name**>.

In aggiunta alle suddette proprietà, il tag <score-part> può contenere anche informazioni riguardanti gli strumenti utilizzati nelle tracce ed i canali corrispondenti.

Terminata la parte di intestazione del file, vengono riportate analiticamente tutte le tracce presenti. Ognuna di esse è costituita da un elemento <**part**>, il quale presenta un attributo "id" corrispondente a quello definito nell'intestazione.

Il primo figlio diretto di <part> è costituito dall'elemento <**attributes**>, il quale è destinato a contenere informazioni generali sulla singola traccia, come possono essere la chiave ed il metro della misura.

La durata delle figure musicali viene spesso rappresentata attraverso frazioni. MusicXML cerca invece di specificare ogni durata utilizzando un unico numero, calcolato sulla base delle cosiddette "divisioni".

Il tool realizzato estrapola dal file MIDI alcune informazioni generali delle tracce, specificando il numero di divisioni per nota da $\frac{1}{4}$ attraverso il tag <**divisions**> ed il metro della misura attraverso il tag <**time**>. Quest'ultimo presenta due figli: l'elemento <**beats**> indica il numero di pulsazioni e corrisponde esattamente al numeratore della frazione presente all'inizio della partitura; l'elemento <**beat-type**> indica le unità di battito, ovvero il denominatore della frazione indicante il tempo.

In aggiunta a questi elementi, gli attributi delle tracce possono contenere informazioni sulla chiave del brano, sulle battute e su altri elementi grafici del pentagramma.

Ogni traccia presenta poi la sequenza di battute (<**measure**>), numerate attraverso l'attributo "number", ognuna delle quali contiene la lista delle note che vengono suonate.

Ogni nota (<**note**>) viene poi definita attraverso i seguenti sottoelementi:

- <**rest**/>: utilizzato per indicare una nota che in realtà corrisponde ad una pausa o ad un silenzio. Solitamente questo elemento è vuoto, ma è possibile specificare alcune caratteristiche grafiche come ad esempio la posizione sul pentagramma oppure se è una pausa dell'intera durata della battuta.

- <**pitch**>: rappresenta una combinazione di tre elementi, l'altezza della nota (<**step**>) espressa in notazione anglosassone, l'alterazione (<**alter**>) che può essere o meno presente, e l'ottava della nota (<**octave**>).

- <**duration**>: utilizzato per indicare la durata di una nota (o di una pausa). Si presenta come un intero positivo calcolato sulla base del numero di divisioni specificato all'inizio del documento.

- <**type**>: indica il tipo "grafico" della nota o della pausa, ovvero se si tratta di una minima, una seminimina, una croma, una semicroma e così via.

- <**chord**/>: utilizzato per indicare che la nota in questione fa parte di un accordo. É un elemento vuoto che viene inserito a partire dalla seconda nota che forma l'accordo.

L'elemento <note> può contenere anche molti altri sotto-elementi che ne descrivono particolari caratteristiche. Ad esempio, possono esservi specifiche riguardanti accidenti, legature di valore o di portamento, gruppi irregolari, ornamenti, armonia, tecniche di esecuzione (pizzicato, staccato, legato, trattenuto, appoggiato...), proprietà di particolari strumenti e caratteristiche grafiche della rappresentazione su pentagramma.

Questi dettagli non verranno presi in considerazione durante lo sviluppo del progetto in quanto non di fondamentale importanza per lo scopo del lavoro. Pur costituendo l'oggetto principale che verrà analizzato, le informazioni di interesse da estrarre dall'elemento <note> riguardano prevalentemente il tipo di nota (o di pausa) e la durata.

2.4 La musica come reti complesse

Ogni brano musicale, ogni melodia, ogni singolo assolo può essere pensato come un sistema di entità (in questo caso le note) interconnesse tra di loro. Ciò rispecchia pienamente il concetto di rete, in particolare di rete complessa, inteso come insieme di elementi che si scambiano informazioni secondo una struttura ben definita.

Le note, e quindi anche le pause, che formano una composizione musicale possono essere viste come i nodi che costituiscono la rete. Analogamente, le sequenze con cui le note vengono suonate e sono inframmezzate da pause rispecchiano i collegamenti tra i singoli nodi della rete.

In un'ottica di rappresentazione del brano musicale come una rete, i nodi possono essere costituiti dai seguenti elementi:

- note con altezza differente
- note con stessa altezza, ma durata differente

- pause con diversa durata

Un semplice esempio di rete realizzata partendo da due battute di un brano è mostrato in Figura 2.5.

La nota Do (C) è rappresentata da un unico nodo a cui si accede sia all'inizio del brano, sia successivamente dopo aver suonato un Sol (D). Analogamente, la nota Si (D) vede la presenza di un unico nodo al quale si accede sia dopo aver suonato un Do, sia dopo aver suonato la nota stessa (motivo per il quale è presente un auto-anello). Per rappresentare la nota Sol (G) sono stati invece utilizzati due nodi differenti poiché le note in questione presentano durate differenti: una croma di durata $\frac{1}{8}$ ed una minima di durata $\frac{2}{4}$, ovvero $\frac{1}{2}$. Infine, il nodo "rest" rappresenta la pausa di croma di durata $\frac{1}{8}$.

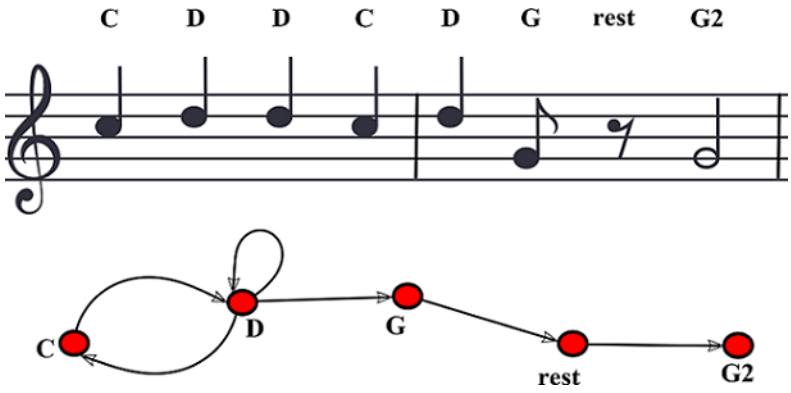


Figura 2.5: Esempio di partitura musicale rappresentata tramite rete. Autore: Stefano Ferretti

Lo studio di partenza su cui si è basato l'intero lavoro si focalizza specialmente sugli assoli musicali, intesi come parte di un brano in cui un musicista suona e spesso improvvisa una melodia, accompagnato o meno dagli altri strumenti.

Nella teoria della musica è molto comune affermare che gli assoli prodotti da un musicista siano legati alle loro capacità tecniche ed artistiche. Perciò, i musicisti possono essere identificati grazie al loro "stile" personale durante l'esecuzione di un assolo all'interno di un pezzo musicale. Non a caso un artista può essere riconosciuto dagli altri ed è possibile classificarli in categorie o gerarchie.

Se si pensa a modellare una linea musicale come una rete complessa di note e pause che si susseguono, è possibile far emergere proprietà della rete dovute alle interazioni di questi elementi. Le reti complesse sono in grado di fornire una modellazione appropriata ed un potente quantitativo di metriche atte a catturare l'essenza della complessità.

Sono stati recuperati ed analizzati differenti assoli di alcuni dei più rinomati chitarristi, come Eric Clapton, David Gilmour, Jimi Hendrix, B.B. King, Eddie Van Halen. Il motivo per cui tra tutti i possibili strumenti è stato scelto la chitarra deriva dalla presenza di una grande e attiva comunità di chitarristi disponibili a condividere partiture musicali sul Web.

Gli spartiti vengono pubblicati e formattati, solitamente, utilizzando schemi di descrizione alternativi, come MusicXML, più semplici ed intuitivi da leggere ma che rispecchiano la partitura originale.

L'analisi dei suddetti assoli di chitarra ha predisposto il calcolo delle metriche tipiche delle reti complesse. Sono stati misurati la lunghezza dell'assolo, la dimensione della rete, il grado di distribuzione, metriche relative alla distanza, il coefficiente di clustering e la possibilità di identificare le rappresentazioni di alcuni assoli come reti di tipo Small World.

A titolo di esempio, vengono riportate le rappresentazioni grafiche di due degli assoli di chitarra analizzati.

In Figura 2.6 è mostrata la rete derivante dall'assolo di B.B. King, Rock Me Baby, mentre la Figura 2.7 rappresenta l'assolo di Jimi Hendrix, Red House.

Come si riesce ad intuire dai grafi, il primo assolo corrisponde ad una rete relativamente semplice, con alcuni nodi aventi un in/out degree maggiore di altri. Differentemente, il secondo assolo produce una rete dalla struttura alquanto più complessa, con un numero elevato di nodi e

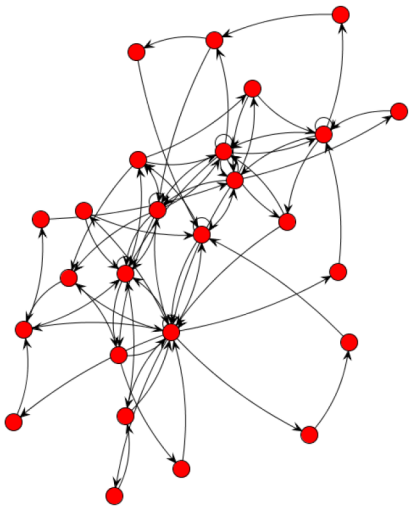


Figura 2.6: B.B. King - Rock Me Baby (da: "*Guitar Solos As Networks*" - Stefano Ferretti, 2016)

con la presenza di diversi hub, ovvero nodi con un numero di collegamenti assai più elevato della media.

Attraverso l'uso di ulteriori misure aggregate si è cercato di ottenere una visuale generale degli assoli, dei quali si considera la durata come caratteristica fondamentale. Infatti, durate differenti hanno un impatto non trascurabile su alcune delle metriche quantitative adottate. Ad esempio, la durata influisce sul numero di note che vengono suonate e quindi sul numero di nodi presenti nella rete.

Metriche di questo tipo possono esprimere quanto un musicista sia incline ad elaborare le melodie che crea durante l'assolo, nonostante sia comunque un fattore legato anche allo specifico genere musicale.

Nei due casi riportati in precedenza, è osservabile quanto Jimi Hendrix abbia un dizionario di note molto più ricco rispetto a B.B. King.

Se si tiene conto della distribuzione dei nodi è facile osservare come, nel secondo assolo, alcuni nodi siano maggiormente "attraversati" rispetto ad altri. Ciò conferma il fatto che una delle tendenze di B.B. King era

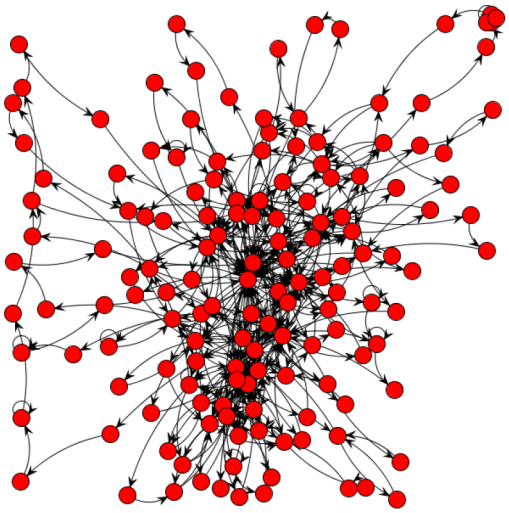


Figura 2.7: Jimi Hendrix - Red House (da: "*Guitar Solos As Networks*" - Stefano Ferretti, 2016)

suonare e ripetere specifiche combinazioni di note.

Valutando poi la distanza media tra i nodi, intesa come il percorso di lunghezza media in una rete, si è notato che in genere, musicisti blues come B.B. King producono assoli con una distanza minore rispetto a musicisti rock.

Inoltre, è stato calcolato il coefficiente di clustering delle reti nell'ottica di scoprire la presenza di "triangoli" di nodi: in questo contesto, un triangolo di nodi significa che il musicista ha suonato durante l'assolo le tre note corrispondenti in sequenza, ma con diverso ordine.

In altre parole, la presenza di molti triangoli (mostrata da un alto coefficiente di clustering) può indicare la preferenza di un musicista a seguire una specifica linea melodica, rispettando un ordine preciso con cui suonare le tre note.

Infine è giusto accennare il tentativo di verificare la presenza o meno di reti di tipo Small World.

I relativi calcoli hanno dimostrato come, tra i due casi presentati in

precedenza, la rete derivante dall'assolo di Jimi Hendrix (Red House) può essere classificata come di questo tipo.

Capitolo 3

Approccio proposto

In questo capitolo verrà affrontato nel dettaglio il problema della conversione di file da formato MIDI a MusicXML. Sarà presentato il modo con cui si è operato per risolvere il problema, soffermandosi sulle maggiori criticità e sulle strumentazioni che sono state utilizzate.

Si è pensato di costruire un progetto scritto in Java, avvalendosi dell'ambiente di sviluppo integrato sviluppato da JetBrains, quale IntelliJ IDEA.

3.1 Analisi del file MIDI

In primo luogo, è stato necessario studiare la struttura di un file MIDI attraverso l'utilizzo di un editor esadecimale. Questo passaggio ha consentito di verificare con esattezza quali fossero i dati presenti che sarebbe stato necessario elaborare nel progetto.

Durante lo sviluppo si è fatto uso delle librerie Java fornite da Oracle, in particolare del package *javax.sound.midi*, in quanto in grado di fornire una rappresentazione efficiente dell'architettura di un file MIDI. In assenza di tale libreria si sarebbe resa necessaria un'analisi del file MIDI partendo direttamente dal codice esadecimale di cui esso si compone.

In un'ottica mirata alla rappresentazione di melodie musicali come reti complesse, gli elementi fondamentali che è stato necessario estrarre sono le note, le cui caratteristiche di maggiore interesse riguardano:

- il "pitch", comprensivo di altezza, ottava e alterazione;
- la durata
- il tipo di nota (minima, semiminima, croma, ecc...)

I suddetti elementi possono essere ricavati dagli eventi MIDI di Note ON e Note OFF.

La libreria Java utilizzata modella tali eventi attraverso una classe definita "**ShortMessage**": in essa vengono incorporati i messaggi MIDI che contengono al massimo due byte di dati a seguito del byte di stato. Questa classe fornisce metodi per ottenere e settare i contenuti di ogni messaggio MIDI di questo tipo.

Per ottenere, quindi, la sequenza di tutte le note presenti nel file, si è operato come segue.

É stato necessario recuperare dal file MIDI la lista di tutte le tracce presenti, attraverso i metodi appositi presenti all'interno della libreria.

Di ogni traccia sono stati poi intercettati tutti gli eventi MIDI, controllando per ognuno di essi se fosse di tipo Meta oppure se contenesse un "ShortMessage" identificabile come evento di Note ON o Note OFF.

A tal proposito, è importante specificare che il momento in cui una nota termina non sempre è codificato tramite un evento di Note OFF. Il protocollo MIDI, infatti, in un'ottica mirata all'ottimizzazione, spesso prevede di sostituire gli eventi di Note OFF con eventi di Note ON con *velocity* pari a 0. Il risultato è il medesimo, ma viene utilizzato un minor numero di byte e ciò consente un risparmio di memoria.

Una volta identificati tali eventi, dal messaggio vengono estratte ed opportunamente elaborate le informazioni contenute nei byte di dati, quali altezza, ottava e alterazione.

Per quanto riguarda la durata, nei byte di dati non è presente nessun elemento che esprima direttamente questo valore, per cui si è reso necessario calcolarlo come il tempo intercorrente tra un evento di Note ON ed il relativo evento di Note OFF.

La libreria fornisce un metodo in grado di catturare il momento esatto di inizio di un evento, in termini di "**tick**", o impulsi. Quindi, il numero di tick corrispondenti alla durata di ogni nota è stato calcolato come differenza dei due valori iniziali degli eventi.

Come accennato nel capitolo precedente, nell'intestazione di ogni file MIDI è presente un valore indicante il numero di impulsi per nota da $\frac{1}{4}$. Grazie a questo valore è stato possibile capire il tipo di ogni nota, confrontandolo col numero impulsi calcolati per ogni nota.

In ogni spartito, le note sono suddivise lungo più battute, ognuna delle quali ha una durata corrispondente al tempo indicato all'inizio della partitura.

Sebbene non di fondamentale importanza per lo scopo del progetto, si è pensato di riprodurre la struttura delle battute anche nel file MusicXML di destinazione. Tuttavia, la libreria utilizzata non fornisce un supporto per l'identificazione delle battute, per cui si è operato autonomamente. L'algoritmo utilizzato a tal proposito prevede una scansione della lista di tutte le note del brano ed il calcolo della somma delle relative durate fino ad ottenere un valore corrispondente alla lunghezza di una battuta. Le battute sono quindi memorizzate in una struttura appositamente implementata ed ognuna di esse contiene solo uno specifico numero di note. É possibile che una nota faccia parte di due battute, se la sua durata eccede la lunghezza della prima: in questo caso la nota è stata memorizzata su entrambe le battute, ma col valore di durata corrispondente ad ognuna.

Durante questa analisi è stato possibile verificare (sebbene anche questo non sia un fattore essenziale) se una nota fa parte di un accordo, nel qual caso una nota può cominciare a suonare sia simultaneamente a un'altra sia prima che quest'ultima abbia terminato.

3.2 Creazione del file MusicXML

Per implementare la creazione di un file XML si è fatto uso di un package, sempre tra le librerie fornite da Oracle, definito *javax.xml*. Nello specifico sono stati utilizzati *javax.xml.parsers* e *javax.xml.transform*.

Per definire gli elementi costitutivi del file MusicXML di destinazione è stato utilizzato un ulteriore package quale *org.w3c.dom*. Esso fornisce un'interfaccia ottimale per il Document Object Model (DOM).

Per DOM si intende un modello in grado di descrivere come i diversi oggetti di una pagina siano collegati tra di loro. Il DOM è indipendente dalla piattaforma, in quanto è un'interfaccia definita dal W3C (*World Wide Web Consortium*) per essere uno strumento universale per tutti i creatori di pagine Web.

Per creare il documento finale è stato necessario definire una parte di intestazione con all'interno i seguenti dati:

- autore del file;
- software utilizzato per la creazione del file;
- data di codifica;
- lista di tutte le tracce che verranno descritte in seguito, ognuna delle quali caratterizzata da un id e da un nome.

Dopo l'intestazione segue tutta la lista delle tracce, definite come "parti", all'inizio delle quali vengono riportate le informazioni generali sulla traccia, recuperate durante l'analisi del file MIDI. Nello specifico, tra gli attributi di ogni traccia vanno inseriti:

- il numero di divisioni, sulla base del quale si calcola la durata di ogni nota;
- la descrizione del tempo del brano, intesa come la frazione presente all'inizio di ogni partitura, per la quale vengono salvati separatamente numeratore e denominatore.

La definizione degli attributi è seguita dall'elenco di tutte le battute presenti nella traccia e precedentemente calcolate. Ogni battuta contiene quindi la lista degli elementi "nota" con le rispettive proprietà.

3.3 Architettura del progetto

Si è scelto di utilizzare il pattern MVC (*Model View Controller*) in modo da scorporare la logica del modello dall'implementazione realizzata dal controller e quindi anche dalla rispettiva view.

Nell'ottica di creare un file MusicXML dal quale estrarre le informazioni essenziali per costruire una rete complessa, il **modello** del progetto contiene le seguenti classi, con le rispettive interfacce:

- ***Note***: questa è la classe principale che modella ogni singola nota e che quindi deve contenere tutte le relative informazioni. I campi principali sono costituiti da altezza, ottava, alterazione e durata della nota. Essendo la durata calcolata come tempo intercorrente tra l'evento di Note ON e l'evento di Note OFF, sono stati inseriti anche due campi contenenti gli impulsi (tick) dei rispettivi eventi. Un campo booleano è stato aggiunto inoltre per memorizzare se la nota in questione in realtà è una pausa o un silenzio.

- ***Signature***: siccome durante l'esecuzione di un brano possono esservi delle variazioni di tempo, descritte dai relativi Meta Eventi, si è scelto di modellare i diversi valori assunti dal tempo attraverso questa classe. Essa si compone di tre campi fondamentali, quali numeratore e denominatore della frazione indicante il tempo ed il momento (tick) in cui si comincia ad utilizzare questa notazione temporale (per il tempo definito all'inizio del brano questo campo assumerà valore 0).

- ***Measure***: questa classe modella il concetto di battuta, per cui il campo principale è costituito dalla lista delle note in essa contenute. Siccome il tempo di un brano può cambiare durante la sua esecuzione,

per ogni battuta viene anche memorizzata la frazione temporale corrispondente (attraverso due campi, uno per il numeratore ed uno per il denominatore). Per facilitare alcuni calcoli, in ogni battuta è inserito anche il numero di impulsi che devono esservi contenuti.

Il controller è stato implementato attraverso due classi principali e le rispettive interfacce:

- ***MidiAnalyzer***: questa classe ha il compito di analizzare il file MIDI, estraendone tutte le informazioni di interesse e creando le strutture atte alla loro memorizzazione. É la classe che utilizza il package di libreria *javax.sound.midi*, servendosene per ricavare i dati in modo più semplice, veloce ed intuitivo.
  Entrando brevemente nei dettagli implementativi, questa classe fa largo uso delle *Java Collections* e di tutte le metodologie di gestione ad esse collegate.

- ***MusicXmlConverter***: il compito di questa classe è invece quello di creare il file MusicXML di destinazione utilizzando i dati ricavati dal file MIDI originario grazie alla classe descritta in precedenza. Essa utilizza i package di libreria *javax.xml.parsers*, *javax.xml.transform* e *org.w3c.dom*, i quali consentono una creazione più immediata del file XML.
  Essendo l'interfaccia grafica del progetto composta esclusivamente da due pulsanti, si è scelto di gestire in questa classe anche le azioni relative alla pressione di entrambi, attraverso l'implementazione di due metodi appositi.

É stata implementata anche una classe "***Utility***", presente nell'omonimo package, in grado di fornire i metodi statici per la creazione degli elementi XML opportuni da inserire nel file di destinazione.

Il tool che si propone di realizzare ha il compito di effettuare una conversione in formato MusicXML di un file MIDI scelto dall'utente. Per questa ragione, l'interfaccia grafica che si propone risulta poco interattiva

Figura 3.1: MainView del programma

ed è costituita essenzialmente da un'unica ***MainView***, presentata in Figura 3.1.

Questa classe rappresenta l'interfaccia grafica che viene mostrata all'utente e, come accennato in precedenza, si compone di due pulsanti principali:

- "***Choose File***": alla pressione viene mostrata all'utente una finestra in cui è possibile selezionare il file che si desidera convertire navigando il file system (come mostrato in Figura 3.2).

- "***Convert***": questo è il pulsante che ha il compito di avviare tutto il meccanismo di conversione del file. Il file convertito verrà salvato nella stessa cartella in cui è presente il file di partenza selezionato dall'utente.

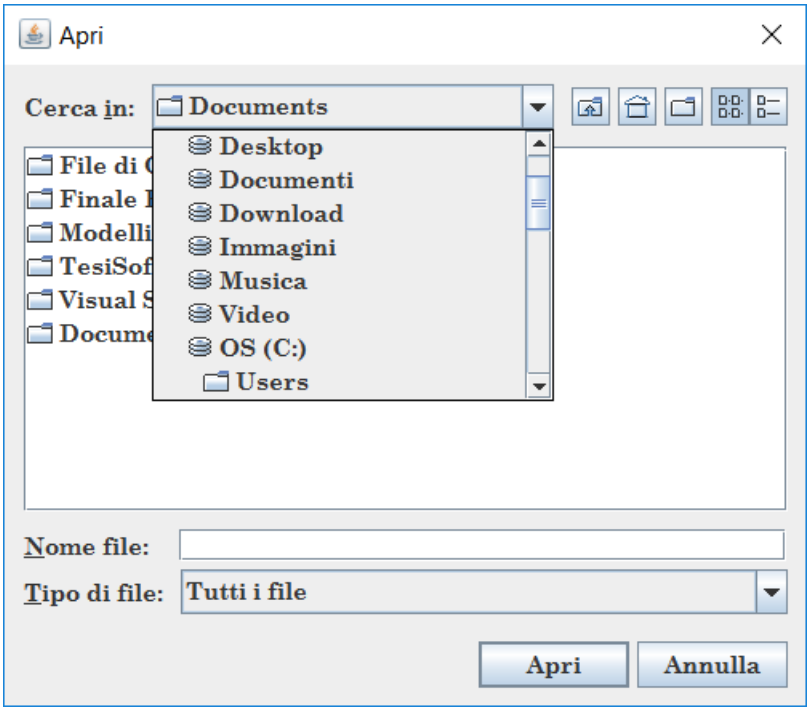


Figura 3.2: Finestra per la scelta del file da convertire

L'immagine di sfondo dell'interfaccia grafica è stata recuperata online, in particolare dal sito *www.spyderonlines.com*, mentre i due loghi presenti corrispondono agli standard MIDI e MusicXML.

A conversione ultimata, se non vi sono stati errori, viene mostrato all'utente un messaggio di fine conversione, di cui in Figura 3.3.

Sono state intercettate e gestite le possibili eccezioni derivanti da tutto il procedimento di conversione. Nel caso di eccezioni durante il caricamento del file da convertire oppure di eccezioni durante la conversione

Figura 3.3: Messaggio di fine conversione

e la creazione del file di destinazione vengono visualizzati all'utente due messaggi di errore di cui si riporta un esempio in Figura 3.4.

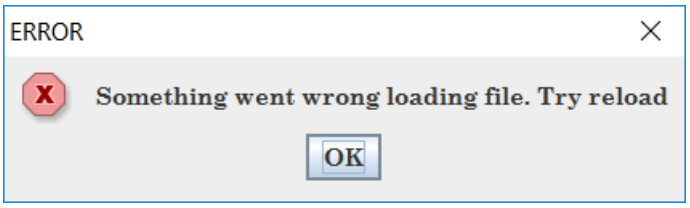


Figura 3.4: Messaggio di errore nel caricamento del file

3.3.1 Note sul procedimento di conversione

Così come ogni file MIDI presenta nella traccia d'intestazione un valore indicante il numero di impulsi per nota da $\frac{1}{4}$, così anche lo standard MusicXML prevede che all'inizio di ogni parte o traccia sia indicato il numero di "divisioni" per nota da $\frac{1}{4}$.

Questo valore è un numero intero che è stato necessario calcolare sulla base dei tick indicati dal file MIDI. Si è cercato di ottenere un valore che potesse generare una durata delle note intera, verificandone la compatibilità con tutte le note presenti nel brano.

Inizialmente la durata delle note è stata memorizzata in impulsi, in quanto si sono calcolati gli impulsi esistenti tra l'evento di Note ON e l'evento di Note OFF. Per cui si è reso necessario in un secondo momento esprimere la durata di ogni nota in termini di divisioni. Questo sarà il

valore che effettivamente verrà inserito come durata delle note nel file MusicXML di destinazione.

Le pause presenti nel brano sono state identificate calcolando il tempo intercorrente tra il momento in cui una nota cessa di suonare ed il momento in cui inizia una nuova nota. Se è presente una pausa, questo valore di differenza sarà maggiore di 0.

La durata delle figure di pausa e quindi anche il relativo tipo sono stati definiti in modo analogo a quanto spiegato precedentemente per le note.

Capitolo 4

Valutazione sperimentale

In questo capitolo verranno presentati alcuni casi pratici presi in esame, mostrando i risultati ottenuti e le eventuali diversità rispetto ai risultati attesi.

Verrà inoltre effettuata un'analisi delle caratteristiche delle reti derivanti dai diversi elementi. A tal proposito si specifica che sono state utilizzate librerie Java come *Apache Commons* (in particolare la componente *Math*), *EPSGraphics* e *JUNG*. Tali librerie open source si sono rese necessarie per la creazione e l'analisi dei grafi relativi alle reti derivanti dagli spartiti musicali. Grazie a questi strumenti è stato possibile ottenere rappresentazioni grafiche accurate che hanno consentito una più efficace analisi delle reti in questione. In particolare, la libreria *JUNG* consente la manipolazione di reti ed il calcolo di tutte le metriche atte all'analisi delle diverse proprietà; *Apache Commons* fornisce funzionalità matematiche che sono state utilizzate per il calcolo di numerose statistiche.

4.1 Conversione MIDI - MusicXML

Si coglie l'occasione per effettuare alcune osservazioni.

La rappresentazione grafica di un brano, sotto forma di spartito musicale, non è sempre visualizzata allo stesso modo, in quanto i software

possiedono potenzialità diverse. A questo proposito, occorre fare una precisazione in merito alla possibilità automatica o voluta dall'utente, di regolarizzare la metrica musicale attraverso il processo di quantizzazione, presente nel menu del programma.

Quando il musicista scrive, con l'ausilio di una tastiera MIDI (anche senza suoni al suo interno), può nella prima fase di lavoro suonare le note senza la dovuta precisione relativamente alla loro durata. Tali imprecisioni possono venire interpretate come silenzi e quindi considerate come pause a seconda dell'impostazione di una diversa quantizzazione.

In tal caso, ci si avvale della funzione "*Quantize*", ormai presente in tutti i "Sequencer", che provvede alla sistemazione grafica, dal punto di vista metrico musicale, offrendo al compositore una partitura perfetta. É buona pratica impostare la quantizzazione sui valori musicali più piccoli contenuti nel brano, riuscendo, in tal modo, a regolarizzare anche le durate maggiori.

Il tool che è stato realizzato si propone di prescindere dallo spartito grafico. In assenza di quantizzazione, fornisce una rappresentazione del brano spesso più precisa di quella che può derivare da un qualsiasi spartito musicale, in quanto si basa sull'analisi diretta del file MIDI corrispondente.

4.1.1 Caso 1: Midi creato ad hoc

Durante lo sviluppo del progetto, al fine di facilitare in parte l'analisi degli elementi del file MIDI, è stato creato un MIDI ad hoc, attraverso il "Sequencer" *Digital Performer* su piattaforma Apple.

Si riporta in Figura 4.1 lo spartito musicale del brano creato.

I valori esadecimali della traccia d'intestazione risultano essere i seguenti:

4D 54 68 64 00 00 00 06 00 01 00 1E 01 E0

per cui ci si aspetta di trovare un file MIDI in Formato 1, 30 tracce di tipo MTrk ed un numero di impulsi per nota da $\frac{1}{4}$ pari a 480. In realtà, solamente una delle 30 tracce MTrk contiene degli eventi identificativi di

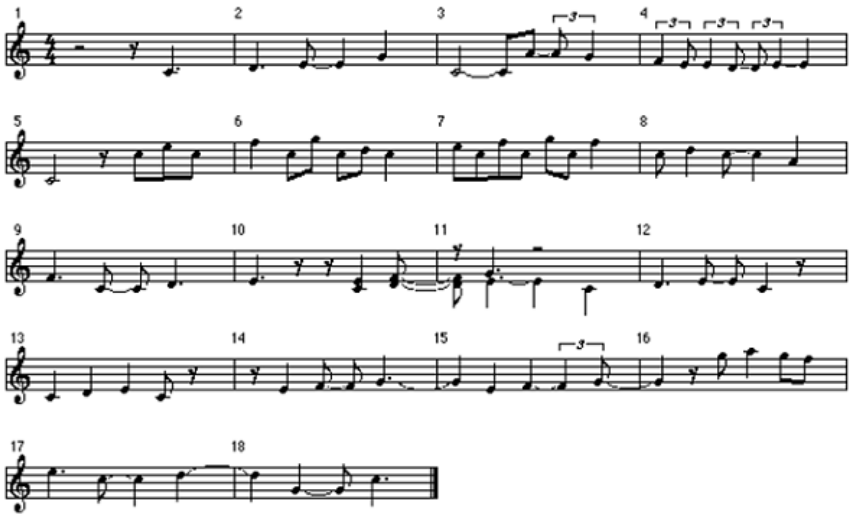


Figura 4.1: Partitura del file MIDI creato appositamente per il progetto

note suonate, per cui il file MusicXML di destinazione conterrà un'unica parte. Le restanti sono formate esclusivamente da Meta Eventi contenenti informazioni non di interesse.

Il procedimento di conversione in file MusicXML porterà quindi ad una parte di intestazione mostrata in Figura 4.2.

La traccia 1 conterrà l'elenco di tutte le note e le pause presenti nel brano.

A titolo di esempio si riporta in Figura 4.3 la codifica ottenuta per la prima battuta del brano.

É possibile notare che le due pause iniziali sono state codificate attraverso un'unica pausa della durata totale.

La nota possiede una notazione puntata che aumenta la durata di metà del suo valore, per cui, anziché $\frac{1}{4}$, la durata sarà di $\frac{3}{8}$. Essendosi reso necessario effettuare alcune semplificazioni, il tipo della nota è stato "approssimato per eccesso" impostandolo come se si trattasse di una minima.

Come mostrato, le durate delle figure sono state calcolate sulla base

```
<identification>
  <encoding>
    <encoder>Sofia Rosetti</encoder>
    <software>MidiToXmlConverter</software>
    <encoding-date>2016-09-08</encoding-date>
  </encoding>
</identification>
<part-list>
  <score-part id="P1">
    <part-name>Traccia 1</part-name>
  </score-part>
</part-list>
```

Figura 4.2: Intestazione del file MusicXML generato attraverso il tool

del numero di divisioni definito all'inizio della parte, in questo caso 228. Le due pause, codificate come una sola, avranno quindi una durata complessiva di 570 (corrispondente ai $\frac{5}{8}$ della battuta), mentre la nota sarà di 342.

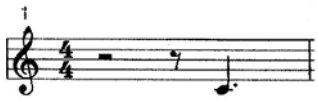
Il file MusicXML che è stato ottenuto effettuando una conversione con il tool realizzato rispetta le caratteristiche fondamentali per la modellazione come rete complessa. Ciò può risultare intuitivamente dall'osservazione del grafo della rete (Figura 4.5) che è possibile ottenere grazie ad alcune librerie appositamente implementate.

4.1.2 Caso 2: Midi di brani di repertorio

A verifica del tool realizzato, sono stati convertiti i file MIDI (recuperati su Web) di alcuni brani di repertorio, per i quali sono poi state svolte le analisi derivanti dalla modellazione tramite rete.

Sono stati scelti brani di diverso genere e con divisione del tempo differente ($\frac{4}{4}$, $\frac{3}{4}$ o $\frac{3}{8}$), col tentativo di verificare la correttezza del lavoro svolto. In particolare, si è rivolta l'attenzione alle seguenti composizioni:

- "*Per Elisa*", Ludwig van Beethoven



(a) Battuta 1 nello spartito

```
<measure number="1">
  <note>
    <rest/>
    <duration>570</duration>
    <type>whole</type>
  </note>
  <note>
    <pitch>
      <step>C</step>
      <octave>4</octave>
    </pitch>
    <duration>342</duration>
    <type>half</type>
  </note>
</measure>
```

(b) Battuta 1 nel file MusicXML

Figura 4.3: Codifica della prima battuta del brano

- "*Sultans Of Swing*", Dire Straits
- "*Yesterday*", The Beatles

In seguito alle analisi delle reti derivanti dalla trasformazione del file MusicXML, è stato possibile effettuare le seguenti valutazioni.

Tutte le reti generate presentano una struttura adatta alla descrizione del brano che rappresentano. Infatti, brani più lunghi e complessi, come "*Sultans Of Swing*" dei Dire Straits, producono reti caratterizzate da un numero molto elevato di nodi.

Alcune delle reti prese in esame possiedono valori di **Betweenness centrality** decisamente elevati. A conferma di ciò, i corrispondenti brani sono caratterizzati dalla presenza di un certo numero di note e accordi che rivestono un'importanza maggiore rispetto ad altri.

Ogni musica presenta almeno una melodia (o tema musicale) che la caratterizza e che viene ripetuta più volte nel corso del brano. Il numero

delle ripetizioni del tema incide sulla capacità di riconoscimento del brano da parte dell'ascoltatore.

Un esempio a tale proposito è il famoso foglio d'album "*Per Elisa*" di Beethoven, il cui tema d'apertura (mostrato in Figura 4.4) presenta numerosi ritornelli corrispondenti alla parte A del brano, il quale essendo un "*Rondò*" ha la forma A-B-A-C-A.

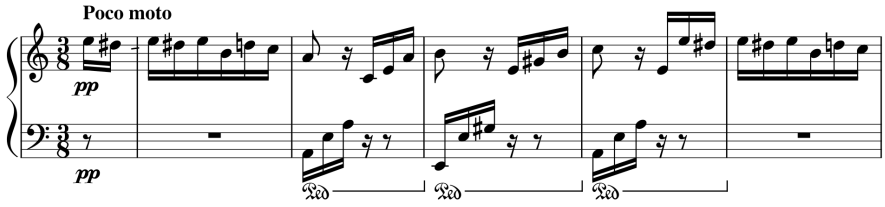


Figura 4.4: Tema principale del foglio d'album "*Per Elisa*"

Le note presenti nei temi più ricorrenti durante l'esecuzione di una musica producono quindi nodi con un valore di **Eigenvector centrality** più elevato rispetto agli altri. Ciò è stato positivamente verificato tra i brani analizzati. Infatti, i valori di questa metrica per il brano "*Per Elisa*" risultano sostanzialmente maggiori rispetto agli altri in cui, per l'appunto, il motivo principale non viene ripetuto così frequentemente.

É stato infine verificato per ogni brano analizzato, se la rete corrispondente fosse di tipo **Small-World**.

Sono stati confrontati i valori del coefficiente di clustering della rete con i valori di grafi random di uguali dimensioni: se la differenza tra i due valori è alta e se è presente un cammino minimo relativamente corto, come nei grafi random, allora la rete in questione presenta le caratteristiche per essere di tipo Small-World.

Nei grafi random e nelle reti Small-World, la lunghezza media dei cammini è assimilabile al logaritmo del numero dei nodi. Per cui, se:

- il coefficiente di clustering della rete è nettamente maggiore del coefficiente di clustering di un grafo random di uguali dimensioni;

- la lunghezza media dei cammini nella rete è pressoché uguale alla lunghezza media dei cammini in un grafo random

la rete in questione è di tipo Small-World.

Le condizioni appena descritte sono rispettate dalla rete corrispondente al brano "*Yesterday*", dei Beatles, per cui è possibile classificarla come rete Small-World.

Le rappresentazioni grafiche delle reti relative a tutti brani presi in considerazione sono riportate a fine capitolo.

4.2 Analisi di assoli musicali modellati tramite reti

É stato poi svolto un lavoro mirato ad analizzare assoli musicali di dieci strumenti differenti, con l'obiettivo di evidenziarne le caratteristiche principali ed eventualmente verificarne analogie e/o differenze. Gli strumenti in questione sono:

- Saxofono contralto
- Saxofono baritono
- Saxofono tenore
- Liuto
- Organo
- Pianoforte
- Tromba
- Violino
- Viola

- Vibrafono

Il lavoro è stato articolato come segue.

Innanzitutto sono stati reperiti online i file midi dei differenti assoli, provvedendo ad effettuare un'estrazione della singola traccia dove fosse necessario, tramite il software Finale Print Music. Attraverso il tool realizzato sono stati poi ottenuti i file in formato MusicXML relativi, dai quali è stato quindi possibile estrarre le rappresentazioni grafiche di rete corrispondenti.

Gli indicatori di centralità descritti nel secondo capitolo identificano quali sono i vertici più importanti all'interno di un grafo. Applicati alle reti di questo contesto, essi forniscono indicazioni sulla presenza di note musicali (o accordi) particolarmente rilevanti durante l'assolo.

Le topologie di reti ottenute rispecchiano le caratteristiche dello strumento dal quale sono state generate. In particolare, concentrandosi sul numero di nodi, è evidente quanto gli strumenti a tastiera (pianoforte e organo) creino reti composte da un numero molto elevato di nodi, con un conseguente diametro del grafo nettamente più elevato rispetto alla media degli altri strumenti. Questo perché si tratta di strumenti armonici in grado di eseguire più note contemporaneamente.

Le reti risultanti dai fiati in ottone (saxofono e tromba) presentano meno nodi ed un diametro minore, in quanto sono strumenti melodici e quindi in grado di eseguire solamente una nota per volta. In questo caso i nodi molto distanti sono una chiara rappresentazione di suoni lontani fra di loro nella scala dell'estensione relativa al singolo strumento.

Strumenti come la chitarra e, in questo caso, il liuto, pur essendo strumenti armonici possiedono un'estensione più limitata rispetto alle tastiere, per cui presentano un numero di nodi intermedio rispetto a quello di tastiere e fiati. Il numero dei nodi ed i valori assunti dal diametro di ogni grafo sono mostrati nella tabella seguente.

Strumento	Numero di nodi	Diametro
Saxofono contralto	8	5
Saxofono baritono	10	5
Saxofono tenore	8	7
Liuto	21	5
Organo	48	13
Pianoforte	120	11
Tromba	9	4
Violino	20	7
Viola	16	3
Vibrafono	19	3

Per quanto riguarda la **Degree distribution** è interessante evidenziare come nel caso del pianoforte il valore medio (4.78) ed il valore massimo (76) si discostino di molto: ciò implica la presenza di pochi nodi con grado molto elevato, i quali rappresentano quindi note che durante l'assolo vengono ripetute molte volte.

Una Degree distribution più omogenea è osservabile invece negli strumenti a fiato, dove valore medio e valore massimo si discostano di un massimo di 5 punti.

Pianoforte ed organo, essendo strumenti armonici, accompagnano la melodia con accordi che esprimono i giri armonici attraverso note cardine della tonalità. Tali note cardine si rispecchiano esattamente in nodi dei grafi di grande importanza, evidenziati dai valori assunti dalla **Betweenness centrality** che, per i suddetti strumenti, sono decisamente maggiori rispetto agli altri. A seguire vi sono poi i valori assunti da vibrafono, violino, liuto e viola.

I valori massimi e medi assunti dalla Betweenness centrality per i diversi assoli sono mostrati nella tabella seguente.

Strumento	Massimo	Medio
Saxofono contralto	33	10.5
Saxofono baritono	38	12
Saxofono tenore	27	18.4
Liuto	199	28.5
Organo	1184.4	199.8
Pianoforte	8757.4	295.4
Tromba	22	7.9
Violino	249.5	31.1
Viola	173.5	14.9
Vibrafono	295.5	16.5

Per quanto riguarda il **coefficiente di clustering**, esso misura il grado in cui i nodi di un grafo tendono ad essere connessi fra loro.

É possibile notare che gli assoli di tromba e saxofono contralto presentano un coefficiente di clustering maggiore rispetto agli altri. Ciò sta ad indicare che i nodi dei relativi grafi tendono a creare gruppi più uniti e caratterizzati da una densità di collegamenti relativamente più alta.

I valori risultanti dal calcolo della **Eigenvector centrality** sui diversi grafi sono interessanti nel caso del vibrafono: lo strumento in questione, a differenza degli altri, presenta suoni più vicini fra loro che privilegiano formule ritmiche ripetitive. Ciò sta a significare la presenza di "percorsi ritmici" più importanti di altri, il che si identifica esattamente con un valore massimo di Eigenvector centrality decisamente maggiore rispetto a tutti gli altri assoli.

Quindi, i nodi con valore di Eigenvector centrality più elevato rispecchiano le note ricorrenti nei pattern ritmici tipici di questo strumento.

Anche in questo caso è stato verificato, per ogni tipo di assolo, se la rete corrispondente fosse di tipo **Small-World**.

Analizzando le condizioni necessarie descritte nel paragrafo precedente, è possibile affermare che l'assolo di organo possiede le caratteristiche per essere classificato come rete di questo tipo, in quanto presenta una

differenza tra i due coefficienti nettamente maggiore rispetto agli altri assoli.

I valori assunti dai parametri analizzati sono mostrati nella tabella seguente.

Parametro	Valore
Coefficiente di clustering	0.45
CC grafo random	0.04
Lunghezza media dei cammini	3.32
Lunghezza media grafo random	3.87

Si può quindi concludere che la rete derivante dall'assolo di organo è di tipo Small-World.

Di seguito vengono riportate le rappresentazioni grafiche di tutte le reti analizzate in questo capitolo. I grafi dei diversi assoli utilizzati sono stati generati utilizzando un colore differente per ciascuno.

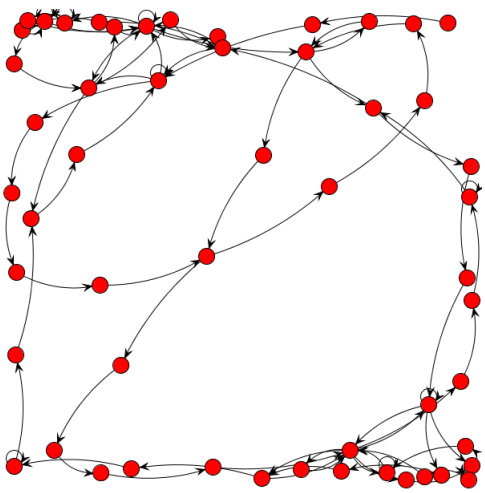


Figura 4.5: Rete derivante dal brano autonomamente realizzato

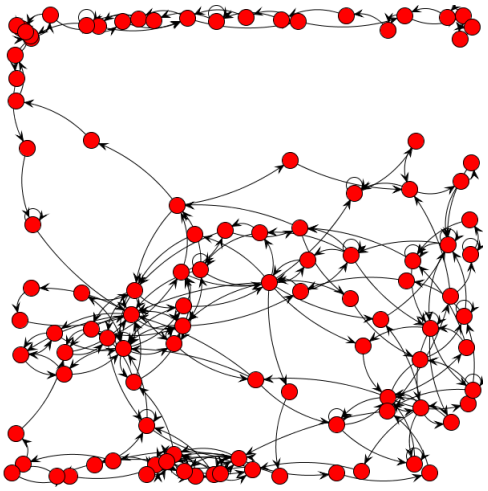


Figura 4.6: Rete corrispondente al brano "*Per Elisa*"

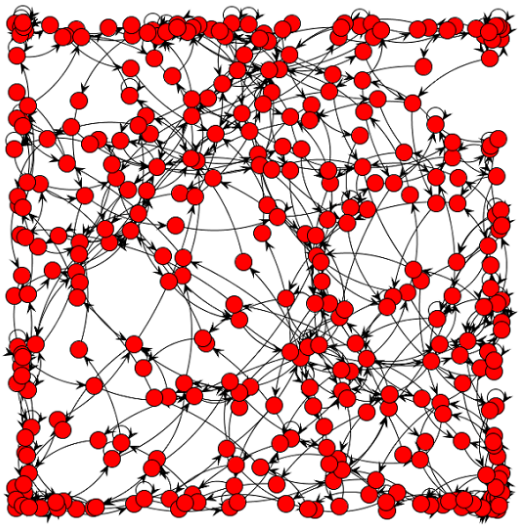


Figura 4.7: Rete corrispondente al brano "*Sultans Of Swing*"

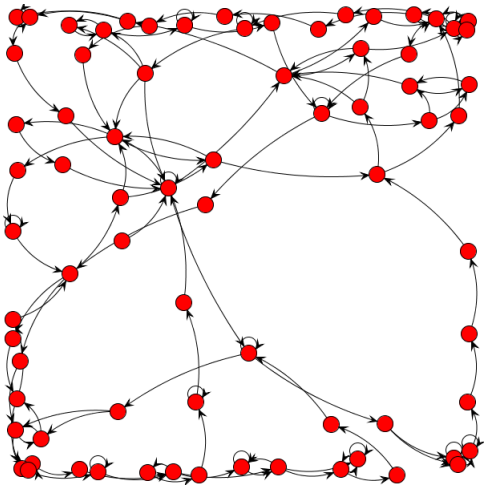


Figura 4.8: Rete corrispondente al brano "*Yesterday*"

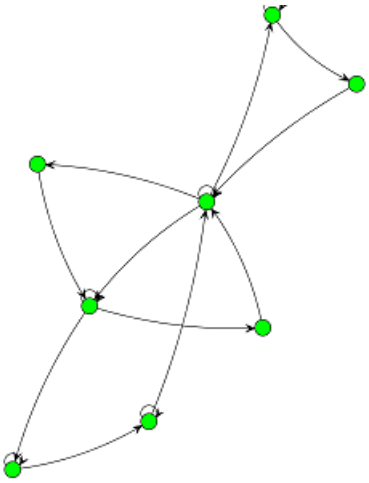


Figura 4.9: Rete corrispondente all'assolo di saxofono contralto

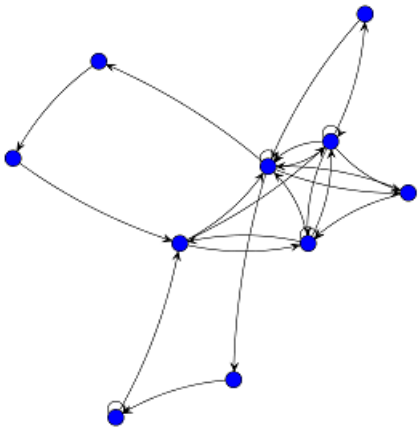


Figura 4.10: Rete corrispondente all'assolo di saxofono baritono

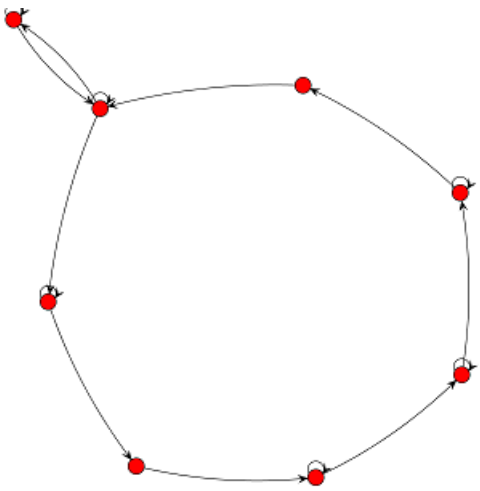


Figura 4.11: Rete corrispondente all'assolo di saxofono tenore

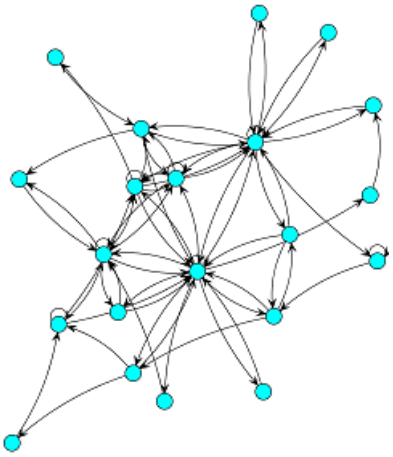


Figura 4.12: Rete corrispondente all'assolo di liuto

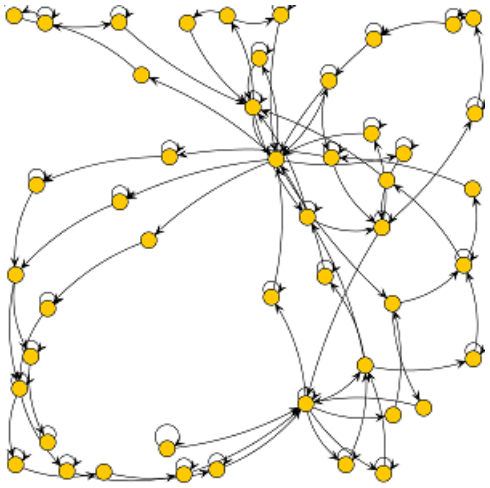


Figura 4.13: Rete corrispondente all'assolo di organo

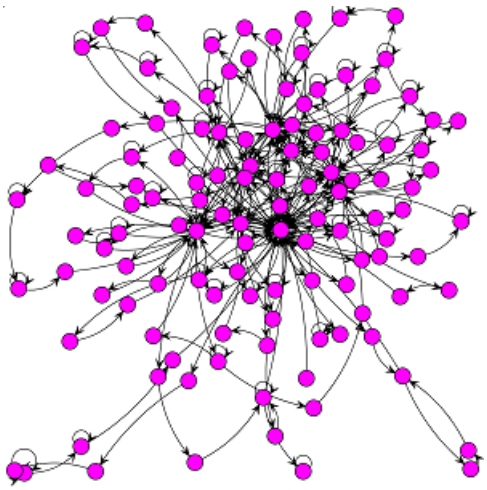


Figura 4.14: Rete corrispondente all'assolo di pianoforte

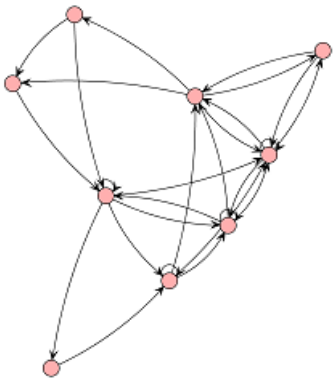


Figura 4.15: Rete corrispondente all'assolo di tromba

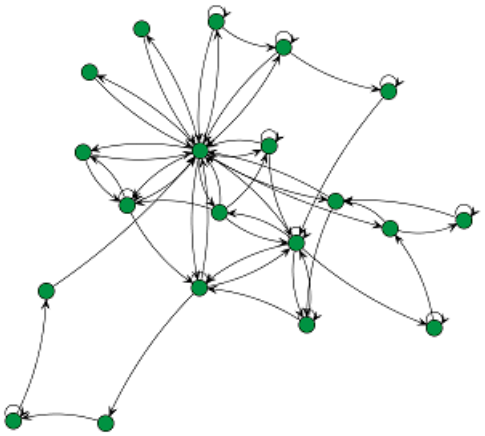


Figura 4.16: Rete corrispondente all'assolo di violino

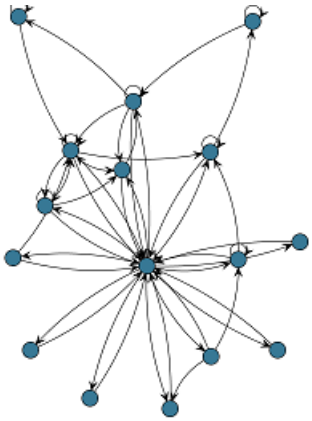


Figura 4.17: Rete corrispondente all'assolo di viola

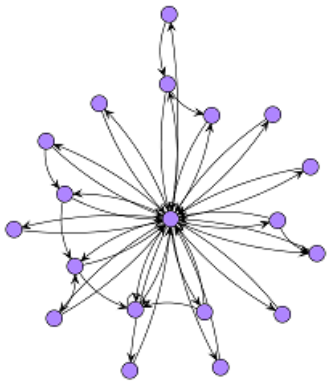


Figura 4.18: Rete corrispondente all'assolo di vibrafono

Capitolo 5

Conclusioni

Nel presente documento è stato proposto uno strumento software in grado di effettuare una conversione da file in formato MIDI a file in formato MusicXML. Lo strumento realizzato ha il compito di affiancare un tool che consente di modellare melodie (e brani musicali in genere) come reti complesse. Essendo di fondamentale importanza in questo contesto le partiture in formato MusicXML, la realizzazione di un software di conversione ha consentito di generare un database di file MusicXML più ampio di quello che sarebbe stato possibile reperire sul Web.

In seguito, sono state poi effettuate alcune analisi sulle reti ottenute modellando differenti brani musicali. Le metriche principali proprie delle reti complesse sono state utilizzate per confrontare le diverse strutture e verificarne analogie e/o differenze.

Il tool in questione è stato realizzato con l'obiettivo di prescindere dall'intermediazione di una partitura grafica, producendo file MusicXML direttamente da un'analisi di file in formato MIDI.

A tal proposito, bisogna specificare che in alcune circostanze si è visto necessario effettuare opportune semplificazioni a causa dell'assenza degli elementi grafici in questione. Ciò ha portato alla creazione di reti con una perdita di nodi con basso "*degree*".

Tuttavia, in generale questo fattore non altera in modo significativo la struttura della rete che viene prodotta. Tra gli sviluppi futuri è possibile

inserire un perfezionamento del tool che consenta di non dover ricorrere alle semplificazioni di cui si è citato.

Lo studio sulla struttura delle reti che possono essere ottenute da una modellazione di melodie ha evidenziato la presenza di proprietà che sono in relazione diretta con le caratteristiche proprie di ogni brano musicale. L'utilizzo di un modello matematico fornisce quindi un modo generale e compatto per analizzare la musica.

Il risultato di questo studio mostra chiaramente quanto una modellazione di questo tipo possa avere un impatto non irrilevante sulla realizzazione di molteplici applicazioni multimediali, spaziando da propositi di classificazione e identificazione fino alla generazione automatica di musica.

Applicazioni di questo tipo possono risultare di interesse in ambiti differenti, che partendo dalla didattica musicale è possibile espandere fino all'intrattenimento multimediale e alla generazione di musica digitale.

Bibliografia

[1] Ferretti, Stefano (2016), "*Guitar Solos As Networks*", in *Proc. of the IEEE International Conference on Multimedia and Expo (ICME 2016)*, IEEE, Seattle (USA)

[2] Riganti, Farina (1996), "*Enciclopedia della musica*", Garzanti Editore, Milano

[3] Redazioni Grandi Opere e Varia (1978), "*Dizionario enciclopedico della musica e dei musicisti*", Fratelli Fabbri Editori, Milano

[4] "*Dizionario enciclopedico universale della musica e dei musicisti*", Unione Tipografico - Editrice Torinese, Torino (1983)

[5] Berenzweig, Logan, Ellis, Whitman (2004), "*A large-scale evaluation of acoustic and subjective music similarity measures*", *Computer Music Journal*, vol. 28, no. 2

[6] Guérin, Robert (2003), "*MIDI - L'interfaccia digitale per gli strumenti musicali*", Apogeo Editore, Milano

[7] The MIDI Manufacturers Association (2014), "*The Complete MIDI 1.0 Detailed Specification*", third edition, Los Angeles

[8] Miles Huber, David (2007), "*The MIDI Manual, A practical guide to MIDI in the project studio*", third edition, Burlington (Massachusetts)

[9] "*MusicXML 3.0 Documentation*" (2015), MakeMusic, Boulder (Colorado)

[10] Maarten van Steen (2010), "*An introduction to graph theory and complex networks*"

[11] Caldarelli, Guido (2007), "*Scale-Free Networks: complex webs in nature and technology*", Oxford University Press, New York

[12] Watts, Duncan J. (2004), "*Six Degrees: the science of a connected age*", W. W. Norton & Company, New York

[13] Lambertini, Mattia (2010), "*Analisi di grafi su architetture a memoria distribuita*", Bologna

[14] Costa, Rodrigues, Travieso, Villas Boas (2008), "*Characterization of Complex Networks: A Survey of measurements*", São Carlos (Brasile)

[15] Newman, Mark E.J. (2005), "*Random graphs as models of networks*", in *Handbook of Graphs and Networks*, Wiley-VCH Verlag

[16] Bondy, Murty (1976), "*Graph theory with applications*", Macmillan, Londra

[17] Barrat, Barthélémy, Pastor-Satorras, Vespignani (2004), "*The architecture of complex weighted networks*", *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 11

[18] Albert, Barabasi (2001), "*Statistical mechanics of complex networks*", in *Reviews of Modern Physics* 74, 47

[19] Dorogovtsev, Mendes (2001), "*Evolution of networks*", in *Advances in Physics* 51, 1079