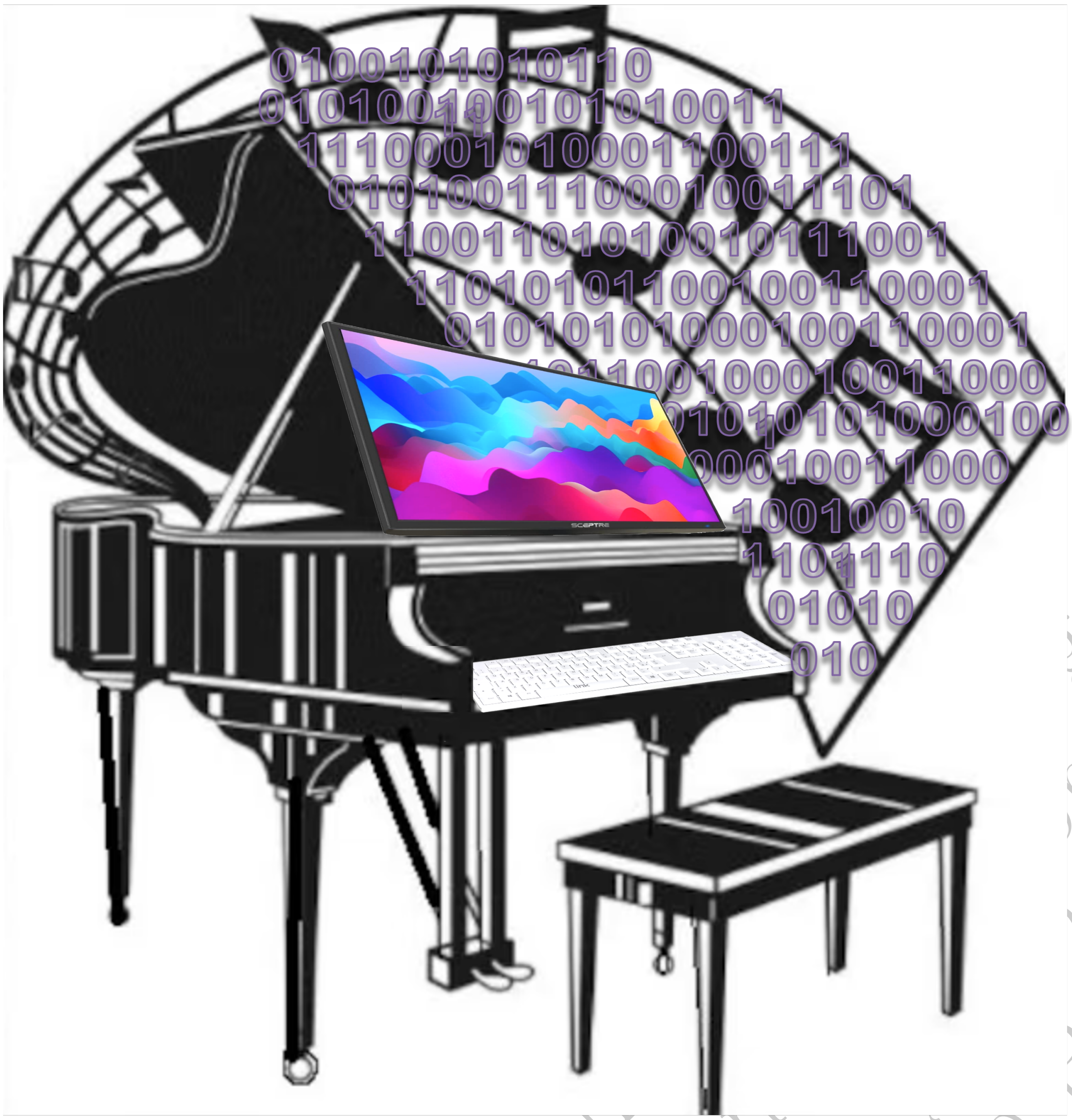
Composizione Musicale Algoritmica



Appunti per i corsi di Musimatica e Algorithmic Music and Sound Computing
v2023.01

robdep

Indice

Elenco delle figure

Capitolo 1

Nozioni basilari

Questo capitolo fornisce delle nozioni musicali di base. L'obiettivo è quello di fornire le conoscenze minime, a particolare beneficio del lettore non esperto di musica, per poter proseguire con gli argomenti più avanzati. L'esposizione è solo introduttiva e limitata: il lettore interessato può approfondire gli argomenti trattati su libri di teoria e armonia musicale.

1.1 Suoni

Il suono è ciò che percepisce il nostro orecchio come effetto della vibrazione di un corpo in oscillazione. Un corpo in vibrazione produce delle onde sonore che si propagano nell'aria (o in un altro mezzo elastico) raggiungendo l'orecchio. Le onde sonore provocano un cambiamento periodico di pressione sul timpano dell'orecchio; tali cambiamenti di pressione vengono elaborati dal nostro complesso sistema uditivo e interpretati come suoni. La percezione del suono elaborata dal sistema uditivo, è direttamente correlata alla natura delle onde sonore prodotte dalla vibrazione: in funzione delle caratteristiche delle onde sonore percepiamo suoni diversi.

La propagazione delle onde impiega tempo per percorrere la distanza che separa la sorgente sonora dall'orecchio di chi ascolta. La velocità con cui le onde si propagano dipende sia dal mezzo trasmissivo sia da altre condizioni, come ad esempio la temperatura del mezzo trasmissivo. Se il mezzo trasmissivo è l'aria, la velocità del suono è di circa 343,1 m/s (cioè circa 1235 km/h) ad una temperatura di 20°. A temperature inferiori la velocità è più bassa, ad esempio a 10° è di circa 337,5 m/s (1215 km/h).

Le principali caratteristiche di un'onda sonora sono la frequenza, che determina l'*altezza* del suono, e l'intensità che determina il *volume* del suono percepito. Un'altra caratteristica importante delle onde sonore è la forma delle onde; la forma delle onde determina il *timbro* del suono: la forma delle onde sonore prodotte da un violino è diversa dalla forma delle onde sonore prodotte da un sassofono.

La frequenza è misurata in Hertz (Hz, cicli per secondo). Il sistema uditivo umano può percepire suoni con frequenze comprese (approssimativamente) fra 17Hz e 17.000Hz. Un pianoforte produce suoni in un intervallo che va approssimativamente da 27 a 4.000 Hz. Suoni al di fuori di questo intervallo sono difficili da distinguire.

L'intensità del suono viene misurata in decibel (dB). In un ambiente estremamente silenzioso (ad es. una biblioteca) l'intensità del suono è di circa 40dB, mentre il limite massimo per il nostro orecchio è di circa 120dB (es. suono prodotto da una navicella spaziale al decollo). L'intensità media del suono prodotto da un pianoforte è di circa 90dB.

1.2 Note

Alcuni suoni sono meno piacevoli di altri. Un suono sgradevole viene chiamato rumore. I suoni più gradevoli all'orecchio umano, per esempio quelli prodotti dagli strumenti musicali, vengono utilizzati per fare musica. Una nota è un suono gradevole con un proprio timbro, una propria altezza e un proprio volume. Inoltre una nota è "posizionata nel tempo", cioè inizia in un determinato istante di tempo e ha una durata che ne determina ovviamente la fine. Le note sono gli elementi di base per la composizione (creazione/costruzione) della musica.

1.2.1 Altezza e frequenza

L'altezza di una nota è quella caratteristica che ci permette di distinguere un suono grave da un suono acuto e come vedremo in seguito è una caratteristica fondamentale per la consonanza di più suoni. L'altezza è determinata dalla frequenza dell'onda sonora che produce la nota. Sebbene ogni frequenza corrisponda a un suono, e quindi a una nota, l'odierno sistema musicale prevede l'utilizzo di un determinato sottoinsieme delle infinite frequenze. La particolare frequenza di 440Hz è associata alla nota `La` posizionata nella parte centrale del pianoforte, mostrata nella Figura 1.1, e ognuno degli altri 87 tasti corrisponde a una specifica altra frequenza; quindi le note prodotte da un pianoforte corrispondono a 88 specifiche frequenze. Le 88 note sono divise in cosiddette *ottave*, ognuna della quali raggruppa 12 note.

La sensibilità del nostro orecchio nel percepire note diverse è la base per la creazione delle melodie e delle armonie. Due suoni con la stessa frequenza vengono percepiti come la stessa nota (in effetti sono la stessa nota). Due suoni con frequenze diverse vengono percepite come distinti. Se le frequenze sono molto vicine fra di loro, ad esempio, 440Hz e 441Hz, è estremamente difficile distinguere i due suoni. All'aumentare della differenza delle frequenze la diversità dell'altezza appare evidente. La capacità di individuare la differenza è soggettiva. Un ottimo orecchio musicale riesce a distinguere anche suoni con frequenze molto vicine; un orecchio "stonato" ha maggiori

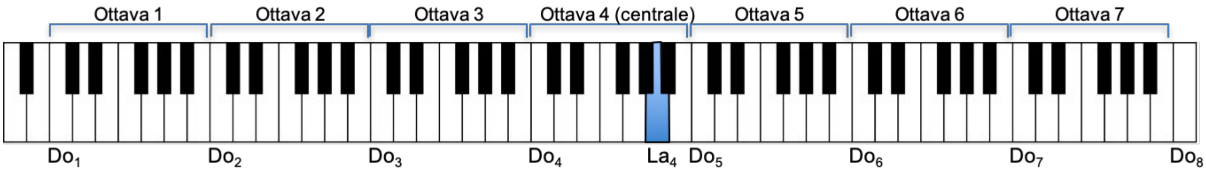


Figura 1.1: Tastiera a 88 tasti

difficoltà. Una capacità abbastanza rara è quella chiamata “orecchio assoluto” e consiste nel riuscire a individuare le frequenze delle note/suoni che si ascoltano. Pochissime persone hanno il dono dell’orecchio assoluto. Ancor più raro è l’orecchio “armonico” che riesce a percepire le singole note di un insieme di note suonate contemporaneamente. Più comune è l’orecchio relativo che riesce a individuare l’intervallo fra due note, cioè la differenza fra le frequenze delle due note.

Due (o più) suoni percepiti insieme o in sequenza possono risultare più o meno gradevoli. La “consonanza” di due suoni fa riferimento proprio a questo aspetto. Due note con la stessa frequenza sono consonanti. Anche due note le cui frequenze sono una il doppio dell’altra sono molto consonanti; sono talmente consonanti che il nostro orecchio è portato a considerare i due suoni come lo stesso suono.

Usare un programma che permetta di controllare la frequenza del suono prodotto per ascoltare suoni con frequenze diverse ma vicine. Ad esempio provare a distinguere un suono con frequenza 440Hz da un suono con frequenza 441Hz. Aumentare la differenza se i suoni sembrano uguali. Ascoltare due suoni con frequenze una il doppio dell’altra.

Un intervallo, cioè la distanza fra due suoni, può essere rappresentato con il rapporto delle frequenze dei due suoni. Un intervallo costituito da due note con la stessa frequenza f corrisponde al rapporto $f/f = 1$.

Un intervallo costituito da due note con frequenze f e $2f$ corrisponde al rapporto $\frac{2f}{f} = 2$. I due intervalli appena descritti sono dei casi particolari e corrispondono all’unisono (rapporto=1) e all’ottava (rapporto=2). Si noti come il termine *ottava*, che indica anche i gruppi di 12 note sul pianoforte, venga utilizzato per indicare anche un suono con una frequenza doppia rispetta a un altro suono. La motivazione per la quale si usa il numero 8, da cui deriva ottava, è dovuta al fatto che, come vedremo in seguito, fra un suono e la sua ottava si costruiscono delle sequenza di altri suoni, dette scale, che solitamente, ma non sempre[1], sono formate da 7 suoni per cui l’ottavo suono corrisponde a quello di partenza.

Data una frequenza di riferimento f_r, possiamo ottenere tutte le ottave corrispondenti a f_r raddoppiando o dimezzando f_r. Quindi le frequenze delle ottave di f_r sono

$$f_n = f_r \cdot 2^n, n \in Z$$

Le frequenze $f_n > f_r$ corrispondono alle ottave più acute (alte) di f_r, mentre quelle con $f_n < f_r$, ottenute scegliendo n negativo corrispondono alle ottave più gravi (basse). L’unisono può essere considerato come l’ottava ottenuta scegliendo $n = 0$.

Le ottave (e l’unisono) di una nota sono le note che risultano più consonanti al nostro orecchio. Che l’unisono sia una consonanza non sorprende, visto che le due note sono in realtà la stessa

[1] E da questo punto di vista l’uso del numero 8 è improprio, visto che ci possono essere scale con un numero diverso di note.

nota. Le ottave vengono percepite come la stessa nota di partenza, solo più acuta o più grave. Volendo usare un linguaggio matematico potremmo dire che l'unisono corrisponde all'uguaglianza e l'ottava all'equivalenza.

Per questo motivo il nostro sistema musicale è basato sulle ottave. Volendo usare di nuovo un parallelo con la matematica, possiamo dire che la consonanza delle ottave è l'assioma del nostro sistema musicale.

1.2.2 Timbro

Il timbro è l'insieme delle caratteristiche di un suono che permette di distinguire diverse sensazioni uditive per suoni con timbri diversi. Ad esempio il violino produce un suono con un timbro diverso da quello prodotto da un pianoforte o da un flauto. Differenze di timbro possono essere ottenute anche usando lo stesso strumento ma in maniere diverse, ad esempio pizzicando le corde piuttosto che suonandole con l'arco, nel caso di uno strumento ad arco.

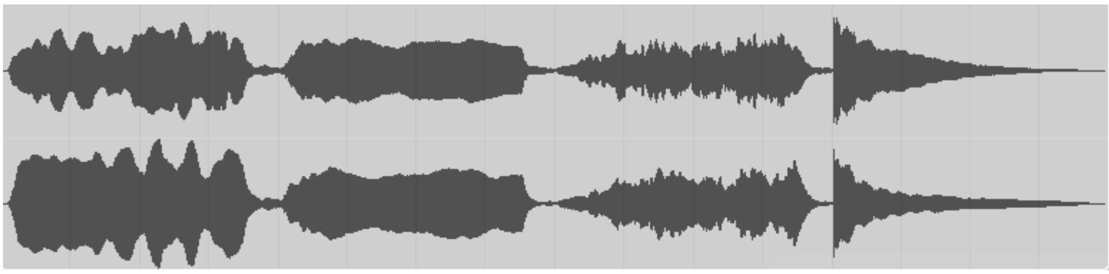


Figura 1.2: Forma d'onda di suoni di flauto, violino, organo e pianoforte

La Figura 1.2 mostra le diverse forma d'onda per i timbri di un flauto, un organo, un violino, e un pianoforte[2] quando suonano la stessa nota. In questa figura la scala del tempo è molto piccola. La Figura 1.3 mostra un piccolo pezzo della forma d'onda del suono del pianoforte con una scala molto grande che permette di vedere meglio la forma d'onda.

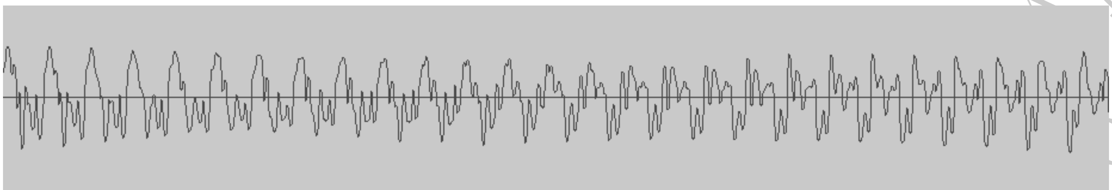


Figura 1.3: Forma d'onda (pianoforte) ingrandimento

1.2.3 Durata

Le note hanno una durata e sono posizionate nel tempo, cioè per ogni nota c'è l'inizio della nota e successivamente, dopo un intervallo di tempo pari alla durata della nota, il suona cessa. Da un punto di vista matematico le note possono essere viste come funzioni che associano a un determinato intervallo temporale una determinata altezza. Pertanto è possibile rappresentare la musica usando un sistema di assi cartesiani in cui le ordinate indicano il tempo e le ascisse la frequenza. Ogni nota sarà rappresentata da un segmento orizzontale il cui inizio a la cui fine, e di conseguenza la cui durata, dipendono dalla posizione della nota nella partitura e dalla durata della nota. La Figura 1.4 riporta un esempio.

[2] Il file audio `timbri.mp3` contiene i suoni la cui forma d'onda è mostrata nella figura.

Tabella 1.1: Frequenze delle note

Ottava 0			
#	Midi	Nota	Frequenza
1	21	A	27.500 Hz
2	22	A#	29.135 Hz
3	23	B	30.867 Hz

Ottava 1			
#	Midi	Nota	Frequenza
4	24	C	32.703 Hz
5	25	C#	34.647 Hz
6	26	D	36.708 Hz
7	27	D#	38.890 Hz
8	28	E	41.203 Hz
9	29	F	43.653 Hz
10	30	F#	46.249 Hz
11	31	G	48.999 Hz
12	32	G#	51.913 Hz
13	33	A	55.000 Hz
14	34	A#	58.270 Hz
15	35	B	61.735 Hz

Ottava 2			
#	Midi	Nota	Frequenza
16	36	C	65.406 Hz
17	37	C#	69.295 Hz
18	38	D	73.416 Hz
19	39	D#	77.781 Hz
20	40	E	82.407 Hz
21	41	F	87.307 Hz
22	42	F#	92.498 Hz
23	43	G	97.999 Hz
24	44	G#	103.826 Hz
25	45	A	110.000 Hz
26	46	A#	116.541 Hz
27	47	B	123.471 Hz

Ottava 3			
#	Midi	Nota	Frequenza
28	48	C	130.813 Hz
29	49	C#	138.591 Hz
30	50	D	146.832 Hz
31	51	D#	155.563 Hz
32	52	E	164.814 Hz
33	53	F	174.614 Hz
34	53	F#	184.997 Hz
35	55	G	195.998 Hz
36	56	G#	207.652 Hz
37	57	A	220.000 Hz
38	58	A#	233.081 Hz
39	59	B	246.942 Hz

Ottava 4			
#	Midi	Nota	Frequenza
40	60	C	261.625 Hz
41	61	C#	277.182 Hz
42	62	D	293.664 Hz
43	63	D#	311.126 Hz
44	64	E	329.627 Hz
45	65	F	349.228 Hz
46	66	F#	369.994 Hz
47	67	G	391.995 Hz
48	68	G#	415.304 Hz
49	69	A	440.000 Hz
50	70	A#	466.163 Hz
51	71	B	493.883 Hz

Ottava 5			
#	Midi	Nota	Frequenza
52	72	C	523.251 Hz
53	73	C#	554.365 Hz
54	74	D	587.329 Hz
55	75	D#	622.253 Hz
56	76	E	659.255 Hz
57	77	F	698.456 Hz
58	78	F#	739.988 Hz
59	79	G	783.990 Hz
60	80	G#	830.609 Hz
61	81	A	880.000 Hz
62	82	A#	932.326 Hz
63	83	B	987.766 Hz

Ottava 6			
#	Midi	Nota	Frequenza
64	84	C	1046.502 Hz
65	85	C#	1108.721 Hz
66	86	D	1174.658 Hz
67	87	D#	1244.506 Hz
68	88	E	1318.310 Hz
69	89	F	1396.912 Hz
70	90	F#	1479.996 Hz
71	91	G	1567.980 Hz
72	92	G#	1661.218 Hz
73	93	A	1760.000 Hz
74	94	A#	1864.652 Hz
75	95	B	1975.532 Hz

Ottava 7			
#	Midi	Nota	Frequenza
76	96	C	2093.004 Hz
77	97	C#	2217.442 Hz
78	98	D	2349.316 Hz
79	99	D#	2489.012 Hz
80	100	E	2636.620 Hz
81	101	F	2793.824 Hz
82	102	F#	2959.992 Hz
83	103	G	3135.969 Hz
84	104	G#	3322.436 Hz
85	105	A	3520.000 Hz
86	106	A#	3729.304 Hz
87	107	B	3951.064 Hz

Ottava 8			
#	Midi	Nota	Frequenza
88	108	C	4186.008 Hz

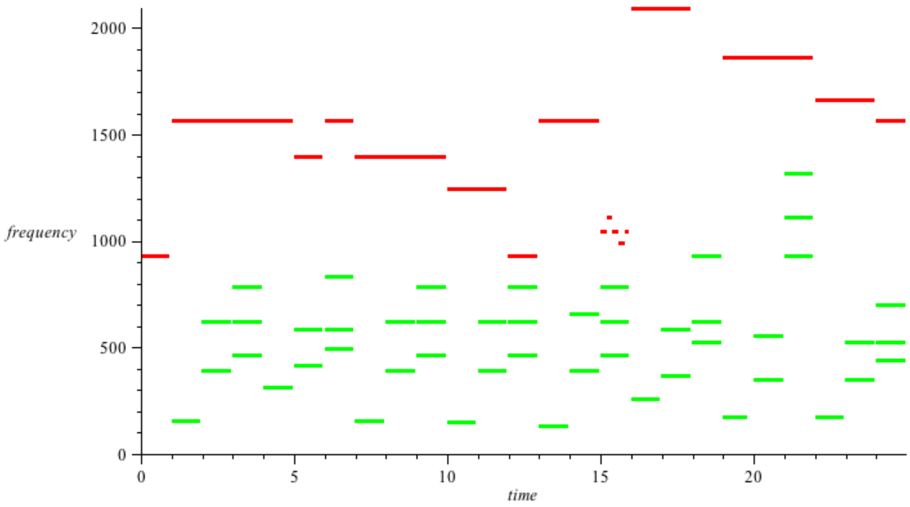


Figura 1.4: Esempio di rappresentazione su assi cartesiani (Chopin, Notturno op.9 n.2)

1.3 Note in un'ottava

Se a ogni frequenza corrispondesse una nota avremmo infinite note, una per ogni possibile frequenza. Poichè però è difficile distinguire due note con frequenze che differiscono di poco, in pratica vengono utilizzate solo determinate frequenze. In realtà questa scelta è abbastanza arbitraria, anche se ci sono delle motivazioni ben precise. Le ottave sono state divise in 12 note. Cioè fra le frequenze incluse tra un valore f e il suo doppio $2f$, vengono scelte "solo" 12 frequenze, 13 considerando anche l'ottava, che determinano le 12 possibili note in un'ottava. Inoltre il sistema utilizza come punto di partenza la frequenza pari a 440 Hz (anche questa scelta è arbitraria e non di rado si sceglie 442Hz o anche 443Hz, mentre in altri periodi storici sono stati usati anche valori leggermente più piccoli di 440).

Nel Capitolo 2 approfondiremo la scelta delle specifiche frequenze per le note. Anticipiamo qui qualche informazione. Data una frequenza di riferimento f_R, la formula $f_x = f_R \cdot 2^x$, $x \in R$ ci permette di calcolare una qualsiasi altra frequenza. In particolare per $x \in Z$ otteniamo le ottave di f_R. Abbiamo già detto che l'ottava è considerata la base del sistema musicale in quanto l'orecchio percepisce le ottave come estremamente consonanti, quasi come se fossero lo stesso suono.

L'ottava è divisa in 12 intervalli, detti *semitoni*, uguali. Le frequenze di questi 12 suoni possono essere calcolate, partendo da una frequenza di rifierimento, con

$$f_k = f_R \cdot \left(2^{\frac{1}{12}}\right)^k, k \in 0, 1, 2, ..., 12.$$

Ad esempio se la frequenza di riferimento è $f_R = 440Hz$, che corrisponde al `La` centrale del nostro sistema musicale, si ha che $f_1 = 440 \cdot 2^{1/12} \simeq 446Hz$. In altre parole l'intervallo di un semitono corrisponde al valore

$$\sqrt[12]{2} \simeq 1.05946.$$

Pertanto fra la nota `La` corrispondente alla frequenza di 440Hz e la sua ottava a 880Hz ci sono altre 11 note, come mostrato nella seguente figura:

Le note di un'ottava vengono identificate da nomi. Nella notazione di origine latina si utilizzano le sillabe `Do`, `Re`, `Mi`, `Fa`, `Sol`, `La` e `Si`, mentre in quella anglosassone i nomi delle note sono le lettere dalla `A` alla `G`, con la `A` che indica il `La`, la `B` il `Si`, la `C` il `Do` e così via, come mostrato nella tabella che segue.

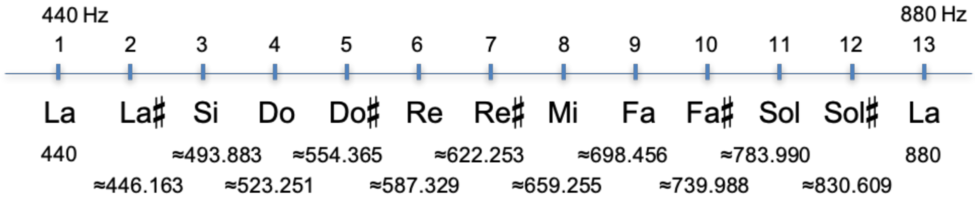


Figura 1.5: Note in un ottava

A	B	C	D	E	F	G
La	Si	Do	Re	Mi	Fa	Sol

Oltre alle 7 che hanno un nome proprio che non fa uso delle *alterazioni*, ci sono altre 5 note che si ottengono usando le alterazioni di bemolle (♭), che abbassa la frequenza alla nota precedente, e diesis (♯), che alza la frequenza alla nota successiva. Dette note si trovano fra il Do e il Re (Do♯ o Re♭), il Re e il Mi (Re♯ o Mi♭), il Fa e il Sol (Fa♯ o Sol♭), il Sol e il La (Sol♯ o La♭), e infine fra il La e il Si (La♯ o Si♭). Queste note possono essere identificate in più modi, come ad esempio Do♯ e Re♭. L'utilizzo delle alterazioni può anche essere multiplo anche se nella pratica musicale ci si limita alla doppia alterazione: un doppio bemolle, indicato con "♭♭" e un doppio diesis indicato con "♯♯". Un doppio bemolle altera la nota abbassandola di due semitoni, pertanto si ha che Sol♭♭ equivale a Fa, come Do♯♯ equovale a Re. Analogamente per il doppio diesis.

La Figura 1.6 illustra i nomi delle note in un ottava che parte dalla nota Do.

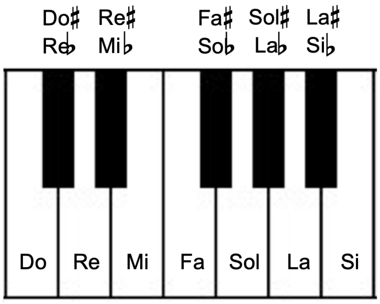


Figura 1.6: Nomi delle note

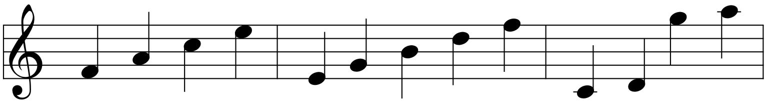
La Tabella 1.1 riporta le frequenze delle note che corrispondono agli 88 tasti di un pianoforte e il loro codice MIDI; del protocollo MIDI parleremo nel Capitolo 3.

1.4 Tono e semitono

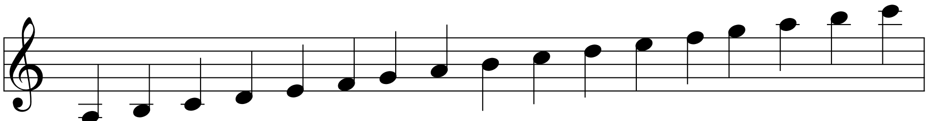
Numerando le note nell'ordine in cui compaiono su un pianoforte, o utiliozzando la numerazione del protocollo MIDI, la distanza fra una nota i e quella successiva $i+1$, ad es. fra un La e il successivo La♯ oppure fra un Mi e il successivo Fa, è detta *semitono*, che nel seguito abbrevieremo con "s", mentre la distanza fra una nota i e la nota $i+2$, ad es. fra La e il successivo Si opppure fra un Mi e il succssivo Fa♯, è detta *tono*, che nel seguito abbrevieremo con "T". Le motivazioni per questa nomenclatura saranno chiarite nel Capitolo 2.

1.5 Rappresentazione: il pentagramma

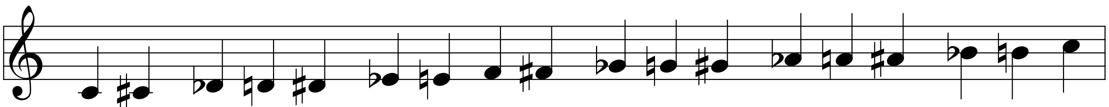
Per rappresentare graficamente le note i musicisti utilizzano il *pentagramma*, che è formato da 5 linee parallele. All'inizio del pentagramma è riportato un simbolo detto *chiave*. La chiave determina l'altezza (ottava di riferimento) delle note riportate nel pentagramma. Nella figura seguente sono riportate le posizioni delle note nel pentagramma con la chiave di violino.



Con questa chiave la seconda linea del pentagramma corrisponde al Sol della quarta ottava (47° tasto del pianoforte, codice MIDI 67). Pertanto, le note negli spazi fra i 5 righi del pentagramma sono `Fa`, `La`, `Do` e `Mi`, mentre le note sui 5 righi sono `Mi`, `Sol`, `Si`, `Re` e `Fa`. È possibile andare anche oltre il pentagramma usando righi addizionali (non espressamente disegnati) nel qual caso si utilizzano dei segni detti *tagli* che indicano il numero di righi addizionali necessari. Nell'esempio precedente la prima e la quarta nota dell'ultimo gruppo fanno uso di un taglio. Nell'esempio seguente le note sono riportate in ordine di altezza crescente.



La seguente figura mostra tutte le 12 note di un'ottava (13 considerando che il Do c'è sia all'inizio sia alla fine come ottava del primo) in un pentragramma in chiave di violino.

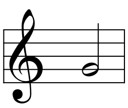


1.5.1 Rappresentazione della durata

Nell'analogia con le frazioni, l'*intero* musicale è la *semibreve* (*whole*), che si indica con un cerchietto vuoto:

Semibreve (*whole*)

Essendo un intero la semibreve equivale a 4 quarti. La metà di una semibreve è la *minima* (*half*) che ovviamente vale 2 quarti. È rappresentata con un cerchietto vuoto con un gambo (che può essere sia verso l'alto, quando lo si mette a destra del cerchietto, che verso il basso, quando lo si mette a sinistra).



Minima (*half*)

La metà di una minima è detta *semiminima* (*crotchet* o *quarter*) e vale 1 quarto e a differenza della minima ha il cerchietto riempito:

Seminima (*crotchet* o *quarter*)

La metà di una semiminima è detta *croma* (*quaver*) e vale 1 ottavo e a differenza della seminimima ha una *cediglia* (o coda) attaccata al gambo:

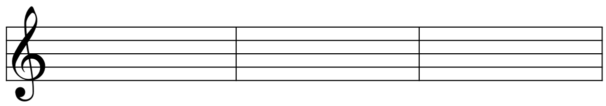
Croma (*quaver*)

La metà di una croma è detta *semicroma* (*semiquaver*) e vale 1 sedicesimo e rispetto alla croma ha due cediglie:

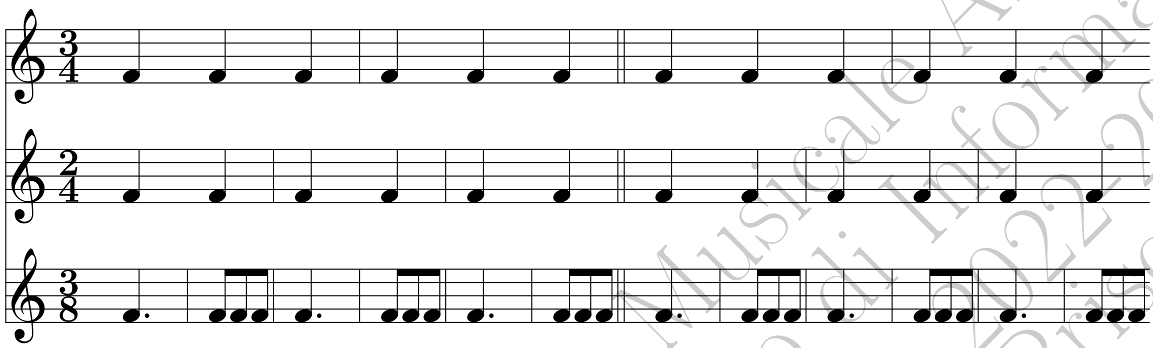
Semicroma (*quaver*)

Esistono anche la biscroma (*demisemiquaver*) che vale un 32esimo e la semibiscroma (*hemidemisemiquaver*) che vale un 64esimo, che hanno rispettivamente 3 e 4 cediglie. Un punto dopo il simbolo della nota aumenta la durata del 50%. Ad esempio, una nota che dura 1/2, seguita da un punto, durerà 1/2+1/4; una nota che dura 1/4, seguita da un punto, deurerà 1/4+1/8; una nota che dura 1/4, seguita da due punti, durerà 1/4+1/8+1/16.

Il pentagramma può essere visto come una sorta di sistema di assi cartesiani in cui l'asse verticale viene usato per rappresentare l'altezza, l'asse orizzontale viene utilizzato per indicare il tempo. In effetti ha la stessa funzionalità della rappresentazione mostrata in Figura 1.4 perchè riporta le stesse informazioni, cioè altezza e durata delle singole note, ma con simboli grafici più facilmente leggibili. Quindi per quanto riguarda il tempo, il pentagramma va letto da sinistra verso destra. Le altezze invece vengono specificate dalla posizione verticale nel pentagramma. La durata utilizza i simboli descritti in precedenza. Inoltre la musica viene divisa in *battute* la cui fine è segnalata da un barra verticale nel pentagramma.



La durata di ogni singola battuta e delle note specificate nel pentagramma dipende da un'indicazione metrica specificata subito dopo la chiave.



La metrica del tempo, nella forma di una frazione senza il simbolo di frazione (cioè due interi uno sopra l'altro), si suole leggere nello stesso modo di una frazione, ad esempio *tre quarti*. Essa indica due cose. Il "numeratore" indica di quanti elementi è fatta una battuta. Il "denominatore" indica la durata di ogni singolo elemento e, di conseguenza, insieme al numeratore, la durata dell'intera battuta. Riprendendo l'esempio del tempo tre quarti si ha che ogni battuta è composta da 3 elementi la cui durata (relativa) è di un quarto. La durata è relativa in quanto dobbiamo specificare con una indicazione separata la velocità con cui si esegue la musica: tale velocità viene specificata in *battiti* al minuto. Ad esempio ♪ =132 , che determina la durata reale di una croma specificando che in un minuto ci sono 132 crome (quindi una croma in questo caso dura

60/132 $\simeq 0,45$ secondi).

1.5.2 Chiavi

Il simbolo della chiave posto all'inizio del pentagramma di fatto indica una particolare "traslazione" delle note, verso il basso o verso l'alto. Detto in altre parole, a ogni chiave viene associata una determinata gamma di note. Esistono varie chiavi per poter indicare all'interno del pentagramma le varie ottave. Per i nostri scopi è sufficiente conoscere le cosiddette chiavi di violino e di basso.

1.6 Scale

Una scala musicale è una sequenza di suoni con altezza crescente che partono da una determinata nota e arrivano all'ottava successiva. La possiamo mettere in relazione a una scala che ci permette di salire da un piano al successivo; per la musica i "piani" sono una nota e le sue ottave. Proprio come succede per le scale fisiche, la scala musicale ha dei gradini che ci permettono di salire, e anche scendere ovviamente, per gradi. Il numero di gradini, che nel gergo musicale sono chiamati *gradi*, è variabile. A differenza delle scale fisiche (almeno quelle ben costruite!), i "gradini" delle scale musicali non sono tutti uguali. Ogni gradino ha la sua "alzata": l'intervallo fra un grado e quello successivo può variare. Le scale più comuni hanno 7 gradini per cui l'ottavo suono corrisponde alla consonanza con frequenza doppia rispetto a quella di partenza. Per questo motivo la frequenza doppia viene chiamata ottava. Ovviamente il termine è improprio per scale che sono fatte da un numero diversi di gradini, ma essendo d'uso comune e universalmente accetato, il termine ottava indica la frequenza doppia anche quando non è l'ottavo suono della scala.

Dunque, genericamente parlando, una scala è una sequenza di un numero arbitrario di suoni, che determina una sequenza di intervalli (le alzate dei gradini). Pertanto la sequenza di intervalli identifica univocamente la scala. Da un punto di vista informatico/matematico è comodo misurare gli intervalli in semitoni, visto che il semitono è la più piccola distanza fra due note e che tutte le altre distanze sono multipli di semitoni. Inoltre, considerando che la scala ci deve far salire da una nota alla sua ottava, che dista esattamente 12 semitoni dalla nota di partenza, una scala può essere definita nel seguente modo.

Scala musicale. Una scala musicale è identificata da una sequenza di n interi positivi, $\{s_1, s_2, \ldots, s_n\}$, tali che

$$\sum_{i=1}^{n} s_i = 12.$$

Ad esempio la scala che otteniamo salendo da un `Do` al `Do` successivo usando tutti i tasti bianchi del pianoforte inclusi fra il `Do` di partenza e il `Do` di arrivo, è `Do`, `Re`, `Mi`, `Fa`, `Sol`, `La`, `Si` e il `Do` all'ottava, e corrisponde alla sequenza di intervalli $\langle 2, 2, 1, 2, 2, 2, 1\rangle$. Questa scala è la scala denominata *maggiore*. Gli intervalli di semitono e tono corrispondono rispettivamente a 1 e 2. Quindi la sequenza di intervalli della scala maggiore è TTsTTTs.

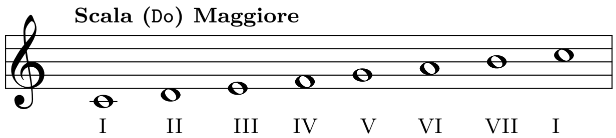
Nota	Intervalli	Modo (nome scala)
`Do`	T T s T T T s	Ionico (Maggiore)
`Re`	T s T T T s T	Dorico
`Mi`	s T T T s T T	Frigio
`Fa`	T T T s T T s	Lidio
`Sol`	T T s T T s T	Misolidio
`La`	T s T T s T T	Eolio (Minore)
`Si`	s T T s T T T	Locrio

Tabella 1.2: Scale diatoniche

Dovrebbe essere chiaro che, con questa definizione, il numero possibili di scale è enorme. Ad esempio $\langle 1, 2, 3, 4, 2\rangle$ è una scala, come pure $\langle 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\rangle$. Ma anche la sequenza $\langle 1, 1, 1, 1, 1, 1, 6\rangle$ o qualsiasi altra sequenza di interi positivi la cui somma è 12. Non tutte le sequenze che hanno questa proprietà però corrispondono a scale effettivamente utilizzate in musica; in pratica le scale utilizzate sono poche. Quelle più utilizzate, in particolare nella musica cosiddetta classica, sono la scala maggiore e la scala minore. Nella musica contemporanea, in particolare nel jazz, vengono usate anche altre scale.

1.6.1 Scala maggiore e scale diatoniche

Una scala diatonica è costruita usando le sette note senza alterazioni in sequenza di altezza ascendente; la scala inizia da una nota di riferimento finisce con l'ottava superiore della nota di partenza. Partendo dal `Do` si ottiene una scala detta *maggiore*:



Nella scala maggiore e la successione di intervalli è la seguente:

```
Do  Re  Mi  Fa  Sol  La  Si  Do
  T   T   s   T    T   T   s
```

In questa particolare scala diatonica l'intervallo di semitono esiste fra il terzo ed il quarto grado della scala (`Mi` e `Fa`) e fra il settimo e l'ottavo grado (il `Si` ed il `Do` all'ottava). Fra tutti gli altri gradi successivi intercorre un intervallo di tono.

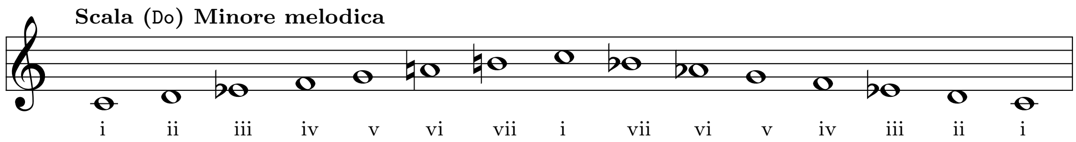
Poichè una scala diatonica può partire da una qualsiasi delle 7 note, esistono sette diverse scale diatoniche, che corrispondono a degli "shift circolari" della sequenza di intervalli della scala maggiore, come riassunto nella Tabella 1.2.

1.6.2 Scala minore melodica

Partendo dal La si ottiene la scala minore, la cui sequenza diversa di intervalli, è TsTTsTT.

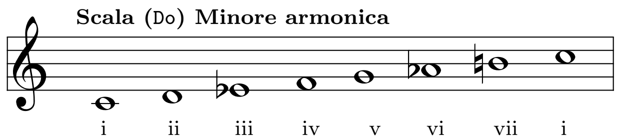
```
La  Si  Do  Re  Mi  Fa  Sol  La
  T   s   T   T   s   T    T
```

Inoltre la scala minore, quando eseguita in senso ascendente, prevede la sesta e la settima nota *alterate*, cioè innalzate di un semitono. Viene detta scala minore *melodica* per distinguerla dalla scala minore detta *armonica*.



1.6.3 Scala minore armonica

La scala minore armonica è data dalla sequenza di intervalli TsTTs3s, dove il 3 indica un intervallo d 3 semitoni; quindi non può essere costruita con le sole note senza alterazioni. A differenza della minore melodica, è uguale sia in senso ascendente che in senso discendente.



1.6.4 Scala cromatica

La scala cromatica è una scala che è formata da tutti i suoni dell'ottava in sequenza di frequenza crescente. Corrisponde alla sequenza di intervalli 1,1,1,1,1,1,1,1,1,1,1,1. In una scala cromatica due note sucessive sono sempre a distanza di un semitono.

Figura 1.7: Scala cromatica

1.6.5 Altre scale

La scala pentatonica maggiore è formata da 5 suoni con i seguenti intervalli misurati in numero di semitoni: {2,3,2,2,3} (`Do`,`Re`,`Fa`,`Sol`,`La`,`Do`). I 5 tasti neri di un pianoforte, a partire dal Do♯ formano una scala pentatonica maggiore. La pentatonica minore invece è {3,2,2,3,2} (`La`,`Do`,`Re`,`Fa`,`Sol`,`La`). I 5 tasti neri di un pianoforte, a partire dal La♯ formano una scala pentatonica minore. Come per le scale maggiori e minori diatoniche la scala pentatonica minore è uno shift ciclico di quella maggiore. La scala esatonale, usata molto da Debussy, è {2,2,2,2,2,2} (`Do`,`Re`,`Mi`,`Fa`♯,`Sol`♯,`La`♯,`Do`) La scala blues è {3,2,1,1,3,2} (`Do`,`Mi`♭,`Fa`,`Sol`♭,`La`,`Si`,`Do`).

1.6.6 Scale maggiormente utilizzate

Le scale utilizzate nella musica contemporanea occidentale sono il modo maggiore ed il modo minore; gli altri modi vengono usati molto raremente o praticamente mai, con l'eccezione della musica Jazz e Blues.

1.6.7 Gradi della scala

Confinando il discorso alle scale classiche di 7 suoni, le 7 note di una scala musicale vengono chiamate *gradi* della scala, con il primo grado che corrisponde alla prima nota e il settimo grado alla settima nota. I gradi vengono indicati con i numeri romani, in maiuscolo per la scala maggiore e in minuscolo per quella minore.

I gradi della scala hanno anche un nome che descrive una specifica funzione musicale. Il primo grado è detto *tonica*, in quanto rappresenta il "tono" (nota) sul quale è costruito la scala: la "tonalità" è la scala stessa. Il quinto grado è detto *dominante*. La tonica è la dominante sono i gradi più importanti; nella musica popolare spesso i brani sono costruiti usando solo accordi di tonica e di dominante. La seguente tabella riporta i nomi di tutti i gradi delal scala.

Grado	Nome
I	Tonica
II	Sopratonica
III	Mediante
IV	Sottodominante
V	Dominante
VI	Sopradominante
VII	Sensibile

1.7 Scale e trasposizione

La suddivisione dell'ottava in 12 semitoni è stata ideata per poter utilizzare una determinata scala, ad esempio la scala maggiore, a partire da una qualunque delle 12 possibili note. Ad esempio per ottenere la scala maggiore a partire da `Re`, dobbiamo utilizzare le seguenti note `Re`, `Mi`, `Fa`♯, `Sol`, `La`, `Si`, `Do`♯, `Re`. Si può facilmente verificare che gli intervalli fra i gradi di questa scala corrispondono a quelli della scala maggiore:

```
Re  Mi  Fa#  Sol  La  Si  Do#  Re
  T   T    s    T   T   T    s
```

Pertanto qualsiasi nota può fungere da punto di partenza per una scala maggiore. Poichè ci sono 12 note, possiamo avere 12 scale maggiori, fra di loro equivalenti in termini di distanza fra i gradi successivi. La Tavola 1.3 riporta tutte le scale maggiori che si possono costruire a partire da una qualsiasi delle 12 note. Le note alterate (i tasti neri del pianoforte) possono essere specificate sia da un ♯ che da un ♭.

Le alterazioni utilizzate in ognuna delle scale servono ad avere sempre la stessa successione di intervalli, cioè TTsTTTs, della scala maggiore. Le tonalità maggiori possono essere organizzate nel cosiddetto circolo delle quinte in funzione del numero di alterazioni utilizzate dalla scala maggiore. Nella sequenza del circolo delle quinte si parte dalla scala di Do che non ha alterazioni e ogni scala successiva ha un'alterazione in più rispetto alla precedente. Potendo usare 2 alterazioni (il ♯ ed il ♭) possiamo spostarci in due direzioni, verso destra per i ♯ e verso sinistra per ♭. Ad un certo punto proseguendo in entrambe le direzioni si ottengono scale equivalenti (stesse note anche se con nomi diversi). Per questo motivo si ha un "cerchio", da cui il nome circolo.

Le alterazioni possono essere anche specificate *in chiave*, in modo tale da non doverle ripetere per ogni nota. Stabilita la tonica automaticamente abbiamo stabilito l'intera scala che, in funzione della tonica, determinirà le alterazioni, ♯ o ♭, necessarie per mantenere gli intervalli richiesti dalla scala. Per non ripetere l'alterazione a ogni nota le alterazioni determinate dalla scala vengono scritte una sola volta all'inizio del rigo subito dopo la chiave.

La Figura 1.8, fornisce una descrizione grafica del circolo delle quinte..

1.8 Intervalli musicali

Un intervallo musicale fra due note di una scala è dato dalla distanza fra i relativi gradi, distanza che viene misurata contando i gradi inclusi quello di partenza e quello di arrivo. Ad

Scala di	Note								Numero di ♯	Numero di ♭
Do	Do	Re	Mi	Fa	Sol	La	Si	Do	-	-
Do♯	Do♯	Re♯	Mi♯	Fa♯	Sol♯	La♯	Si♯	Do♯	7	-
Re♭	Re♭	Mi♭	Fa	Sol♭	La♭	Si♭	Do	Re♭	-	5
Re	Re	Mi	Fa♯	Sol	La	Si	Do♯	Re	2	-
Mi♭	Mi♭	Fa	Sol	La♭	Si♭	Do	Re	Mi	-	3
Mi	Mi	Fa♯	Sol♯	La	Si	Do♯	Re♯	Mi	4	-
Fa	Fa	Sol	La	Si♭	Do	Re♯	Mi	Fa	-	1
Fa♯	Fa♯	Sol♯	La♯	Si	Do♯	Re♯	Mi♯	Fa♯	6	-
Sol♭	Sol♭	La♭	Si♭	Do♭	Re♭	Mi♭	Fa	Sol♭	-	6
Sol	Sol	La	Si	Do	Re	Mi	Fa♯	Sol	1	-
La♭	La♭	Si♭	Do	Re♭	Mi♭	Fa	Sol	La♭	-	4
La	La	Si	Do♯	Re	Mi	Fa♯	Sol♯	La	3	-
Si♭	Si♭	Do	Re	Mi♭	Fa	Sol	La	Si♭	-	2
Si	Si	Do♯	Re♯	Mi	Fa♯	Sol♯	La♯	Si	5	-
Do♭	Do♭	Re♭	Mi♭	Fa♭	Sol♭	La♭	Si♭	Do♭	-	7

Tabella 1.3: Scale Maggiori

esempio, fra un Do e il successivo Mi in senso ascendente c'è un intervallo di terza, fra un Re e il successivo La in senso ascendente c'è un intervallo di quinta, fra un La il "successivo" Re in senso discendente c'è un intervallo di quarta. Si noti come sia stata sfruttata la ciclicità delle scale e anche l'importanza della direzione ascendente o discendente: da un La al successivo Re, se il senso è ascendente c'è un intervallo di quarta, mentre se il senso è discendente c'è un intervallo di quinta.

Un'altra osservazione importante è relativa alla reale distanza fra le note. Tale distanza dipende dalla scala. Ad esempio in una scala maggiore la distanza fra il I e il III grado è di 4 semitoni (Do-Do♯-Re-Re♯-Mi) o equivalentemente di 2 toni. Mentre in una scala minore è di 3 semitoni (La-La♯-Si-Do). Quindi si distingue fra terza maggiore e terza minore. In generale un intervallo può essere aumentato o diminuito di un semitono. Ad esempio l'intervallo di quinta è fatto da 7 semitoni (Do-Do♯-Re-Re♯-Mi-Fa-Fa♯-Sol); aumentandolo di un semitono si ha un quinta detta *eccedente* (es. Do-Sol♯) mentre dimunuendolo di un semitono si ha una quinta detta *diminuita* (es. Do-Sol♭). Si noti anche come per l'intervallo di quinta non c'è differenza fra scala maggiore e scala minore: ci sono sempre 7 semitoni. Nel gergo musicale si usa l'aggettivo *giusta* per la quinta; quindi una quinta è giusta sia nella scala maggiore che nella scala minore. Per entrambe può essere sia eccedente che diminuita.

Oltre alla quinta anche l'unisono, la 4^a e l'8^a vengono detti giusti, in quanto non differisco nelle scale maggiori e minori. Di contro, la 2^a, la 3^a la 6^a e la 7^a sono maggiori nella scala maggiore e minori nella scala minore. Un intervallo minore può essere ulteriormente diminuito, e un intervallo maggiore può essere anche aumentato.

Un modo sistematico per classificare gli intervalli è quello di contare i semitoni fra le due note. La seguente tabella riassume gli intervalli in funzione del numero di semitoni.

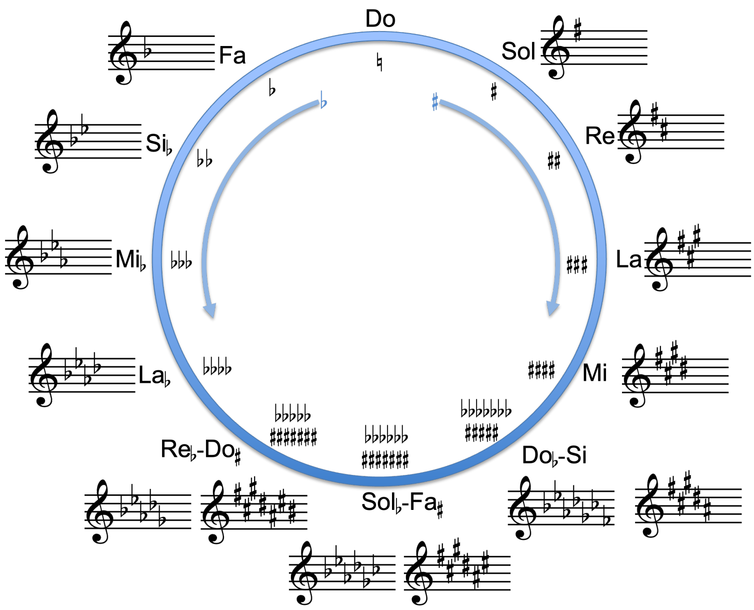


Figura 1.8: Circolo delle quinte, scale maggiori

intervallo	semitoni	esempio
seconda diminuita	0	Do♯-Re♭
seconda minore	1	Do-Re♭
seconda maggiore	2	Do-Re
seconda aumentata	3	Do-Re♯
terza diminuita	2	Do♯-Mi♭
terza minore	3	Do-Mi♭
terza maggiore	4	Do-Mi
terza aumentata	5	Do-Mi♯
quarta diminuita	4	Do♯-Fa
quarta giusta	5	Do-Fa
quarta eccedente	6	Do-Fa♯

intervallo	semitoni	esempio
quinta diminuita	6	Do♯-Sol
quinta giusta	7	Do-Sol
quinta aumentata	8	Do-Sol♯
sesta diminuita	7	Do♯-La♭
sesta minore	8	Do-La♭
sesta maggiore	9	Do-La
sesta aumentata	10	Do-La♯
settima diminuita	9	Do♯-Si♭
settima minore	10	Do-Si♭
settima maggiore	11	Do-Si
settima aumentata	12	Do-Si♯

1.9 Melodia e armonia

La melodia e l'armonia sono i due "assi cartesiani" della musica. Una melodia è una successione temporale di note che nel loro insieme formano un canto, una *melodia* appunto. Mentre l'armonia è un insieme di note simultanee che in sinergia formano un *accordo*. Un brano musicale è costituito da più linee melodiche che proseguono insieme e, in sinergia, formano accordi che cambiano nel tempo. La Figura 1.9 fornisce una rappresentazione grafica: più lenee melodiche procedono insieme e analizzandole verticalmente in uno specifico istante di tempo otteniamo un insieme di note che formano un accordo. In realtà il discorso è un po' più complesso in quanto spesso ci sono note che non fanno parte dell'accordo, ma questa semplificazione rende l'idea di base.

Nella figura sono state rappresentate quattro linee melodiche. Ovviamente non è sempre così. Il numero di linee melodiche è variabile; dipende dal tipo di composizione. Inoltre una delle linee melodiche, in molti casi ma non sempre quella più acuta, è la linee melodica principale che è quella che dovrebbe essere messa in risalto. Le altre linee melodiche tipicamente servono per creare l'accordo, anche se a volte hanno una vera e propia funzione melodica. Normalmente il

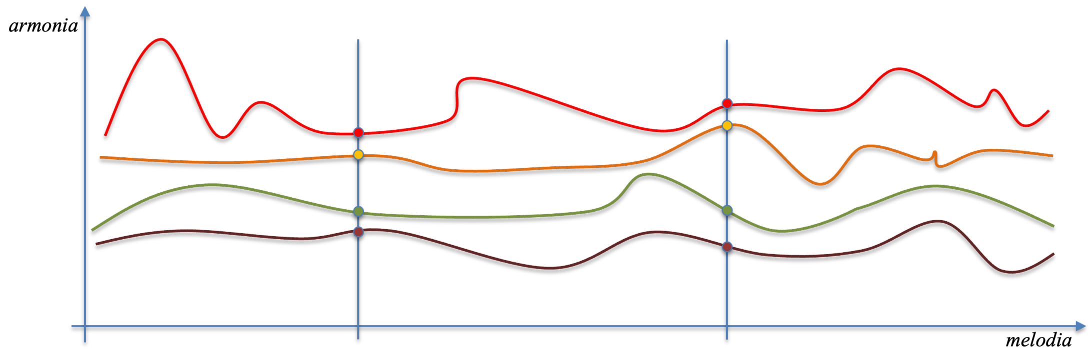


Figura 1.9: Assi "musicali": melodia e armonia.

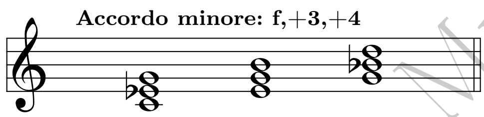
"tema" di una composizione viene proposto nella linea melodica principale ma può essere anche proposto nelle altre linee.

1.10 Accordi

Un accordo è un insieme simultaneo di almeno 3 note. Gli accordi più semplici sono le *triadi*, costituite da esattamente 3 note. Una triade è formata da un nota di partenza, detta *fondamentale* dell'accordo, dalla nota a distanza di una terza e dalla nota a distanza di una quinta dalla fondamentale, e quindi di una terza dalla seconda noda. Esistono 4 tipi di accordi di 3 suoni che si defferenziano negli intervalli di terza fra le 3 note:

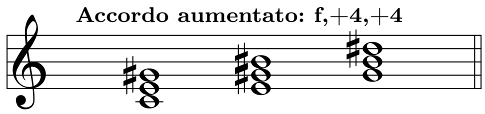
- Accordo (perfetto) maggiore: formato da terza maggiore e quinta giusta; se guardiamo gli intervalli fra le note successive abbiamo una terza maggiore e una terza minore. Distanze in semitoni: 4 e 3.

- Accordo (perfetto) minore: formato da terza minore e quinta giusta; se guardiamo gli intervalli fra le note successive abbiamo una terza minore e una terza maggiore. Distanze in semitoni: 3 e 4.



- Accordo di 5^a diminuita: formato da terza minore e quinta diminuita; gli intervalli fra le note successive sono due terze minori. Distanze in semitoni: 3 e 3.

- Accordo di 5^a aumentata: formato da terza maggiore e quinta aumentata; gli intervalli fra le note successive sono due terze maggiori. Distanze in semitoni: 4 e 4.



La definizione della triade prevede che la seconda e la terza nota sia a distanza di una terza. Tuttavia la triade può esser suonata anche invertendo l'ordine, ottenendo così tre possibili posizioni che in gergo vengono dette *rivolti*.

Quando la fondamentale corrisponde alla nota più grave dell'accordo, si dice che la triade è nella *posizione fondamentale*; se la nota più grave è la terza, allora si dice che la triade è in *primo rivolto*; se la nota più grave è la quinta, allora si dice che la triade è in *secondo rivolto*. La Figura 1.10 mostra un esempio.

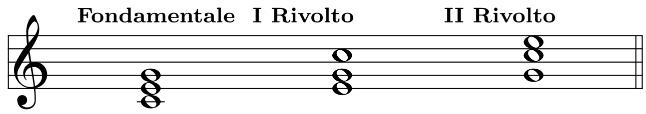


Figura 1.10: Rivolto delle triadi

Gli accordi possono usare anche più di 3 suoni. In particolare gli accordi di 4 suoni sono detti accordi di settima, in quanto aggiungono la settima a un accordo di triade, e gli accordi di 5 suoni sono detti di nona, in quanto aggiungono una nona ad un accordo di settima.

1.11 Forme musicali

Le composizioni musicali hanno, nella maggior parte dei casi, una struttura ben definita. In gergo si utilizza il sostantivo *forma* musicale per indicare la struttura della composizione. Ovviamente esistono varie forme musicali. La forma determina il modo in cui gli elementi musicali, come ad esempio il tema, vengono organizzati. Ad esempio, in un brano di musica pop, solitamente sono previste due sezioni che vengono chiamate *strofa* e *ritornello* e che spesso proposte nella sequenza SSRSSR, dove S indica la strofa e R il ritornello. Come altro esempio possiamo citare la forma *sonata* nella musica classica. Tale forma prevede 3 sezioni: l'esposizione del tema, lo sviluppo e la ripresa. Ognuna di queste 3 parti ha una propria struttura interna. L'esposizione presenta un tema principale, un ponte modulante (cioè che porta a un altra tonalità), un secondo tema e una coda che porta a un'altra tonalità in contrasto con quella di impianto (solitamente verso la dominante per le tonalità maggiori e la relativa maggiore per le tonalità minori). Lo sviluppo presenta una eleborazione del materiale tematico e finisce con un ritorno nella tonalità di impianto. La ripresa ha una struttura simile all'esposizione ma con il ponte modulante che evita lo spostamento nella tonalità di contrasto e quindi nella ripresa il tema principale e quello secondario vengono proposti entrambi nella tonalità di impianto e la coda porta alla conclusione della composizione.

1.11.1 Musica pop

La musica pop è tipicamente rappresentata tramite una sola linea melodica, quella della melodia principale, accompagnata da una indicazione dell'armonia (accordi da utilizzare). Tale rappresentazione è una semplificazione di quello che poi viene effettivamente suonato: l'arrangiamento infatti è un aspetto cruciale che determina (buona) parte del risultato finale. Tuttavia, per i nostri scopi questa semplificazione in melodia più armonia, ci aiuta nella gestione di questo tipo di brani.

La Figura 1.11 riporta un frammento della partitura di Memory, un famoso brano di Andrew Loyd Webber tratto dal musical Cats.

Figura 1.11: La melodia e l'armonia del brano Memory.

La Figura 1.12, invece, mostra lo spartito di *O sole mio*. Il brano è composto da un'introduzione (I), una strofa (S) e un ritornello (R) con strofa e ritornello che si ripetono per 2 volte, quindi la forma di questo brano può essere schematizzata con ISRSR. Anche nella loro lunghezza questi 3 elementi seguono una struttura ben definita: l'introduzione è di 8 battute, mentre sia la strofa che il ritornello sono lunghi 16 battute. Organizzare la musica in gruppi di 8 battute è molto tipico della musica pop, e non solo. Si noti come anche il tema musicale di Memory sia di 8 battute.

Figura 1.12: Melodia, armonia e struttura di *O sole mio*.

Oltre alla lunghezza delle frasi musicali e all'organizzazione in strofa e ritornello, e a volte introduzioni o anche anche altre parti, come code oppure ponti (*bridge*), ci sono anche altre caratteristiche che sono comuni. Una di queste è la struttura armonica, cioè la sequenza di accordi utilizzati per accompagnare la melodia. Un esempio semplice è il cosiddetto *giro di Do*. La sequenza di accordi utilizzata è `Do`, `La`*m*, `Fa` (oppure `Re`*m*), `Sol`o, più genericamente, gli accordi di I, vi, IV (oppure ii) e V grado. I brani di musica pop che sfruttano questa struttura armonica sono innumerevoli. Alcuni esempi: Sapore di sale (Gino Paoli), Il cielo in una stanza (Gino Paoli), Ti amo (Umberto Tozzi), Il gatto e la volpe (Eduardo Bennato), Viva la mamma (Eduardo Bennato), Stand by me (Ben E. King), Every breath you take (Police).

La sequenza di accordi usata nel *giro di Do* deriva da osservazioni più generali riguardo a quali sequenze di accordi funzionano meglio di altre. Approfondiremo questo discorso nella Sezione 1.12.

1.11.2 Corale

Una particolare forma musicale della musica classica è quella definita *corale*. In un corale ci sono tipicamente 4 linee melodiche che si sviluppano in sinergia. Il nome deriva dal fatto che la musica corale è pensata per essere eseguita da un coro in cui ci sono 4 voci, dette *soprano*, *contralto*, *tenore* e *basso*, che corrispondono grossolanamente alle estensioni vocali delle persone. Le voci possono salire a 6 se si considerano anche le estensioni vocali intermedie fra soprano e contralto, il *mezzosoprano* e fra il tenore e il basso, il *baritono*. Con gli strumenti si posso avere anche più di 6 voci. Il numero di voci non è prettamente legato alle estensioni, ma alle diverse linee melodiche. Per un corale abbiamo esattamente 4 linee melodiche.

La figura 1.13 mostra le prime battute del corale BWV66.6 di J.S.Bach.

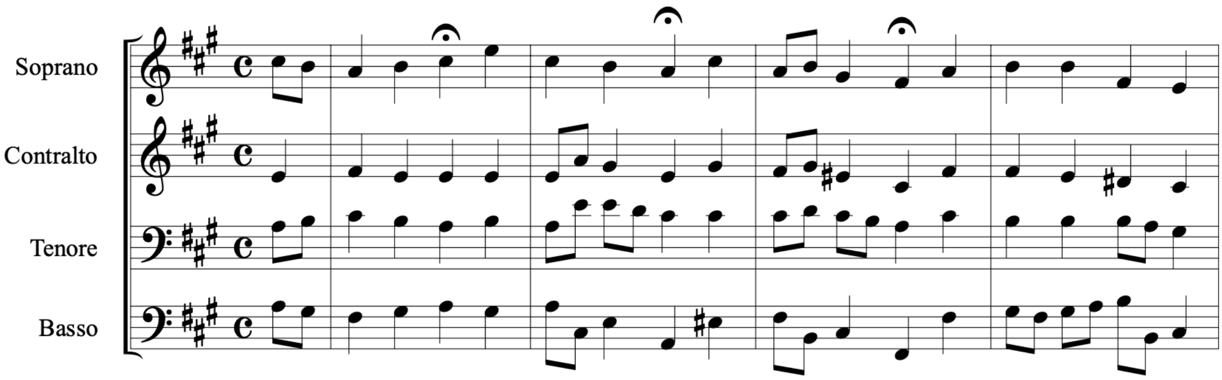


Figura 1.13: Le prime battute del corale BWV 66.6 di J.S. Bach.

1.11.3 Regole sul movimento delle voci

In una composizione in cui più voci si muovono simultaneamente, come ad esempio in un corale, ci sono delle regole che restringono il movimento delle voci. Non sono regole ferree ma derivano dalla buona pratica musicale sono tipiche della musica classica.

Moto delle parti

Se consideriamo 2 linee melodiche simultane possiamo distinguire 3 modalità per il loro movimento reciproco. Due linee melodiche procedono per moto retto se entrambe le voci salgono oppure scendono. Procedono per moto contrario se una sale e l'altra scende. Infine procedono per moto obliquo se una si muove e l'altra rimane ferma.

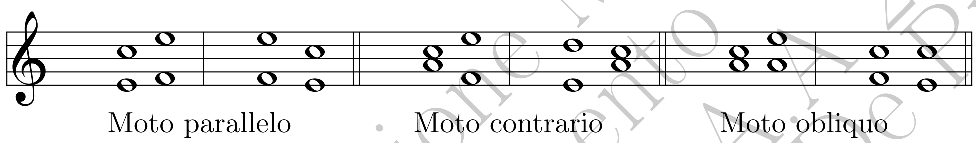


Figura 1.14: Moto delle parti

Errori nel movimento delle parti

Due voci che procedono simultanemanete devono procedere evitando delle disposizioni che producono effetti sgradevoli. Gli errori riguardano le consonanze perfette, cioè l'unisono, la quinta e l'ottava. Possono essere più (errori reali) o meno (errori nascosti) gravi. Gli errori reali si hanno quando le due voci procedono parallelamente con lo stesso intervallo.

Meno grave ma sempre da evitare sono gli intervalli delle consonaze perfette a meno che non vengano raggiunti con una delle due voci che procede per gradi congiunti.

Per approfondimenti si veda un libro di armonia (es. Mazzotta)

Salti melodici vietati

All'interno di una singola linea melodica è buona prassi evitare tutti i salti melodici diminuiti ed eccedenti. Sono consentiti i salti diminuiti se sono discendenti e cadono sulla sensibile che va alla tonica. Inoltre vanno evitati anche il salto di 7^a sia minore che maggiore e anche due salti successivi che sommati formano una 7^a. Il salto di 8^a è possibile purchè la linea melodica cambi direzione dopo il salto.

1.12 Successioni armoniche tipiche

Non esistono delle regole che possano determinare le successioni armoniche delle composizioni. Tuttavia è possibile inidividuare delle successioni tipiche. Le seguenti successioni tipiche sono riportate in Piston [pag. 21, pag 45].
Nel modo maggiore:

- Il I grado è seguito dal IV o dal V, a volte dal VI; meno sovente dal II o dal III
- Il II grado è seguito dal V, a volte dal IV o dal VI; meno sovente dal I o dal III

- Il III grado è seguito dal VI o dal IV; meno sovente dal I, II o dal V
- Il VI grado è seguito dal V o dal I o dal II; meno sovente dal III o dal VI
- Il V grado è seguito dal I o dal IV o dal VI; meno sovente dal II o dal III
- Il VI grado è seguito dal II o dal V, a volte dal III o dal VI; meno sovente dal I
- Il VII grado è seguito dal I o dal III, a volte dal dal VI; meno sovente dal II, IV o V

Per il modo minore valgono gli stessi criteri con qualche differenza:

- Il I grado può essere seguito dalla triade maggiore del VII grado
- La triade maggiore del III grado può essere seguita dalla triade maggiore del VII grado
- La triade maggiore del VII grado è seguita dal III, a volte dal VI; più raramente dal IV
- La triade diminuita del VII è seguita dal I

Alcune di queste successioni armoniche sono talmente tipiche da avere dei nomi specifici.

- *Cadenza autentica*: V-I, allargata anche alla successioni II-V-I oppure IV-V-I. La cadenza autentica viene detta anche perfetta quando gli accordi di V e I prevedono la nota fondamentale al basso e la tonica è nella voce del soprano, cioè quella più acuta. La cadenza perfetta è una tipica chiusura di un brano musicale.
- *Cadenza sospesa*: una successione che "termina" sul V grado. Facendo un parallelo letterario, la cadenza sospesa si può paragonare ad una virgola del discorso, in quanto lascia intendere che ci sarà un prosieguo.
- *Cadenza plagale*: IV-I e, nella maggior parte dei casi, viene usata dopo una cadenza autentica.
- *Cadenza a inganno*: V-VI. Si chiama a inganno in quanto la dominante porta l'aspettativa di una cadenza autentica, ma al posto del I grado viene proposto il VI.

Le Tabelle 1.4 e 1.5 forniscono una sintesi.

Maggiore	**Successione**			
Grado	*spesso*	*a volte*	*raramente*	*mai*
$I \to$	I, IV, V	vi	ii, iii	vii°
$ii \to$	ii, V	IV, vi	I, iii	vii°
$iii \to$	iii, vi	IV	I, ii, V	vii°
$IV \to$	IV, V	I, ii	iii, vi	vii°
$V \to$	I, V	IV, vi	ii, iii	vii°
$vi \to$	ii, V, vi	iii, IV	I	vii°
$vii^\circ \to$	I, iii, vii°	vi	ii, IV, V	$-$

Tabella 1.4: Successioni tipiche nel modo maggiore. Il simbolo $-$ indica che non ci sono accordi nella corrispondente classe.

Minore	**Successione**			
Grado	*spesso*	*a volte*	*raramente*	*mai*
$i \rightarrow$	i, iv, V	VI	$ii^{\circ}, III, vii^{\circ}$	VII
$ii^{\circ} \rightarrow$	ii°, V	IV, VI	i, III	vii°
$III \rightarrow$	III, VI	iv	i, ii°, V	vii°
$iv \rightarrow$	iv, V	i, ii°	III, VI	vii°, VII
$V \rightarrow$	i, V	IV, VI	ii°, III	vii°, VII
$VI \rightarrow$	ii°, V, VI	III, iv	i	vii°, VII
$vii^{\circ} \rightarrow$	i	$-$	vii°	$-$
$VII \rightarrow$	III, VII	VI	iv	$i, ii^{\circ}, V, vii^{\circ}$

Tabella 1.5: Successioni tipiche nel modo minore. Il simbolo $-$ indica che non ci sono accordi nella corrispondente classe.

1.13 Metriche di valutazione

Un algoritmo è una sequenza ben definita di istruzioni che partendo da un input fornisce un output. L'output è collegato all'input in quanto l'input rappresenta l'istanza di un problema e l'output rappresenta una soluzione al problema. Un problema può essere di tipo decisionale: in questo caso l'output è binario, cioè è o un *si* oppure un *no*. Un esempio di problema decisionale è il seguente: un dato numero x è primo? Un esempio di istanza di questo problema è: il numero 278938829187238921 è un numero primo? Un altro tipo di problema è quello di ricerca della (o di una) soluzione; in questo caso l'output è la soluzione o una delle possibili soluzioni al problema. Un esempio di problema di ricerca della soluzione è il seguente: dato un insieme di numeri trovare il valore massimo. Una istanza di questo problema è: dato l'insieme $\{23, 5, 46, 7, 22, 54, 34, 45, 51\}$ qual è il massimo? Un probema può essere di *ottimizzazione*; in questo caso l'output è la *migliore* soluzione. Un esempio di problema di ottimizzazione è: dato un punto di partenza ed un punto di arrivo trovare la strada più breve. Un esempio di istanza di questo problema è: per andare da Taranto a Genova quale è la strada più breve?

Che tipo di problema è quello della composizione automatica? È un problema decisionale? Di ricerca di una soluzione? Di ottimizzazione? Forse sarebbe addirittura giusto chiedersi se *è* un problema! La risposta a quest'ultima domanda, almeno nel contesto in cui stiamo parlando è ovviamente sì, nel senso che se fosse no, la discussione finirebbe qui. Quindi diciamo che *vogliamo* vedere la composizione musicale come un problema matematico da risolvere con un algoritmo. A questo punto diventano lecite le altre domande.

Sicuramente non è un problema decisionale. Sarebbe troppo facile essere compositore se le uniche possibili composizioni fossero solo la composizione "si" e quella "no"! Fortunatamente la musica dà ampio spazio alla creatività permettendo la creazione di infinite possibili composizioni. Ci possono essere composizioni belle, composizione meno belle, composizioni meravigliose, decisamente brutte, mediocri, ottime, etc. Inoltre una composizione può essere bella, piacevole per una persona e meno bella per un'altra persona. Cioè la valutazione è soggettiva. Possiamo vedere il problema della composizione automatica come un problema di ricerca di una soluzione? Oppure è un problema di ottimizzazione?

La risposta non è facile. Il problema di fondo è dovuto al fatto che gli algoritmi possono risolvere problemi matematicamente ben formulati mentre il problema della composizione musicale non ha una definizione formale. Un problema di ottimizzazione è legato alla presenza di una metrica che permetta di confrontare due soluzioni per stabilire quale delle due è migliore o se sono equivalenti. Sfortunatamente (o forse fortunatamente) per le composizioni musicali non esiste una metrica che permetta di dire che una composizione è "migliore" di un'altra. É migliore la musica di Beethoven, o quella di Bach? Quella dei musical di Broadway o quella dei gruppi rock? Si potrebbe rispondere che sono tutti generi e/o stili diversi quindi un confronto non ha molto senso. Ma anche stabilendo un preciso genere musicale e un preciso tipo di composizione il problema rimane. Quale è la "migliore" fra le nove sinfonie di Beethoven? Di fatto non esiste una metrica formale che permetta di dire che una composizione è migliore di un'altra. Questo rende difficile considerare il problema della composizione come un problema di ottimizzazione. A dire il vero non esiste nemmeno una definizione che permetta di dire che un insieme di note costituisce una composizione mentre un altro insieme no. E questo rende difficile considerare il problema della composizione musicale come un problema di ricerca di una soluzione. Siamo quindi in un vicolo cieco? Comporre è esclusivamente un arte ed un procedimento automatico non può servire a niente o comunque non a molto? Una facile risposta a questa domanda è "si". Tuttavia l'uso di algoritmi per la composizione musicale automatica è molto intrigante e non pochi sono coloro, sia musicisti che non, che si sono cimentati con questo dilemma.

Ovviamente per affrontare algoritmicamente il problema della composizione musicale è necessario usare metrica. Abbiamo però appena detto che una tale metrica non esiste. Vero. Ma esistono delle regole ben formulate che vengono utilizzate dai compositori nella creazione delle opere musicale. Tali regole possono in qualche modo costituire la base per la definizione di una metrica di valutazione. Nel prosieguo vedremo come definire delle metriche di valutazione.

Esercizi

1. Quali sono le principali caratteristiche di un suono? Cosa determina ognuna di esse?

2. Si scriva una funzione con dominio $\mathbb{Z}$ e codominio $\mathbb{R}$ che fornisca tutte le ottave di una data frequenza di riferimento f_r.

3. Si scriva una funzione con dominio $\{1, 2, \ldots, 12\}$ e codominio $\mathbb{R}$ che fornisca i 12 semitoni di un ottava a partire da una frequenza di riferimento f_r nel sistema ben temperato.

4. Siano $n_1, n_2, \ldots, n_i$ gli intervalli, misurati in semitoni, di una scala musicale che copre un'ottava. Quanto vale la loro somma $\sum_{j=1}^{i} n_i$? Perchè?

5. Che cosa è la scala cromatica?

6. Siano f_1 e f_2 le frequenze di due suoni. Supponendo che f_1 e f_2 differiscono di pochi Hz, quando è più facile distinguere le due note? Quando le due frequenze sono basse o quando le due frequenza sono alte? Perchè?

7. In una rappresentazione della musica su un sistema di assi cartesiani cosa viene rappresentato sugli assi? E a cosa corrisponde una nota in una tale rappresentazione?

8. A che velocità (approssimativa) si propaga il suono nell'aria? Se cambia la temperatura cosa succede a tale velocità?

9. Quale è l'estensione in frequenze di un pianoforte? E quali sono i limiti (approssimativi) che l'orecchio umano può percepire?

10. Nel circolo delle quinte due scale successive in quante alterazioni differiscono? Perché?

11. Volendo rappresentare una musica su assi cartesiani in cui l'asse (orizzontale) delle ascisse rappresenta il tempo, a quale asse associeresti la melodia e a quale asse l'armonia?

12. Fornisci una caratterizzazione degli accordi maggiori, minori, diminuiti e aumentati in termini di intervalli di semitoni.

13. Da quante voci è formato un corale? E come si chiamano?

Capitolo 2

Temperamento equabile

In questo capitolo approfondiremo la costruzione delle scale e il temperamento equabile. Quindi, facendo un passo indietro, contrariamente a quanto abbiamo visto nel capitolo precedente, le note ancora non corrispondono a delle precise frequenze. Analizzeremo le considerazioni che hanno portato al sistema già descritto, che viene detto temperamento equabile.

2.1 Consonanza e dissonanza

Il termine consonanza, che deriva dal latino *consonare*, cioè suonare insieme, è usato per indicare suoni che ascoltati insieme producono un effetto gradevole. Al contrario il termine dissonanza, che deriva dal latino *dissonare*, cioè suonare con separazione, è usato per indicare suoni che ascoltati insieme producono un effetto sgradevole.

Il grado di consonanza o dissonanza può avere delle giustificazioni fisico-matematiche, come vedremo in seguito, ma può anche dipendere da cultura, abitudini e gusto soggettivo.

Ma cosa rende due suoni consonanti oppure dissonanti? La teoria dei suoni armonici e quella dei rapporti matematici semplici cercano di spiegare questo aspetto.

2.2 Suoni armonici e ottave

I suoni sono prodotti da corpi in vibrazione. L'altezza del suono, o della nota che associamo a quel suono, dipende dalla frequenza f con la quale vibra il corpo che lo produce. Quando un corpo (ad esempio una corda di un violino) vibra con una frequenza f, oltre al suono con frequenza f, chiamato fondamentale, vengono prodotti anche altri suoni, molto più deboli quindi molto più difficili da percepire, che hanno frequenze multiple di f:

$$f, 2f, 3f, 4f, 5f, 6f, 7f, \ldots$$

Il primo suono armonico corrisponde al suono fondamentale con frequenza f, il secondo armonico corrisponde alla frequenza $2f$ e così via (l'n-esimo armonico corrisponde alla frequenza $n \cdot f$). L'intensità degli armonici decresce rapidamente al crescere del numero armonico: di fatto l'orecchio percepisce il suono fondamentale, anche se sono presenti i suoni armonici. Il fatto che mediamente si percepisce solo il suono fondamentale è dovuto in primo luogo all'intensità degli altri armonici che, a partire già dal secondo, è molto più bassa. In secondo luogo è dovuto anche al fatto che i suoni armonici, poichè prodotti dalla vibrazione del corpo che produce il suono fondamentale, sono in sostanza "parte" del suono fondamentale. Quindi, se consideriamo le singole frequenze, possiamo dire che il suono con frequenza $2f$ è molto consonante rispetto al suono con frequenza f, in quanto $2f$ è il secondo armonico di f.

Questo dato di fatto fornisce un modo per definire la consonanza. Basandoci sui suoni armonici possiamo dire che i due suoni più consonanti rispetto a un suono con frequenza f sono quelli corrispondenti ai suoi armonici, nell'ordine della serie degli armonici. Quindi i due suoni più consonanti corrispondono al suono fondamentale (frequenza f) e al secondo suono armonico (frequenza $2f$). Il successivo suono nella serie corrisponde alla frequenza $3f$, che rispetto al secondo armonico ha frequenza $3/2f'$, dove $f' = 2f$. Considerando ancora il successivo suono armonico, il quarto, con frequenza $4f$, si ha che la sua frequenza è $4/3f''$ dove $f'' = 3f$ è la frequenza del terzo armonico.

Una prospettiva diversa ma con conlcusioni simili è quella pitagoriana secondo la quale i suoni più consonanti sono quelli i cui rapporti fra le rispettive frequenze possono essere espressi come rapporti di numeri interi piccoli, come $2/1$, $3/2$ e $4/3$. Tale punto di vista è giustificato dalle seguenti considerazioni empiriche: se una corda di una certa lunghezza produce un suono con frequenza f, allora (i) facendo vibrare solo metà ($1/2$) della corda si ottiene un suono consonante che ha frequenza $2f$; (ii) facendo vibrare $2/3$ della corda si ottiene un suono consonante che ha frequenza $3/2f$; (iii) facendo vibrare $3/4$ della corda si ottiene un suono consonante che ha frequenza $4/3f$. Questi rapporti corrispondono proprio ai rapporti fra i primi suoni armonici.

L'intervallo definito dal rapporto $2/1$ viene chiamato *ottava*[1]. Due suoni distanti un'ottava sono talmente consonanti che vengono in pratica considerati "uguali". Infatti le note le cui frequenze sono una il doppio dell'altra hanno lo stesso nome. La Figura 2.1 mostra tutte le note `Do` del pianoforte.

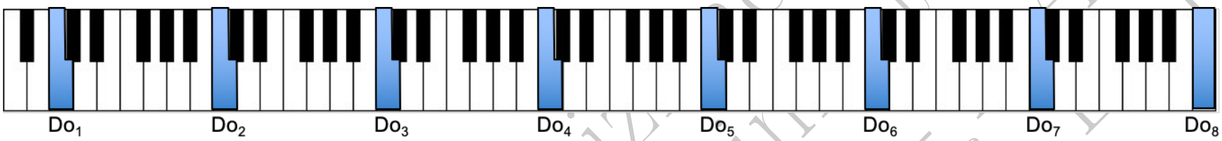


Figura 2.1: Le note `Do` di un pianoforte

[1]Come abbiamo spiegato nel capitolo precedente, il termine *ottava* deriva dal fatto che le scale hanno 7 suoni e quindi l'ottavo corrisponde alla nota di partenza. Per lo stesso motivo gli intervalli fra note della scala prendono i nomi di seconda, terza, etc. in funzione della "distanza" in termini di note della scala.

Tipo	Intervallo	Rapporto	Somma
Intervalli perfetti	Unisono	1/1	1+1=2
	Ottava	2/1	2+1=3
	Quinta	3/2	3+2=5
	Quarta	4/3	4+3=7
Intervalli imperfetti	Sesta maggiore	5/3	5+3=8
	Terza maggiore	5/4	5+4=9
	Terza minore	6/5	6+5=11
	Sesta minore	8/5	8+5=13
Intervalli dissonanti	Seconda maggiore	9/8	9+8=17
	Settima maggiore	15/8	15+8=23
	Settima minore	16/9	16+9=25
	Seconda minore	16/15	16+15=31
	Tritono	64/45	64+45=109

Tabella 2.1: Classificazione degli intervalli fra suoni armonici

L'eccezionale consonanza delle ottave viene presa come un *assioma* musicale: note con frequenze una doppia dell'altra sono la stessa nota anche se una più acuta e l'altra più grave.

Sfruttando questo assioma musicale, possiamo in qualche modo restringere la nostra attenzione a una singola ottava. Tutto ciò che succede in questa singola ottava potrà essere ripetuto nelle altre ottave con una semplice proporzione per cambiare le frequenze. In realtà questo è possibile nell'attuale sistema musicale, denominato temperamento equabile, mentre in altri sistemi questa corrispondenza non è così semplice, come cercheremo di spiegare in questo capitolo.

La Tavola 2.1 riporta una classificazione di alcuni intervalli che compaiono nei suoni armonici. L'ordine va dagli intervalli più consonanti a quelli meno consonanti.

2.3 Quali suoni utilizziamo all'interno di un'ottava?

All'interno di un'ottava ci sono infiniti suoni (ognuno corrispondente ad una frequenza fra f e $2f$). Infiniti suoni sono un po' troppi per essere usati in pratica. Quali scegliamo per costruire la nostra "scala"? La scala è appunto l'insieme dei suoni che utilizziamo per comporre musica. Nel capitolo precedente abbiamo identificato una scala con un insieme di intervalli misurati in semitoni la cui somma è 12; questa definizione si basa sul concetto di semitono che ora non abbiamo (ancora). Quindi più genericamente la scala è una successione di suoni il cui primo suono ha frequenza f e il cui ultimo suono ha frequenza $2f$.

Nel prosieguo del capitolo useremo i nomi delle note `Do`, `Re`, etc. Tuttavia il loro uso è solo indicativo, nel senso che le esatte frequenze non sono quelle definite nel capitolo precedente. L'obiettivo di questo capitolo è proprio quello di individuare quali sono le migliori frequenze da utilizzare per le note. Vari criteri portano a scelte diverse. La conclusione finale porterà al sistema del temperamento equabile che fornisce le frequenze già descritte. Prima di arrivare alla conclusione però i nomi delle note sono da intendersi "approssimati".

2.4 L'aritmetica degli intervalli

Prima di proseguire con la discussione sulla costruzione delle scale è oppurtuno ricordare "l'aritmetica degli intervalli" che ci sarà molto utile per la costruzione delle scale. Per comodità di scrittura rappresenteremo un intervallo fra due suoni con frequenza a e b, con $a < b$, usando la notazione $[a, b]$ in analogia con gli intervalli numerici. Come sappiamo, l'intervallo $[f_1, f_2]$ è rappresentato dal rapporto delle frequenze $i_1 = f_2/f_1$. In realtà i_1 è l'*ampiezza* dell'intervallo. Supponiamo ora di avere un secondo intervallo, contiguo al precedente, $[f_2, f_3]$, la cui ampiezza è $i_2 = f_3/f_2$. Poichè gli intervalli sono contigui la loro "somma" è l'intervallo $[f_1, f_3]$ la cui ampiezza è $i_3 = f_3/f_1$. Non è difficile verificare che $i_3 = i_1 \cdot i_2$. Quindi per sommare due intervalli contigui è necessario *moltiplicare* le loro ampiezze. Analogamente, per sottrarre un intervallo da un altro è necessario dividere l'ampiezza del primo per l'ampiezza del secondo.

Si noti che le operazioni di somma e sottrazione degli intervalli sono intuitive quando si sommano due intervalli contigui o quando si sottrae da un dato intervallo un'altro intervallo più piccolo il cui inizio o la cui fine coincide con il primo. Facendo riferimento alla Figura 2.2 si ha che $i_3 = i_1 \cdot i_2$ e ovviamente che $i_1 = i_3/i_2$ e anche $i_2 = i_3/i_1$.

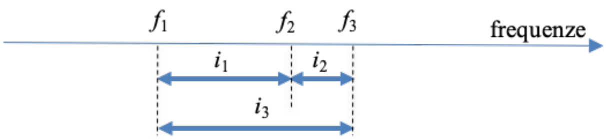


Figura 2.2: Somma e sottrazione di intervalli

Consideriamo un esmepio numerico. Se $f_1 = 200, f_2 = 300$ e $f_4 = 400$, si ha che $i_1 = 3/2$ e $i_2 = 4/3$. La somma di i_1 e i_2 è $i_3 = \frac{3}{2}\frac{4}{3} = 2$ che corrisponde all'intervallo fra la frequenza 200 e la frequenza 400. Infatti mettendo insieme $[200, 300]$ e $[300, 400]$ si ottiene $[200, 400]$ che è un'ottava.

Analogamente se dall'intervallo $[400, 700]$ che corrisponde alla frazione $7/4$ vogliamo sottrarre l'intervallo $[400, 500]$, che corrisponde alla frazione $5/4$, dobbiamo dividere $7/4$ per $5/4$ ottenendo $7/5$ che corrisponde all'intervallo $[500, 700]$.

Si noti come gli intervalli usati siano contigui. Tuttavia, da un punto di vista matematico, la definizione di somma e sottrazione degli intervalli è sempre valida, anche se operare su intervalli non contigui non ha molto senso da un punto di vista musicale. Per cui useremo sempre intervalli contigui.

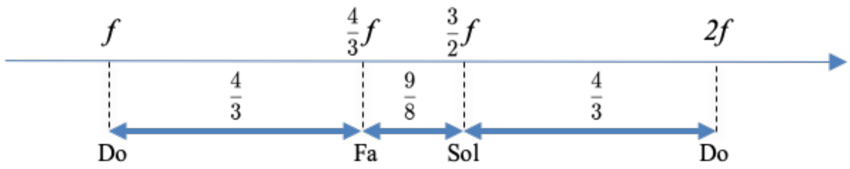
Per calcolare l'ampiezza di un intervallo divideremo la frequenza maggiore per la frequenza minore in modo che il risultato sia sempre maggiore o al più uguale a 1. In realtà, anche per questo aspetto, da un punto di vista matematico possiamo senza problemi dividere la frequenza minore per quella maggiore ottenendo un intervallo minore di 1. Dal punto di vista pratico si potrebbe pensare che gli intervalli maggiori di 1 vanno verso destra, cioè verso frequenze più alte, mentre gli intervalli minori di 1 vanno verso sinistra cioè verso frequenze più basse.

La somma di intervalli può essere usata per spostarsi di un dato intervallo da una frequenza di riferimento. Ad esempio partendo da una frequenza di riferimento f, se si somma ad f un intervallo i si ottiene una nuova frequenza $f' = f \cdot i$ che è a distanza i da f.

Per rendere più concreto il discorso facciamo un esempio. Assumiamo di partire dalla frequenza di riferimento $f = 440Hz$. Per "salire" di un intervallo di ampiezza $3/2$ basterà moltiplicare f per $3/2$ ottenenedo $660Hz$. Per "scendere" dello stesso intervallo basterà dividere per $3/2$, ottenendo $229, \bar{3}Hz$.

2.5 Scala pitagorica

Questa scala di 7 note nasce dall'idea, della scuola pitagoriana, che le consonanze migliori sono date dai rapporti che usano numeri interi piccoli. In particolare, l'ottava corrisponde al rapporto 2/1, la quinta al rapporto 3/2 e la quarta al rapporto 4/3. Partiamo dall'ottava all'interno della quale vengono selezionate delle frequenze che corrispondono ai rapporti d 3/2 e 4/3:



L'intervallo fra `Fa` e `Sol`, corrispondente al rapporto $\frac{9}{8}$, può essere usato per selezionare altre note. Ad esempio partendo dal `Do`, con frequenza f, si può selezionare la frequenza $\frac{9}{8}f$ (`Re`) e la frequenza $\frac{81}{64}f$ (`Mi`). Analogamente partendo dal `Sol`, con frequenza $\frac{3}{2}f$ si possono ottenere le note con frequenza $\frac{3}{2}\frac{9}{8}f = \frac{27}{16}f$ (`La`) e $\frac{27}{16}\frac{9}{8}f = \frac{243}{128}f$ (`Si`).

Si ottiene così la scala scala pitagorica i cui intervalli sono mostrati nella Figura 2.3.

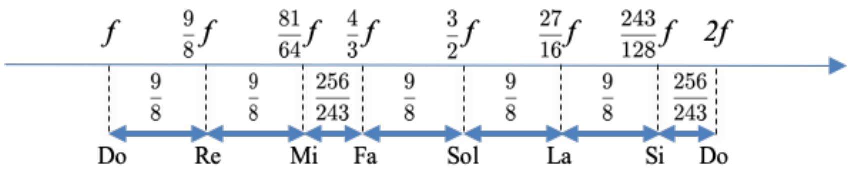


Figura 2.3: Scala pitagorica

Nella scala pitagorica esistono due intervalli fra suoni successivi, uno definito dal rapporto 9/8, chiamato tono pitagorico, e l'altro definita dal rapporto 256/243 chiameto semitono pitagorico. È evidente che un semitono pitagorico non è metà di un tono pitagorico; infatti sommando due semitoni pitagorici non si ottiene un tono pitagorico perchè $\left(\frac{256}{243}\right)^2 \simeq 1.10985$ che è diverso da $\frac{9}{8} = 1.125$. E ovviamente, la metà di un tono pitagorico non corrisponde a un semitono pitagorico in quanto il tono corrisponde a $\frac{\sqrt{9}}{\sqrt{8}} \simeq \frac{3}{2.8284} \simeq 1.0606$, mentre un semitono corrisponde a $\frac{256}{243} = 1.0534$. La differenza è molto piccola, ma come vedremo fra poco ha delle conseguenze molto importanti.

2.6 Scala Naturale

Sebbene la scala pitagorica parta dall'idea di usare degli intervalli consonanti espressi da rapporti che coinvologno numeri interi piccoli e che in effetti corrispondono ai primi suoni armonici (2:1 all'ottava, 3:2 alla quinta e 4:3 alla quarta), alcuni degli altri intervalli della scala, in particolare 81/64, 27/16 e 243/128 sono abbastanza diversi dagli intervalli che ritroviamo fra i suoni armonici. Fra i vari tentativi di modificare la scala pitagorica per avere intervalli più consonanti, quello più noto è la scala Zarliniana o Naturale, sviluppata nel XVI secolo dal teorico musicale Giuseppe Zarlino. Poichè i suoni armonici sono consonanti, una scelta che in qualche modo corrisponda ai suoni armonici sembra quella più "naturale". La scala naturale si basa appunto sulle consonanze dei suoni armonici.

Il primo e il secondo suono armonico secondo, dato il nostro assioma musicale dell'ottava, rappresentano la quintessenza della consonanza. I suoni armonici non si fermano al secondo armonico che abbiamo sfruttato per definire l'intervallo di ottava. Ad esempio possiamo considerare l'intervallo fra il secondo e il terzo armonico. Il secondo armonico ha frequenza $2f$ mentre il terzo armonico ha frequenza $3f$, pertanto l'intervallo che intercorre fra questi due suoni è descritto dal rapporto $3/2$. Analogamente l'intervallo che intercorre fra il terzo e il quarto armonico è descritto dal rapporto $4/3$. L'intervallo descritto dal rapporto $3/2$ viene detto intervallo di quinta, mentre quello descritto dal rapporto $4/3$ viene detto intervallo di quarta. L'intervallo descritto dal rapporto $5/4$ viene dette intervallo di terza.

Usando gli intervalli fra i suoni armonici possiamo costruire la scala naturale: La costruzione matematica di questa scala è basata sul rapporto $5/4$ (terza maggiore) e sul rapporto $3/2$ (quinta). Per costruire la scala si parte dalla nota fondamentale (prendiamo come esempio il `Do`) e si aggiungono la terza e la quinta (`Do-Mi-Sol`) ottenendo così la triade maggiore. A questo punto si parte da `Sol` e si ripete il procedimento ottenendo un'altra triade maggiore (`Sol-Si-Re`). La nota (`Re`) che va oltre l'ottava viene riportata nell'ottava di riferimento. Per ottenere le altre due note (`Fa` e `La`) si procede in modo simile considerando il `Do` come la quinta della triade `Fa-La-Do` e quindi si scende di una quinta dal `Do` con frequenza $2f$ per ottenere il `Fa` dal quale si ottiene il `La`. La scala che ne deriva è riportata nella Figura 2.4.

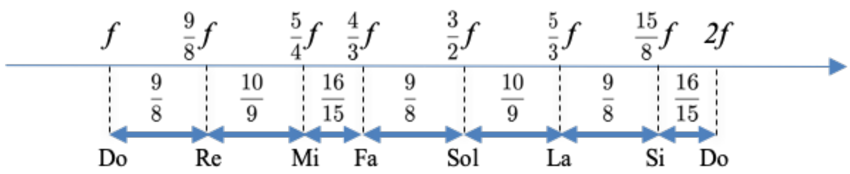


Figura 2.4: Scala Naturale

In questa scala naturale l'intervallo di seconda è fornito dall'intervallo fra l'8^o e il 9^o suono armonico, quello di terza fra il 4^o e il 5^o suono armonico, quello di quarta fra il 3^o e il 4^o suono armonico, dello di quinta fra il 2^o e il 3^o suono armonico, quello di sesta fra il 3^o e il 5^o suono armonico e quello di settima fra l'8^o e il 15^o suono armonico.

Come si può facilmente verificare, gli intervalli fra i gradi della scala sono 3: un tono naturale grande ($9/8$) un tono naturale piccolo ($10/9$) e un semitono naturale ($16/15$). La differenza fra tono naturale grande e tono naturale piccolo, pari a $81/80$, viene detta *comma sintonico*.

Sebbene la scala naturale utilizzi degli intervalli che corrispondono a rapporti fra numeri interi più piccoli rispetto alla scala pitagorica, nella scala naturale le distanze fra note successive sono di tre tipi: $9/8$, che è uguale al tono pitagorico, $10/9$ chiamato tono zarliniano ed è leggermente più grande di un tono pitagorico, e infine $16/15$ chiamato semitono zarliniano (leggermente più grande del semitono pitagorico $256/243$). Anche in questo caso come per la scala pitagorica l'intervallo chiamato semitono non è la metà di un tono; inoltre abbiamo due diversi intervalli di tono.

2.7 Scala delle quinte

L'intervallo di quinta armonico, cioè l'intervallo rappresentato dalla frazione $3/2$, può essere utilizzato per costruire una scala con 12 suoni: la scala cromatica pitagorica. Tale scala si basa solo sull'intervallo di quinta armonico. Per costruire tale scala si parte dalla nota di riferimento che come al solito supponiamo essere un `Do` e si selezionano altre frequenze moltiplicando (o dividendo) per $3/2$.

Ad esempio, alzando di una quinta il `Do`, $f = 1$, si ottiene il `Sol` che corrisponde $3/2$, mentre alzando il `Sol` di un'altra quinta si ottiene un `Re` dell'ottava successiva: $\left(\frac{3}{2}\right)^2 = \frac{9}{4}$. Tale nota appartiene all'ottava successiva in quanto $2 < \frac{9}{4} < 4$. Per riportare questo `Re` nell'ottava di riferimento potremmo dividere la frazione che lo rappresenta per 2 ottenendo $\frac{9}{4}/2 = \frac{9}{8}$.

Per ottenere altre note si può continuare ad alzare la frequenza di un intervallo di quinta. Alzando il `Re` di una quinta si ottiene un `La`: $\frac{9}{4} \cdot \frac{3}{2} = \frac{27}{8}$. Alzando il `La` di un'altra quinta si ottiene un Mi dell'ottava successiva: $\frac{27}{8} \cdot \frac{3}{2} = \frac{81}{16}$. Il Mi si colloca nella terza ottava in quanto $4 < \frac{81}{16} < 8$. Continuando a moltiplicare per 3/2 si ottengono altre note; la sequenza "completa" è:

Do, Sol, Re, La, Mi, Si, Fa♯, Do♯, Sol♯, Re♯, La♯, Mi♯≃Fa, Si♯≃Do.

La Figura 2.5 mostra gli intervalli, rispetto al Do iniziale di ognuna di queste note. Per riportare ogni nota nell'ottava di riferimento, cioè la prima, è necessario dividere per l'opportuna potenza di 2. La Figura 2.6 mostra il risultato. Come si può notare il Do all'ottava non corrisponde al rapporto 2/1, come invece dovrebbe essere l'ottava. Se si divide ulteriormente per 2 non si ottiene, ovviamente, 1 che corrisponde al Do iniziale, ma una nota leggermente più alta rappresentata dal rapporto $\frac{531441}{524288} \simeq 1.0136$, che viene detto *comma* pitagorico.

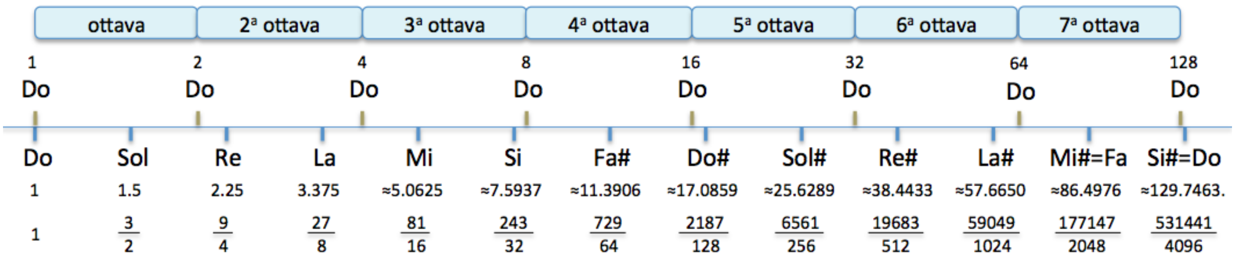


Figura 2.5: Quinte su sette ottave

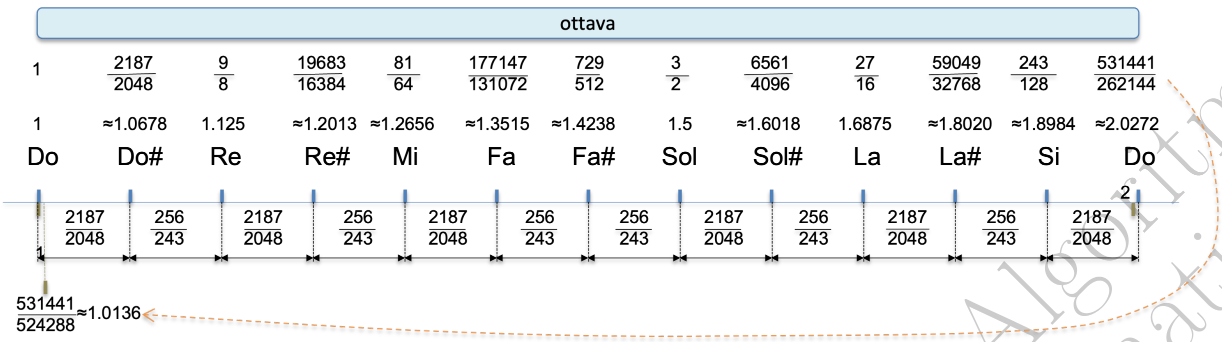


Figura 2.6: Scala cromatica delle quinte

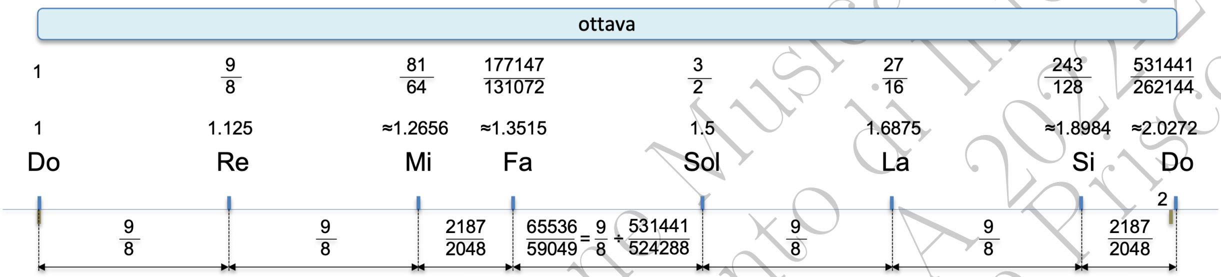


Figura 2.7: Scala delle quinte

2.8 Trasposizione

Trasporre un brano musicale significa traslare tutte le note (frequenze) verso l'alto o verso il basso in modo tale da mantenere inalterati gli intervalli. Dall'aritmetica degli intervalli sappiamo che tale operazione è molto semplice: basta scegliere l'ampiezza x dell'intervallo di trasposizione e moltiplicare (o dividere) tutte le frequenze per x. Il problema che sorge con la trasposizione riguarda la corrispondenza delle note trasposte con quelle già esistenti. Tale problema è particolarmente cruciale per gli strumenti a tasti: per essi è necessario associare un tasto ad ogni possibile nota (frequenza) che si deve suonare.

Per rendere concreto il discorso, supponiamo di usare la scala pitagorica. Per tale scala abbiamo bisogno di 7 tasti per ogni ottava. Supponiamo ora di voler trasporre un brano musicale traslando verso le frequenze alte di un intervallo pari all'intervallo esistente fra il `Do` e il `Re`, cioè di un intervallo pari a 9/8.

Dunque, costruiamo una scala pitagorica a partire dal `Re`. Se chiamiamo f la frequenza di riferimento per la costruzione della scala pitagorica a partire dalla nota `Do` (appunto identificata dalla frequenza f) e chiamiamo $f' = \frac{9}{8}f$ la frequenza del `Re` possiamo sfruttare la scala costruita precedentemente per ottenere le frequenze della nuova scala che parte dal `Re`., come mostrato nella seguente tabella.

nota / scala	`Do`	`Re`	`Mi`	`Fa`	`Fa`$\sharp$	`Sol`	`La`	`Si`	`Do`	`Do`$\sharp$	`Re`
`Do`	f	$\frac{9}{8}f$	$\frac{81}{64}f$	$\frac{4}{3}f$		$\frac{3}{2}f$	$\frac{27}{16}f$	$\frac{243}{128}f$	$2f$		
`Re`		$\frac{9}{8}f$	$\frac{81}{64}f$		$\frac{729}{512}f$	$\frac{3}{2}f$	$\frac{27}{16}f$	$\frac{243}{128}f$		$\frac{2187}{1024}f$	$\frac{9}{4}$

Come si può vedere, mentre il `Re`, il `Mi`, il `Sol`, il `La` e il `Si` hanno esattamente la stessa frequenza nelle due scale, il `Fa` e il `Do` invece corrispondono a due rapporti diversi. Nella scala di `Do`, il `Fa` corrisponde al rapporto $4/3 \simeq 1.3333$, nella scala di `Re` il `Fa` corrisponde al rapporto $729/512 \simeq 1.4238$. Nella scala di `Do`, il `Do` (all'ottava) corrisponde al rapporto $2/1 = 2$, nella scala di `Re` il `Do` corrisponde al rapporto $2187/1024 \simeq 2.1357$.

Questo significa che il "`Fa`" e il "`Do`" della nuova scala sono in realtà due nuove note da includere nel sistema. Queste due note corrispondono più o meno al `Fa`$\sharp$ e al `Do`$\sharp$. Ovviamente per queste due nuove note avremo bisogno di 2 nuovi tasti, pertanto il fabbisogno di tasti è passato da 7 a 9 per ogni ottava. Sebbene si potrebbe pensare che questo non sia un problema visto che in effetti il `Fa`$\sharp$ e il `Do`$\sharp$ effettivamente esisteranno (come sappiamo l'ottava verrà divisa in 12 note), queste due nuove note non corrispondono esattamente a quelli che saranno il `Fa`$\sharp$e il `Do`$\sharp$. Più precisamente, proseguendo con la costruzione di altre scale a partire dalle note esistenti si creeranno altre note che in toeria dovrebbero essere uguali a note già esistenti ma che non lo sono.

Per capire meglio costruiamo altre scale pitagoriche, a partire dalle altre note, come mostrato nella Tabella 2.2. Con la scala che parte dal `Mi` vengono create due nuove note che dovrebbero corrispondere a `Sol`$\sharp$ e `Re`$\sharp$. Si noti come questa scala usa il `Fa`$\sharp$ e il `Do`$\sharp$ creati con la scala di `Re`; infatti le frequenze combaciano perfettamente. Poi la scala di `Fa` crea la nuova nota `La`$\sharp$ (che musicalmente sarebbe un `Si`$\flat$), mentre le altre note combaciano con quelle esistenti. La scala di `Sol` non crea nuove note, tutte le note combaciano con quelle esistenti. Lo stesso accade per la scala di `La`. Arriviamo ora alla scala costruita a partire dal `Si`. Questa scala genera una nota che dovrebbe corrispondere al `La`$\sharp$. La frazione che descrive la frequenza è $\frac{59049}{16834}$. Tuttavia un `La`$\sharp$ era già stato creato nell'ottava più bassa dalla scala di `Fa`, e questo `La`$\sharp$ corrisponde alla frazione $\frac{16}{9}$. E $\frac{59049}{16834}$ non è il doppio di $\frac{16}{9}$. Quindi questi 2 suoni non sono uno l'ottava dell'altro. In altre parole sono 2 `La`$\sharp$ diversi! Non moltissimo. Infatti $\frac{16}{9} = 1.\bar{7}$ e $\frac{59049}{16834} \simeq 3.51$ che è quasi il doppio di $1.\bar{7}$. Quasi, ma non il doppio.

E non è un problema isolato. Ad esempio considerando che abbiamo un `Do`$\sharp$ corrispondente a $\frac{2187}{1024}$, che riportato nell'ottava più bassa diventa $\frac{2187}{2048}$, possiamo costruire una nuova scala a partire da questa nota.

	nota																							
scala	Do	Do♯	Re	Re♯	Mi	Fa	Fa♯	Sol	Sol♯	La	La♯	Si	Do	Do♯	Re	Re♯	Mi	Fa	Fa♯	Sol	Sol♯	La	La♯	Si
Do	f		$\frac{9}{8}f$		$\frac{81}{64}f$	$\frac{4}{3}f$		$\frac{3}{2}f$		$\frac{27}{16}f$		$\frac{243}{128}f$	$2f$											
Re			$\frac{9}{8}f$		$\frac{81}{64}f$		$\frac{729}{512}f$	$\frac{3}{2}f$		$\frac{27}{16}f$		$\frac{243}{128}f$		$\frac{2187}{1024}f$	$\frac{9}{4}f$									
Mi					$\frac{81}{64}f$		$\frac{729}{512}f$		$\frac{6561}{4096}f$	$\frac{27}{16}f$		$\frac{243}{128}f$		$\frac{2187}{1024}f$		$\frac{19683}{8192}f$	$\frac{81}{32}f$							
Fa						$\frac{4}{3}f$		$\frac{3}{2}f$		$\frac{27}{16}f$	$\frac{16}{9}f$		$2f$		$\frac{9}{4}f$		$\frac{81}{32}f$	$\frac{8}{3}f$						
Sol								$\frac{3}{2}f$		$\frac{27}{16}f$		$\frac{243}{128}f$	$2f$		$\frac{9}{4}f$		$\frac{81}{32}f$		$\frac{729}{256}f$	$3f$				
La										$\frac{27}{16}f$		$\frac{243}{128}f$		$\frac{2187}{1024}f$	$\frac{9}{4}f$		$\frac{81}{32}f$		$\frac{729}{256}f$		$\frac{6561}{2048}f$	$\frac{27}{8}f$		
Si												$\frac{243}{128}f$		$\frac{2187}{1024}f$		$\frac{19683}{8192}f$	$\frac{81}{32}f$		$\frac{729}{256}f$		$\frac{6561}{2048}f$		$\frac{59049}{16384}f$	$\frac{243}{64}f$
Do♯		$\frac{2187}{2048}f$		$\frac{19683}{16384}f$		$\frac{177147}{131072}f$	$\frac{729}{512}f$		$\frac{6561}{4096}f$		$\frac{59049}{32768}f$		$\frac{531441}{262144}f$	$\frac{2187}{1024}f$										

Tabella 2.2: Frequenze derivanti dalle trasposizioni della scala pitagorica

Questa nuova scala genera un Fa corrispondente alla frazione $\frac{177147}{131072}$ che è diverso dal Fa che già avevamo corrispondente alla frazione $\frac{4}{3}$. Anche in questo caso i suoni sono molto vicini ($\frac{177147}{131072} \simeq 1.35$ e $\frac{4}{3} = \simeq 1.33$) ma non sono uguali!

Si può facilmente intuire che costruendo altre scale (a partire dalle altre note) si avranno incongruenze simili. Ovviamente nulla vieta di considerare anche le nuove note come punto di partenza di nuove scale. Tuttavia poichè nuove scale produrranno sempre nuove note e da nuove note si possono costruire altre scale, in pratica non finiremo mai di introdurre nuove note nel nostro sistema musicale. Pertanto avremo bisogno di troppe note e troppi tasti!

Il problema appena descritto non è specifico della scala pitagorica. Anche la scala naturale e la scala delle quinte lo generano.

2.9 Temperamento

Il temperamento (dal latino *temperare*, cioè *mescolare nelle giuste proporzioni*) musicale fa riferimento alle soluzioni adottate per risolvere il problema della trasposizione. L'idea di base è quella di alterare gli intervalli presenti nelle scale. Occorre, infatti, che la distanza (cioè l'ampiezza degli intervalli) fra le note sia costante in modo tale che le trasposizioni non creino nuove note! Ostinarsi però a voler usare intervalli descritti da frazioni porta a distanze simili ma diverse fra le note. Poiché sommando due intervalli si ottiene un intervallo che è il prodotto, per dividere l'ottava in 12 suoni equidistanti fra loro occorre scegliere come distanza fra una nota e quella successiva il valore x tale che

$$x^{12} = 2.$$

Pertanto si ha che la distanza fra due note successive deve essere

$$x = \sqrt[12]{2} \simeq 1.059463094.$$

Quindi la soluzione al problema mette in gioco i numeri irrazionali, che non possono essere espressi come frazioni.

Senza i numeri irrazionali la soluzione era impossibile da trovare. In realtà la scuola Pitagoriana era già arrivata alla soluzione[2], ma, non fu adottata, probabilmente a causa del burrascoso rapporto dei pitagoriani con i numeri irrazionali. Un'altra motivazione è dovuta alla difficoltà di accordare gli strumenti su intervalli non "naturali". L'adozione del sistema equabile sì è avuta solo nel XVI secolo.

[2] In [2] la divisione dell'ottava in parti uguali è attribuita a Archita da Taranto (428 a.C.-360 a.C.).

Prima del temperamento equabile, nel corso della storia, vari "temperamenti" (imperfetti) sono stati proposti. Un tentativo fu proposto da Vincenzo Galilei, padre del più famoso Galileo. Vincenzo Galilei (1520-1591) propose di usare un semitono costante, ma razionale, pari a

$$18/17 \simeq 1.05882353 \simeq \sqrt[12]{2}.$$

Ovviamente il problema non viene risolto, tuttavia è un passo verso la risoluzione in quanto contiene l'intuizione della distanza costante fra le note del sistema.

La soluzione adottata da Galilei fu criticata da Simone Stevino (ingegnere, fisico e matematico fiammingo, 1548-1620) che invece sostenne la suddivisione con semitono pari a $\sqrt[12]{2}$; ormai i numeri irrazionali non erano più eresia. Questo temperamento, detto *equabile*, tuttavia non venne adottato subito principalmente perchè poneva dei problemi per l'accordatura degli strumenti data la mancanza di intervalli naturali.

Pertanto altri temperamenti sono stati utilizzati. Fra questi di particolare rilevanza è quello introdotto da Adreas Werckmeister (1645-1706) che propose un temperamento basato su 5 quinte *mesotoniche* e 7 quinte naturali. Una quinta mesotonica è un intervallo usato in un precedente temperamento, detto mesotonico, in cui l'intervallo di quinta è pari a $\sqrt[4]{5} \simeq 1.495$ che è sufficientemente vicino a $3/2$, che è la quinta giusta (o naturale), presente nei suoni armonici. Come già sappiamo l'intervallo di quinta naturale è di $3/2$. Il temperamento proposto da Werkmesiter, detto *temperamento buono*, non corrisponde ovviamente al temperamento equabile e infatti se si prova a salire per quinte, come abbiamo fatto per il circolo delle quinte, usando 5 quinte mesotoniche e 7 quinte naturali, si ottiene

$$(\sqrt[4]{5})^5(3/2)^7 \simeq 127.75$$

che differisce, anche se di poco, dal valore 128 necessario per far chiudere il circolo delle quinte (il valore $128 = 2^7$ corrisponde a 7 ottave).

Dunque il temperamento buono di Werckmesiter da un lato approssima sufficientemente bene il temperamento equabile, dall'altro utilizza intervalli di quinte naturali che rendono l'accordatura più semplice, visto che secoli fa non c'erano strumenti di ausilio per accordare gli strumenti musicali. Per tali caratteristiche il temperamento buono fu molto apprezzato e gli strumenti accordati con tale tecnica venivano detti *ben temperati.* Il temperamento buono è stato anche apprezzato da Joahn Sebastian Bach (1685-1750), autore del *Clavicembalo ben temperato*, una pietra miliare della musica, ancora oggi utilizzata per la didattica nei Conservatori. Nel Clavicembalo ben temperato Bach ha incluso preludi e fughe in tutte le possibili tonalità, maggiori e minori. Il nome dell'opera, deciso da Bach, fa riferimento al temperamento buono in uso in quell'epoca; la cosa probabilmente indica l'approvazione di Bach per questo temperamento. Anche se il temperamento buono non è proprio quello equabile, andava nella direzione di rendere tutte le tonalità equivalenti.

I moderni sistemi di accordatura rendono facile accordare gli strumenti secondo il temperamento equabile che è quindi diventano lo standard. Per concludere, la Figura 2.8 mostra le frequenze delle note, relativamente a una singola ottava, che si ottengono con il temperamento equabile.

Le frequenze delle 12 note sono state espresse nella forma compatta $f_k = (\sqrt[12]{2})^k f$. La scala è ovviamente sempre costituita dalle sole 7 note `Do`, `Re`, `Mi`, `Fa`, `Sol`, `La` e `Si`: Come si può vedere in questa scala esistono solo due tipi di intervalli fra una nota e quella successiva: l'intervallo corrispondente a $\left(\sqrt[12]{2}\right)^2$, che è un tono, e l'intervallo corrispondente a $\sqrt[12]{2}$ che è un semitono. Un semitono è esattamente la metà di un tono. Se si prova adesso a costruire la stessa scala a partire dalla nota `Re`, o da una qualsiasi altra nota, non sarà necessario introdurre nuove note.

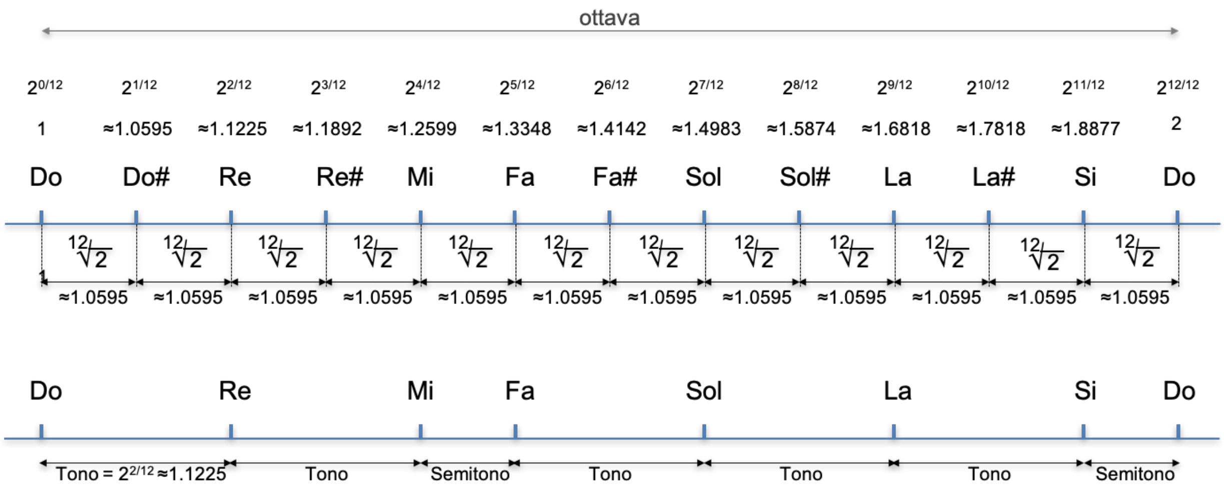


Figura 2.8: Temperamento equabile

Esercizi

1. Se un corpo vibra a una frequenza f, Quali frequenze hanno i suoni armonici prodotti dalla vibrazione?
2. Come si sommano due intervalli di frequenze contigui?
3. Come si costruisce la scala pitagorica a partire da una frequenza f?
4. Un semitono pitagorico è la metà di un tono pitagorico?
5. Come si costruisce la scala naturale a partire da una frequenza f?
6. Quali sono gli intervalli di tono e semitono nella scala naturale?
7. Il semitono della scala naturale, $\frac{16}{15}$, è la meta del tono naturale grande, $\frac{9}{8}$, o del tono naturale piccolo, $\frac{10}{9}$?
8. Come si costruisce la scala delle quinte a partire da una frequenza f?
9. Quale è il problema delle scale pitagoriche, naturali e delle quinte che viene risolto con il temperamento equabile?
10. Quanto vale un semitono nel temperamento equabile? Si può esprimere come frazione?
11. Costruisci un sistema musicale equabile, basato sull'assioma dell'ottava, ma con 10 note (al posto di 12).
12. Costruisci un sistema musicale equabile, cambiando l'assioma di partenza: non più il rapporto $\frac{2}{1}$, ma $\frac{3}{1}$. Usa 20 note.
13. Scrivere un programma che partendo dalla scala pitagorica costruisce tutte le altre scale partendo dalle frequenze generate dalle note della scala pitagorica e da tutte quelle che vengono generate di conseguenza. Quando si ferma il programma?
14. Ripetere l'esercizio 13 partendo dalla scala naturale.

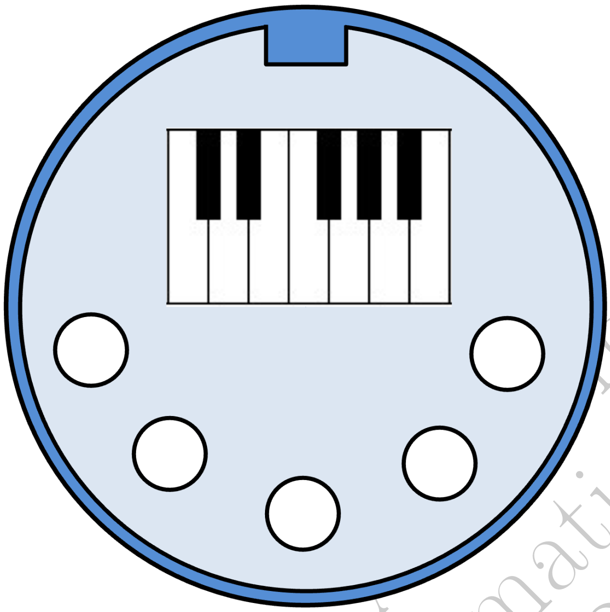
15. Scrivere un programma che genera le note usando l'intervallo di quinta, cioè 3/2. Quando si ferma il programma?

16. Ripetere l'esercizio 15 usando l'intervallo di quarta, cioè 3/2. Quando si ferma il programma?

17. La scala cromatica delle quinte costruisce le 12 note sfruttando esclusivamente l'intervallo di una quinta (3/2). È possibile fare un procedimento simile con l'intervallo di quarta? Motivare e discutere la risposta.

Capitolo 3

MIDI: Standard file e protocollo



In questo capitolo parleremo di Standard MIDI file e del protocollo MIDI. L'acronimo MIDI significa "Musical Instrument Digital Interface". Agli inizi degli anni '80 gli strumenti digitali erano ormai d'uso comune e nasceva l'esigenza di farli comunicare; spesso molti sintetizzatori venivano connessi fra di loro e quindi c'era l'esigenza di coordinarne l'esecuzione. Lo standard MIDI nasce proprio per questo motivo e le maggiori case produttrici di strumenti digitali hanno partecipato alla definizione dello standard. Nel 1984 nasce la "MIDI Manufacturers Association" che si occupa della pubblicazione delle specifiche MIDI. Lo standard MIDI è costituito da 3 componenti: l'interfaccia hardware, il protocollo, e il formato dei file.

3.1 Interfaccia MIDI

Lo standard MIDI stabilisce che il trasferimento dei dati avvenga alla velocità di 31.25 bps usando un collegamento seriale asincrono. Quindi i bit vengono spediti uno di seguito all'altro (trasmissione seriale) e il momento in cui vengono spediti viene deciso dal dispositivo che trasmette (trasmissione asincrona).

I connettori storicamente utilizzati per i dati MIDI sono connettori elettrici di tipo DIN, il cui formato è stato standardizzato dall'istituto tedesco per la standardizzazione (Deutsches Insitiut für Normung). Un cavo DIN usato per trasmettere dati MIDI non deve superare la lunghezza di 6.6 metri in quanto con cavi più lunghi si potrebbero avere problemi di trasmissione dovuti all'attenuazione del segnale elettrico e di conseguenza anche a interferenze esterne. I connettori DIN prevedono 5 pin, di cui solo 3 vengono utilizzati per il protocollo MIDI.

Figura 3.1: Connettore e cavo MIDI

Gli strumenti digitali MIDI prevedono 3 porte fisiche: In, Out e Thru. La porta In serve a ricevere dati MIDI dall'esterno, la porta Out serve a spedire verso l'esterno i dati MIDI prodotti dallo strumento digitale, mentre la porta Thru replica in uscita i dati ricevuti sulla porta In senza l'intervento dello strumento stesso. Attraverso la porta Thru è possibile connettere più strumenti MIDI in cascata.

La figura 3.1 mostra le porte In, Out e Thru di un'interfaccia MIDI, un cavo MIDI maschio-maschio e i dettagli del connettore: i PIN 1 e 3 non sono utilizzati, il 4 porta la corrente, il 2 è collegato alla massa mentre il 5 trasporta i dati MIDI.

Il collegamento con cavi MIDI DIN, tuttavia, non viene più utilizzato in quanto i moderni computer riescono a comunicare molto velocemente con interfacce e cablaggi più recenti, come, ad esempio, lo standard USB: I dati MIDI vengono spediti tramite il cavo USB.

3.2 Protocollo MIDI

Come un qualsiasi altro protocollo, il protocollo MIDI è un insieme di regole che permette la comunicazione fra due strumenti MIDI. Le regole del protocollo sono ovviamente specifiche per gli strumenti digitali e riguardano la nota da suonare, lo strumento da usare e tante altre informazioni riguardanti la riproduzione elettronica del suono.

Gli strumenti MIDI sono sintetizzatori elettronici capaci di produrre suoni. Il protocollo MIDI nasce dalla necessità di far comunicare sintetizzatori elettronici commercializzati da diverse case produttrici. Quindi il protocollo MIDI si pone come linguaggio universale per i sintetizzatori elettronici.

Quali sono le informazioni di cui ha bisogno un sintetizzatore per produrre i suoni? Nel seguito daremo una descrizione di tali informazioni

3.2.1 Le note

Il protocollo MIDI rappresenta le note con i numeri interi da 0 a 127. Quindi in totale possiamo rappresentare 128 note diverse. Considerando che un pianoforte ha 88 tasti e quindi 88 note ed è lo strumento con l'estensione più grande, i 128 codici MIDI per le note sono più che sufficienti. La tabella 3.3 riporta i codici MIDI e le note corrispondenti. È uso comune indicare l'ottava di riferimento di una nota usando un pedice; ad esempio $\mathtt{C}_1$ o $\mathtt{Do}_1$ denota il `Do` della prima ottava del pianoforte, $\mathtt{C}_2$ o $\mathtt{Do}_2$ il `Do` della seconda ottava e così via. Il $\mathtt{Do}_4$ che corrisponde al codice MIDI 60 è il `Do` centrale del pianoforte. Poichè la rappresentazione MIDI permette di distinguere 128 note, andiamo al di fuori dell'estensione di un pianoforte che tipicamente va dalla nota $\mathtt{A}_0$, codice MIDI 21, alla nota $\mathtt{C}_8$, codice MIDI 108.

Ottava	Codici MIDI											
	C	C#	D	D#	C	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	82	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				

Tabella 3.1: Codici MIDI. I tasti del pianoforte corrispondono ai codici dal 21 al 108.

3.2.2 Valore di attacco e di rilascio

Il codice MIDI di una nota individua la nota, cioè la frequenza del suono. In funzione dello strumento scelto oltre a specificare quale nota suonare, che, ad esempio, per un pianoforte significherebbe quale tasto premere mentre per un violino significherebbe in quale punto di quale corda applicare la pressione, è necessario specificare altri parametri che influenzano il suono prodotto. Ad esempio, se si preme un tasto di un pianoforte il suono sarà più forte e corposo se il tasto è premuto con forza e rapidità, mentre sarà più debole e raccolto se il tasto viene premuto con delicatezza. Sul violino dipende da come si usa l'arco, sul flauto da come si soffia. Per specificare il modo in cui si "attacca" il suono, il protocollo MIDI prevede un parametro che specifica la velocità di attacco. Anche in questo caso i valori possibili sono da 0 a 127.

Con funzione analoga (simmetrica) c'è il valore di rilascio che indica in che modo viene "terminata" una nota. Per alcuni strumenti, come ad esmepio il pianoforte, tale valore non ha nessun effetto. Per altri, come ad esempio un flauto, determina il modo in cui viene terminata una nota: un valore alto indica una terminazione brusca, un valore basso indica una terminazione più graduale. La reale implementazione spetta alla periferica MIDI che riceve il messaggio.

3.2.3 Strumenti

I sintetizzatori posso produrre suoni di vario tipo "imitando" il timbro degli strumenti tradizionali. Ad esempio, possiamo suonare delle note MIDI con il timbro di un flauto o di un violino. Anche in questo caso il protocollo MIDI prevede 128 possibilità dando di fatto la possibilità di utilizzare contemporaneamente fino a 128 timbri diversi. La Tabella 3.2 riporta alcuni codici MIDI per i vari timbri. Nel gergo MIDI si utilizza la parola inglese *patch* per indicare un particolare timbro e l'espressione *program change* per indicare un cambiamento di timbro (quindi, in questa accezione *program* è sinonimo di *patch*).

3.2.4 Canali

Le informazioni MIDI vengono raggruppate in canali. Il protocollo MIDI prevede l'utilizzo di 16 canali. I canali costituiscono un mezzo per separare informazioni diverse fra loro da un punto di vista logico. Ad esempio, a ogni canale può essere associato un particolare strumento. Tutti i comandi che producono un suono relativo a un particolare canale utilizzeranno come timbro quello

Codice	Descrizione (inglese)	Descrizione (italiano)
1	Acoustic Grand	Pianoforte classico
3	Electric Grand	Pianoforte elettrico
7	Harpischord	Clavicembalo
19	Rock organ	Organo Rock
20	Church organ	Organo da chiesa (a canne)
23	Harmonica	Armonica
24	Accordion	Fisarmonica
33	Acustic bass	Basso acustico
34	Electric bass	Basso elettrico
41	Violin	Violino
42	Viola	Viola
43	Cello	Violoncello
44	Contrabass	Contrabbasso
49	String Ensemble 1	Sezione archi 1
50	String Ensemble 2	Sezione archi 2
53	Choir Aahs	Coro (aah)
54	Choir Oohs	Coro (ooh)
67	Tenor Sax	Sassofono tenore
72	Clarinet	Clarinetto
74	Flute	Flauto

Tabella 3.2: Alcuni timbri MIDI

associato al canale. Dunque di fatto il numero di timbri diversi utilizzabili contemporaneamente si riduce a 16 in quanto tanti sono i canali. I canali possono essere usati anche per fare in modo che alcuni strumenti MIDI rispondano solo a messaggi inviati su determinati canali.

3.2.5 Suonare una nota

Per suonare una nota occorre spedire a uno strumento MIDI un messaggio che indica allo strumento di suonare la nota e successivamente spedire un altro messaggio che indica allo strumento di smettere di suonare quella nota. Per un pianoforte questi due messaggi corrispondono alla pressione del particolare tasto che produce la nota e al rilascio del tasto precedentemente premuto. Parleremo più in dettaglio di questi e di altri messaggi MIDI nel seguito. Per il momento ci limitiamo a dire che esiste un messaggio di *Note On* che serve a far suonare la nota. Nel messaggio, oltre al canale di appartenenza, deve essere specificato il codice MIDI della nota da suonare e la velocità di attacco. In modo simile il messaggio di *Note Off* serve a dire allo strumento che lo riceve di smettere di suonare una particolare nota. Come il messaggio di Note On, anche per quello di Note Off occorre specificare il canale MIDI, la nota da non suonare più e la velocità di rilascio.

3.2.6 Informazioni sul tempo

Il protocollo MIDI è stato progettato per l'utilizzo in tempo reale. I messaggi relativi alle note non hanno una indicazione riguardo a "quando" l'azione deve essere eseguita. Il motivo per cui questa indicazione manca è che non serve: l'azione viene eseguita nel momento in cui arriva il messaggio.

Tuttavia alcune periferiche MIDI che registrano i dati MIDI, come ad esempio un sequencer, hanno bisogno di informazioni sul tempo per poter riprodurre successivamente le note. Affinchè tali periferiche possano eseguire la musica MIDI con il giusto tempo il protocollo MIDI prevede dei messaggi di Time Clock per la sincronizzazione. Una periferica MIDI master spedisce a intervalli regolari dei messaggi di Time Clock. Vengono spediti 24 messaggi di Time Clock per ogni semiminima. La durata di ogni semiminima dipende dal BPM (Beat Per Minute). Una semiminima corrisponde a un beat. Ad esempio, con un BPM=120 abbiamo 120 semiminime in un minuto, quindi ogni semiminima dura 0.5 secondi, o equivalentemente in ogni secondo ci sono 2 semiminime. Pertanto in questo caso il master MIDI spedisce 48 messaggi di Time Clock in ogni secondo. Le periferiche MIDI che ricevono questi messaggi possono sincronizzarsi per poter memorizzare i messaggi MIDI e successivamente suonare alla velocità corretta.

Per capire facciamo un esempio: consideriamo il semplice caso di due messaggi MIDI uno di Note On e l'altro di Note Off per la stessa nota.

Se la periferica MIDI che riceve tali messaggi è un sintetizzatore di suoni, deve semplicemente eseguire i comandi e li eseguirà nel momento in cui i messaggi arrivano. Assumiamo che il messaggio di Note On arrivi al tempo 0 sec e che il messaggio di Note Off arrivi al tempo 1 sec. La nota verrà suonata per 1 secondo dal tempo 0 sec al tempo 1 sec. Abbiamo completamente ignorato i messaggi di Time Clock.

Se invece la periferica MIDI che riceve i messaggi è un sequencer che deve memorizzare i messaggi per poi poterli suonari successivamente, allora diventano fondamentali i messaggi di Time Clock. Poichè dal momento in cui riceve il messaggio di Note On al momento in cui riceve il messaggio di Note Off il sequencer riceve 48 messaggi di Time Clock (assumiamo BPM=120) il sequencer sa che quando dovrà suonare la nota dovrà farla durare 48 Time Clock. La velocità di esecuzione potrà essere variata variando la velocità di spedizione dei messaggi di Time Clock.

3.3 File MIDI

I file MIDI servono per memorizzare dei dati MIDI in modo tale da poterli riutilizzare successivamente. Ovviamente, come per i sequencer, anche per i file MIDI occorre aggiungere delle informazioni relative ai tempi di esecuzione. In questa sezione descriveremo in dettaglio il fomrato dei file MIDI.

3.3.1 Big-Endian

I valori che vengono rappresentati con più byte vengono spediti/memorizzati nel formato Big-Endian, cioè partendo dal byte più significativo e proseguendo con i byte meno significativi.

3.3.2 Lunghezza variabile

I file MIDI utilizzano per alcune informazioni dei campi a lunghezza variabile. Un valore rappresentato con un campo a lunghezza variabile può essere rappresentato da uno o più byte. In ognuno dei byte utilizzati per la rappresentazione a lunghezza variabile il bit più significativo non fa parte dei dati ma viene utilizzato come segnalatore: se tale bit è 1 allora il prossimo byte fa ancora parte del dato a lunghezza variabile, se è 0 allora la rappresentazione termina con il byte in questione. Chiaramente solo 7 bit per ogni byte della rappresentazione possono essere sfruttati per i dati. Quindi se il valore da rappresentare è compreso fra 0 e 127, basterà un solo byte in cui il bit più significativo (che normalmente vale 128, ma in questo caso è solo un bit di segnalazione) vale 0. Per rappresentare 128 non basterà un singolo byte perché abbiamo bisogno di 8 bit di dati.

Ad esempio, consideriamo la rappresentazione a lunghezza variabile fornita dai seguenti byte:

Byte	Descrizione	Valore o range	Commento
1-4	Stringa di identificazione	"MThd" (0x4d546864)	costante
5-8	Grandezza del blocco	6 (0x00000006)	costante
9-10	Formato del file	0-2	utilizza solo 2 bit
11-12	Numero di tracce	1-255	utilizza solo 1 byte
13-14	PPQN o SMPTE	PPQN: 1-4096	PPQN utilizza solo 10 bit

Tabella 3.3: Blocco di intestazione

10000001**0**0000000

Leggendo il primo byte **1**0000001 vediamo che il bit più significativo è 1, il che significa che ci sarà almeno un altro byte da considerare. Gli altri 7 bit sono parte dei dati e sono i bit più significativi. Leggendo il secondo byte **0**0000000 vediamo che il bit più significativo è 0 e qiondi questo è l'ultimo byte della rappresentazione. Gli altri 7 bit di questo byte sono parte dei dati. Pertanto la rappresentazione binaria del dato, ignorando gli 0 alla sinistra dell'1 più significativo, è

10000000

che corrisponde a 128. Con 2 byte è possibile rappresentare valori di 14 bit, quindi fino a $2^{14}-1$=16383. Per il valore successivo, 16384, occorrono 3 byte. Poiché con la rappresentazione a lunghezza variabile sfruttiamo per i dati solo 7 bit di ogni byte con k byte possiamo rappresentare i valori da 0 a $2^{7k}-1$.

Nel protocollo MIDI i dati a lunghezza variabile possono usare al massimo 4 byte. Pertanto il valore massimo rappresentabile è $2^{28}-1 = 268435455$ che in binario è rappresentato da un stringa di 28 bit pari a 1, in esadecimale da 0x0F FF FF FF, mentre nella rappresentazione a lunghezza variabile è

11111111 **1**1111111 **1**1111111 **0**1111111.

3.3.3 Struttura generale

Un file midi standard (SMF, Standard Midi File) è formato da una sequenza di blocchi, di cui il primo è un blocco di intestazione (header chunk) e quelli successivi sono dei blocchi di traccia (track-chunk). Quindi la struttura generale è:

```
SMF = <header-chunk> + <track-chunk> [+ <track-chunk> ...]
```

3.3.4 Header chunk (Blocco di intestazione)

Il blocco di intestazione (header chunk) contiene informazioni generali relative al brano musicale memorizzato nel file. Tali informazioni riguardono il formato del file MIDI, il numero di tracce e informazioni relative al tempo. Un file MIDI prevede un solo blocco di intestazione che deve essere inserito all'inizio del file.

```
header-chunk = "MThd" + <lunghezza> + <formato> + <tracce> + <tempo>
```

La struttura del blocco di intestazione è riportata nella seguente tabella.

Il blocco di intestazione (e quindi il file MIDI stesso) inizia sempre con la stringa di 4 caratteri "MThd". I successivi 4 byte contengono la lunghezza del blocco che per quello di intestazione è sempre 6; ciò significa che seguiranno altri 6 byte che fanno parte del blocco di intestazione.

I primi 2 di questi 6 byte specificano il formato del file MIDI. Esistono 3 formati identificati dai numeri 0, 1 e 2. Quindi dei due byte riservati per il formato vengono (almeno attualmente) utilizzati solo 2 bit. Nel formato 0 è prevista una sola traccia che contiene tutti gli eventi MIDI per l'intero brano musicale. Nel formato 1 ci sono due o più tracce; nella prima ci sono informazioni generali quali il titolo del brano, il tempo e altro, mentre nelle tracce successive ci sono dati relativi alle singole tracce, inclusi gli eventi MIDI relativi alle note da suonare. Il formato 2 prevede come per il formato 1 più tracce ma mentre nel formato 1 le tracce sono considerate simultanee nel formato 2 ogni traccia è indipendente dalle altre e la riproduzione della tracce non avviene necessariamente in modo simultaneo.

I successivi due byte specificano il numero di tracce. Ovviamente per i file in formato 0 tale numero è 1. Anche se sono previsti due byte, quindi si possono specificare fino a 65535 tracce, il numero massimo di tracce in un file MIDI è 255. In pratica il primo di questi due byte è sempre 0.

Gli ultimi 2 byte del blocco di intestazione specificano il numero di parti per semiminima (PPQN, Part Per Quarter Note) oppure il numero di frame per secondo. Il primo dei 16 bit indica se i successivi 15 bit specificano il PPQN (ticks).

Il PPQN specifica di quanti *tick* è fatta un semiminima. Si noti che questa indicazione temporale è relativa alla durata di una semiminima. La durata di una semiminima è stabilita dal BPM che può essere specificato attraverso un apposito meta-evento. Il BPM di default è pari a 120, e in questo caso una semiminima dura $0.5s = 500.000\mu s$. Stabilita la durata di una semiminima si può calcolare la durata di un tick dividendo la durata della semiminima per il PPQN. Ad esempio, per un BPM=120, se PPQN=100, allora ogni tick dura $5.000\mu s$.

3.3.5 Blocchi di traccia (track chunk)

Dopo il bloco di intestazione ci sono i blocchi di traccia. Un blocco di traccia contiene tutte le informazioni relative a un traccia.

```
track-chunk = "MTrk" + <lunghezza> + <evento> [+ <evento> ...]
```

Un blocco di traccia inizia con la stringa di 4 caratteri "MTrk" (0x4D54726B), seguita da altri 4 byte che specificano il numero totale di byte, a partire dal byte successivo, che fanno parte della traccia.

Gli eventi vengono usati per descrivere tutte le informazioni relative alla traccia a partire dagli eventi che descrivono le note per arrivare a tanti altri tipi di eventi (come ad esempio il nome della traccia o le parole del testo associato alla musica). Ogni evento è preceduto da un *delta time* che specifica dopo quanto tempo, rispetto all'evento precedente, l'attuale evento deve accadere. Il delta time viene specificato in ticks con un campo a lunghezza variabile. Se il delta time vale 0 vuol dire che l'evento che segue il delta time deve essere eseguito insieme all'evento precedente. Ci sono alcuni eventi per i quali il delta time non ha senso (ad esempio il nome della traccia). Poichè tutti gli eventi devono essere preceduti da un delta time il delta time esiste sempre. Per convenzione tutti gli eventi che non hanno un ordine temporale rispetto algi altri eventi vengono inseriti all'inizio del blocco di traccia ed usano un delta time pari a 0. Si rammenti che il delta time è specificato in ticks la cui durata dipende dal BPM e dal PPQN.

Il formato generale di un evento è il seguente:

```
evento = <delta-time> + <status byte> + (<dati-MIDI> | <dati-SYS> | <dati-META>)
```

Valore			Categoria	Messaggio
binario	esadecimale	decimale		
0xxx xxxx	0x00-0x7F	0-127	Running status	
1000 xxxx	0x80-0x8F	128-143	MIDI Voice message	Note Off
1001 xxxx	0x90-0x9F	144-159		Note On
1010 xxxx	0xA0-0xAF	160-175		Note Aftertouch
1011 xxxx	0xB0-0xBF	176-191		Controller
1100 xxxx	0xC0-0xCF	192-207		Program Change
1101 xxxx	0xD0-0xDF	208-223		Channel Aftertouch
1110 xxxx	0xE0-0xEF	224-239		Pitch Bend
1111 0000	0xF0	240	SYS messages	Inizio SysEx
1111 0001	0xF1	241		*undefined*
1111 0010	0xF2	242		Song position (LSB)
1111 0011	0xF3	243		Song position (MSB)
1111 0100	0xF4	244		Song select
1111 0101	0xF5	245		*undefined*
1111 0110	0xF6	246		*undefined*
1111 0111	0xF7	247		Fine SysEx
1111 1000	0xF8	248	Real Time	Timing Clock
1111 1110	0xF9	249		*undefined*
1111 1110	0xFA	250		Start
1111 1110	0xFB	251		Continue
1111 1110	0xFC	252		Stop
1111 1110	0xFD	253		*undefined*
1111 1110	0xFE	254		Active Sensing
1111 1111	0xFF	255	META evento	

Tabella 3.4: Singificato del byte di stato

Dopo il delta time c'è sempre un byte chiamato *byte di stato* (in realtà per risparmiare spazio spesso si usa un trucchetto per omettere questo byte quando il suo valore è lo stesso del precedente evento – descriveremo in seguito questo trucchetto noto con il nome di *running status*). Gli eventi sono di 3 tipi: eventi MIDI, eventi di sistema e meta eventi. Il byte di stato permette di capire il tipo di evento, come specificato Tabella 3.4.

Eventi MIDI

Gli eventi MIDI sono quelli che contengono le informazioni "musicali", come, ad esempio quelle relative all'esecuzione di una particolare nota. Un singolo evento MIDI consiste di

```
<dati-MIDI> = <parametro1> [+ <parametro2>]
```

Il primo elemento di un evento MIDI è lo status byte che specifica il tipo di messagio MIDI (con i 4 bit più significativi) e il canale (con i 4 bit meno significativi). Dopo lo status byte possono essere presenti uno o due parametri in funzione del tipo di messaggio MIDI. Gli eventi MIDI costituiscono la maggior parte di un file MIDI. Ci sono 7 tipi di eventi MIDI. La Tabella 3.5 descrive questi eventi specificando anche il significato dei 2 parametri. In alcuni casi viene usato solo il primo parametro (quando il secondo parametro non è usato esso non è presente nel file).

Descrizione	Status byte		Parametro 1	Parametro 2
	4bit (MSbits)	4bit (LSbits)	1 byte (0-127) usa solo 7 bit (LSbits)	1 byte (0-127) usa solo 7 bit (LSbits)
Note Off	0x8	canale (0-15)	codice MIDI nota	velocità
Note On	0x9	canale (0-15)	codice MIDI nota	velocità
Note Aftertouch	0xA	canale (0-15)	codice MIDI nota	volore di aftertouch
Controller	0xB	canale (0-15)	controller number	controller value
Program Change	0xC	canale (0-15)	program number	non usato
Channel Aftertouch	0xD	canale (0-15)	valore di aftertouch	non usato
Pitch Bend	0xE	canale (0-15)	pitch value (LSB)	pitch value (MSB)

Tabella 3.5: Eventi MIDI

Note Off. L'evento Note Off viene usato per rappresentare la fine di una particolare nota. Il codice MIDI della nota specifica a quale nota si riferisce l'evento, mentre il secondo parametro specifica la velocità con cui viene rilasciato la nota. In molti casi la velocità di rilascio non è molto importante (ad esempio per il pianoforte), ma in alcuni casi può essere utile. Il reale effetto dipende da come la periferica MIDI interpreta questo parametro.

Note On. L'evento Note On viene usato per rappresentare l'inizio di una particolare nota. Il codice MIDI specifica la nota da suonare mentre il secondo parametro rappresenta la velocità di attacco.

Aftertouch. Questo evento viene utilizzato per specificare un cambio di pressione relativo a una nota in esecuzione. Il primo parametro specifica la nota, il secondo la pressione (0 = nessuna pressione, 127, massima pressione).

Controller. Questo evento permette di specificare un cambiamento di stato relativo a un canale MIDI. É possibile specificare fino a 128 tipi di controller utilizzando il primo parametro. Il secondo parametro specifica il valore del controller. Un esempio di controller è il "Foot controller" che permette di specificare l'eventuale pressione di un pedale. La Tabella 3.6 mostra i controller definiti dal proocollo MIDI.

Program Change. L'evento Program Change permette di cambiare lo strumento MIDI da utilizzare per la riproduzione dei suoni. Questo evento ha bisogno solo del primo parametro che specifica lo strumento MIDI.

Channel Aftertouch. Questo evento è simile all'evento Aftertouch, con la differenza che opera su tutte le note del canale MIDI. Ha bisogno di un solo parametro che specifica la pressione da applicare a tutte le note che sono attive sul canale MIDI specificato.

Pitch Bend. Questo evento è utilizzato per specificare una eventuale modifica dell'altezza della nota. In realtà questo evento è molto simile a un controller con la differenza che il tipo di controller è intrinseco (è appunto il controller dell'altezza della nota) e il valore viene specificato con 14 bit (mentre un normale controller può usare solo 7 bit per il valore). Il valore del Pitch Bend è specificato dai 14 byte yyyyyyyxxxxxxx dove i caratteri y sono i 7 bit meno significativi del secondo parametro ed i caratteri x sono i 7 bit più significativi del primo parametro. Pertanto i possibili valori per il Pitch Bend sono 0-16383. L'evento ha effetto su tutte le note attive relative al canale MIDI specificato dall'evento. Il valore 8192 è il punto centrale che rappresenta l'altezza reale della nota. Valori al di sotto di 8192 fanno decrescere l'altezza della nota, mentre valori al di sopra fanno crescere l'altezza della nota. L'intervallo di riferimento varia in funzione dello strumento selezionato, ma tipicamente è di 1 tono sia a scendere che a salire. Quindi un Pitch

Valore	Descrizione del controller
0 (0x00)	Bank Select
1 (0x01)	Modulation
2 (0x02)	Breath Controller
4 (0x04)	Foot Controller
5 (0x05)	Portamento Time
6 (0x06)	Data Entry (MSB)
7 (0x07)	Main Volume
8 (0x08)	Balance
10 (0x0A)	Pan
11 (0x0B)	Expression Controller
12 (0x0C)	Effect Control 1
13 (0x0D)	Effect Control 2
16-19 (0x10-0x13)	General-Purpose Controllers 1-4
32-63 (0x20-0x3F)	LSB for controllers 0-31
64 (0x40)	Damper pedal (sustain)
65 (0x41)	Portamento
66 (0x42)	Sostenuto
67 (0x43)	Soft Pedal
68 (0x44)	Legato Footswitch
69 (0x45)	Hold 2
70 (0x46)	Sound Controller 1 (default: Timber Variation)
71 (0x47)	Sound Controller 2 (default: Timber/Harmonic Content)
72 (0x48)	Sound Controller 3 (default: Release Time)
73 (0x49)	Sound Controller 4 (default: Attack Time)
74-79 (0x4A-0x4F)	Sound Controller 6-10
80-83 (0x50-0x53)	General-Purpose Controllers 5-8
84 (0x54)	Portamento Control
91 (0x5B)	Effects 1 Depth (formerly External Effects Depth)
92 (0x5C)	Effects 2 Depth (formerly Tremolo Depth)
93 (0x5D)	Effects 3 Depth (formerly Chorus Depth)
94 (0x5E)	Effects 4 Depth (formerly Celeste Detune)
95 (0x5F)	Effects 5 Depth (formerly Phaser Depth)
96 (0x60)	Data Increment
97 (0x61)	Data Decrement
98 (0x62)	Non-Registered Parameter Number (LSB)
99 (0x63)	Non-Registered Parameter Number (MSB)
100 (0x64)	Registered Parameter Number (LSB)
101 (0x65)	Registered Parameter Number (MSB)
121-127 (0x79-0x7F)	Mode Messages

Tabella 3.6: MIDI Controllers

Bend di 0 farà scendere tutte le note attive di un tono, mentre un Pitch Bend di 16383 le farà salire di un tono.

Running status

Il protocollo MIDI prevede la possibilità, per gli eventi MIDI, (non per gli eventi di sistema ed i metaeventi, né tantomeno per i messaggi real-time) di non spedire il byte di stato nel caso in cui il valore di questo byte sia lo stesso di quello dell'evento precedente. Questa situazione viene segnalata da uno status byte minore o uguale a 127, cioè uno status byte con l'MSB pari a 0. Infatti i valori dello status byte che hanno significato in termini di eventi sono quelli con valori maggiori o uguali a 128, cioè con l'MSB pari a 1. Pertanto se il valore dello status byte è minore di 128 allora lo status byte è fittizio e in realtà il suo valore verrà usato come valore del primo parametro dell'evento. Il valore reale dello status byte in questo caso è uguale a quello dell'evento precedente. Quindi di fatto lo status byte c'è sempre, solo che con questo trucco riusciamo a codificare con un solo byte sia lo satus byte (che è uguale a quello dell'evento precedente) sia il valore del primo parametro.

Per rendere ancora più chiaro il trucco facciamo un esempio. Supponiamo di avere la seguente sequenza di 3 messaggi MIDI:

```
0x00 0x93 0x3C 0x7F 0x00 0x93 0x40 0x7F 0x00 0x93 0x43 0x7F
```

Questi 3 messaggi, che appaiono in sequenza nel flusso di dati midi, hanno come status byte lo stesso valore (0x90) che corrispondente al messaggio MIDI di Note On sul canale 3. Questa sequenza di 3 messaggi MIDI necessita di 9 byte. Usando il trucco del running status è possibile ridurre la lunghezza della sequenza a 7 byte, semplicemente omettendo gli status byte del secondo e del terzo messaggio. Infatti il valore del primo parametro (come pure quello del secondo) è sempre minore di 128. Pertanto la sequenza di byte

```
0x00 0x93 0x3C 0x7F 0x00 0x40 0x7F 0x00 0x43 0x7F
```

è equivalente alla precendente ma occupa solo 7 byte.

Il running status può essere utilizzato solo per i messaggi MIDI (status byte da 0x80 a 0xEF). I messaggi di tipo real time (status byte da 0xF8 a 0xFF) non hanno nessun effetto sul running status (cioè, è come se non fossero stati trasmessi). Si noti che 0xFF è in realtà il codice per i meta eventi.

I messaggi di sistema, invece, cancellano il running status, cioè dopo un messaggio di sistema è obbligatorio spedire il byte di stato.

Il trucco del Note Off

Omettere lo status byte è possibile solo quando esso è uguale a quello precedente. La maggiore parte dei messaggi MIDI è costituita messaggi di Note On e Note Off che si alternano o si ripetono in funzione della musica che codificano. Quando 2 o più messaggi dello stesso tipo (sia di Note On che di Note Off) si ripetono si può usare il running status. Esiste un trucchetto che permette di evitare completamente i messaggi di Note Off ed usare solo messaggi di Note On. Il trucchetto consiste nel sostituire i messaggi di Note Off con messaggi di Note On la cui velocità di attacco (secondo parametro) è 0. Di fatto un tale messaggio fa sì che la nota venga "spenta" in quanto viene suonata con un volume pari a 0. Usando questo trucco per tutti i messaggi di Note Off si ha che la sequenza di messaggi MIDI diventi una sequenza di sole messaggi di Note On. Questo permette di utilizzare per tutti i messaggi il running status.

Tipo	Descrizione
0x00	Sequence number
0x01	Text event
0x02	Copyright notice
0x03	Sequence or track name
0x04	Instrument name
0x05	Lyric text
0x06	Marker text
0x07	Cue point
0x20	MIDI channel prefix assignment
0x2F	End of track
0x51	Tempo setting
0x54	SMPTE offset
0x58	Time signature
0x59	Key signature
0x7F	Sequencer specific event

Tabella 3.7: Alcuni meta-eventi

Meta Eventi

I meta-eventi servono a inserire informazioni addizionali non prettamente collegate alla musica codificata dagli eventi MIDI, come ad esempio il nome della traccia. Essi hanno la sguente forma:

```
dati-META = 255 + <tipo> + <lunghezza> + [dati-evento]
```

Quindi un meta evento inizia sempre con un byte che vale 255, seguito da altri 2 byte ed eventualmente da ulteriori byte di dati, il cui numero e significato dipende dallo specifico meta-evento. Il secondo byte specifica il tipo di meta evento. La lunghezza dell'evento è un campo a dimensione variabile e specifica quanti byte fanno parte dell'evento a partire da quello successivo. Se il meta-evento non prevede dati la lunghezza è 0.

La Tabella 3.7 riporta alcuni meta eventi. A titolo di esempio descriviamo in dettaglio alcuni di questi meta-eventi.

Text. I meta-eventi 0x01-0x07, hanno tutti la stessa struttura in quanto permettono di inserire informazioni testuali. La struttura è `FF c n` $\mathtt{b}_1$ $\mathtt{b}_2$ ... $\mathtt{b}_n$, dove `c` è il codice del meta-evento — ad esempio 5 per il testo della canzone — `n` è il numero di caratteri di cui è costituito il testo ed i successivi `n` byte sono i caratteri del testo in codifica ASCII, anche se tutti i valori da 0 a 255 sono ammessi. La differenza è solo la categorizzazione del tipo di informazione contenuta nel testo. Il meta evento Text Event è per un testo generico (es. dei commenti). Il meta evento Copyright notice è per le informazioni sui diritti d'autore. Il meta evento Sequence or track name è per specificare un nome della traccia. Il meta evento Instrument name è per specificare il nome dello strumento utilizzato. Il meta evento Lyrics è per le parole della canzone. Il meta evento Marker text è per indicare delle specifiche posizioni (es. l'inizio di un verso o del ritornello).

Il meta-evento MIDI Channel Prefix serve a specificare un canale MIDI solitamente in riferimento ai successivi meta-eventi; ad esempio prima di un Instrument Name Event in modo da associare lil nome dello strumento al canale MIDI. Il formato è `255 (0xFF) 32 (0x20) 1 c`, dove `c` è un valore da 0 a 15.

Il meta event End of track serve a segnalare la fine di una traccia MIDI. Tale evento deve sempre essere l'ultimo evento di una traccia MIDI. Il formato è `255 47 0`.

Tempo (Set Tempo). Questo meta evento permette di cambiare il tempo specificando un nuovo valore del BPM. Il BPM viene specificato indicando il numero di microsecondi per semiminima. Quindi se si vuole un BPM di 120 il numero di microsecondi per semiminima è 500.000 Questo meta evento prevede 3 byte di dati. Il formato è `255 51 3` $\mathtt{b}_1$ $\mathtt{b}_2$ $\mathtt{b}_3$, con b_1 b_2 b_3 che rappresentano il numero di microsencodi. Ad esempio, il meta-evento `FF 51 03 07 A1 20` specifica un BPM di 120 in quanto `07 A1 20` codificano il valore 500.000. La specifica del valore del BPM viene di solito messa nel primo blocco di traccia, spesso allineata temporalmente con un messaggio di MIDI clock. Se non viene specificato si assume un valore standard di 500.000μs (120 BPM).

Time Signature. Questo meta evento specifica la metrica del brano. Il formato è `FF 88 4` $\mathtt{b}_1$ $\mathtt{b}_2$ $\mathtt{b}_3$ $\mathtt{b}_4$. Il primo byte b_1 è il "numeratore", il secondo byte b_2 rappresenta il "denumeratore", il terzo b_3 le pulsazioni del metronomo e, infine, il quarto B_4 un numero di biscrome. Il numeratore viene specificato in modo diretto (valore letterale), mentre il denominatore si ottiene elevando 2 a b_2 e indica la durata: ad esempio, se b_2=1, il denominatore (che vale 2) è una minima, se b_2, il denominatore (che vale 4) indica una semiminima. Le pulsazioni b_3 specificano il numero di messaggi MIDI clock per clik (il protocollo MIDI prevede 24 click per semiminima). Infine b_4 specifica l'unità di tempo in termini di numero di biscrome; solitamente questo valore è 8 perchè l'unità di tempo è spesso la semiminima che equivale a 8 biscrome.

Key Signature. Questo meta evento specifica la tonalità. Il formato è `255 89 2` $\mathtt{b}_1$ $\mathtt{b}_2$ con b_1 che specifica il numero di alterazioni (da -7 a +7) e b_2 che specifica il modo (0 maggiore, 1 minore).

3.4 References

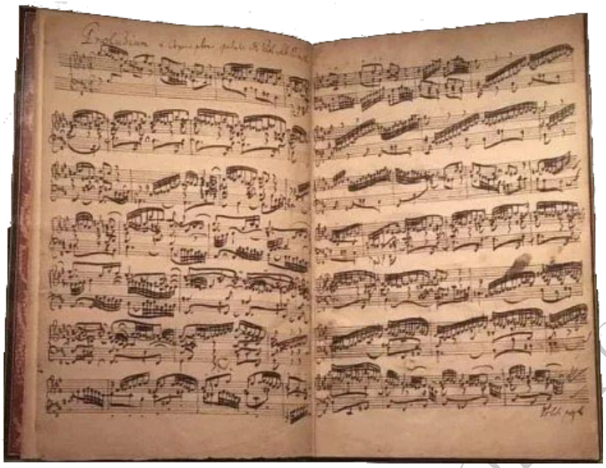
La documentazione ufficiale dello standard MIDI può essere consultata, previa registrazione gratuita, nel sito `https://www.midi.org/specifications`.

Esercizi

1. Come vengono rappresentate le note con il protocollo MIDI? E i diversi timbri? Quante note e quanti timbri diversi possono essere codificati?
2. Quali sono i messaggi MIDI che permettono di iniziare a suonare una nota e di terminare il suono?
3. Si consideri il seguente valore binario in notazione a lunghezza variabile: 10000010 (primo byte) 000000010 (secondo byte). Quale valore (decimale) codifica?
4. Con quali byte inizia un file MIDI? E con quali byte inizia una traccia all'interno di un file MIDI?
5. Che cosa è il delta time in un file MIDI? In che unità viene specificato?
6. Che cosa è lo status byte nel protocollo MIDI? Si fornisca una spiegazione dettagliata.
7. Quanti parametri prevedono i vari messaggi MIDI. Si faccia qualche esempio.
8. Si spieghi come funziona il metodo detto *running status*.
9. In cosa cosniste il "trucco" del Note Off per i file MIDI?
10. Si faccia qualche esempio di meta-evento MIDI fornendo qualche dettaglio.

Capitolo 4

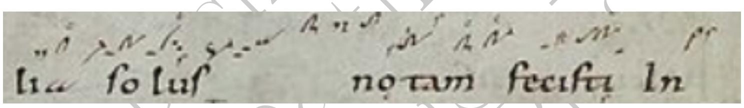
Rappresentazione e analisi della musica



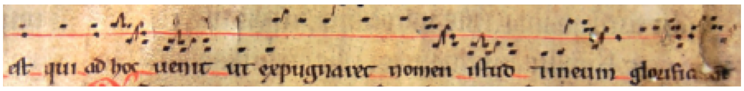
Per *rappresentazione* della musica intendiamo i modi in cui la musica può essere descritta. Ovviamente la musica, nel senso più stretto del termine, è una entità sonora che percepiamo attraverso l'udito. La rappresentazione più classica è quella della partitura (spartito musicale) che è una rappresentazione simbolica risultato di una lunga evoluzione storica. Ma non è l'unica. In particolare le moderne tecnologie permettono di memorizzare la musica in formati digitali che possono spaziare su vari piani, dalla rappresentazione MIDI a quella audio. In questo capitolo parleremo di questi argomenti e di strumenti che ci permettono di analizzare vari aspetti delle composizioni musicali.

4.1 Partitura

La rappresentazione simbolica della musica è fatta, come dice il nome, per mezzo di simboli. I *neumi* sono i primi esempi di rapprsentazione simbolica; sono segni che venivano posti sopra i testi dei canti ecclesiastici, a partire dal IX secolo circa, per dare indicazioni su come il canto doveva essere eseguito. L'evoluzione storica dei nuemi è passata attraverso varie fasi. Dai segni sopra le parole



si è passati all'utilizzo dei segni su un rigo,



poi su 4

e infine su 5 per arrivare all'attuale pentagramma, mostrato nella figura seguente; i neumi stessi hanno subito una notevole evoluzione passando dai segni alla notazione quadrata fino ad arrivare alla moderna rappresentazione tonda[1].

La rappresentazione tramite partitura è specifica per lo studio della musica da parte dei musicisti esecutori. Tuttavia per altri scopi non è la più comoda; è possibile immaginare svariate altre rappresentazioni costruite tramite elementi facilmenti rappresentabili in formato digitale e efficientemente manipolabili da un computer. Per formato digitale non intendiamo una semplice rappresentazione digitale dell'immagine di una partitura ma di formati particolari che permettano di implementare agevolmente delle specifiche operazioni. Al formato MIDI abbiamo già dedicato un intero capitolo. In questo capitolo approfondiremo il formato MusicXML e la libreria Python music21.

4.2 MusicXML

MusicXML è la specializzazione del formato XML (eXtended Markup Language) alle partiture musicali. Il suo uso principale è quello di creare un formato che permetta di scambiare facilmente spartiti musicali. Il formato MusicXML permette di descrivere tutti gli elementi di una partitura in modo tale da poter ricreare la partitura stessa. Rispetto alla partitura standard ha il vantaggio di poter essere facilmente manipolata tramite software.

In questa sezione forniremo una breve descrizione. Una trattazione completa e dettagliata del formato MusicXML può essere trovata visitando il seguente sito web:

`https://www.w3.org/2021/06/musicxml40/tutorial/introduction/`

4.2.1 XML

Lo standard XML (eXtensible Markup Language) è un linguaggio detto di "marcatura" che serve a descrivere dati strutturati. La struttura dei dati è descritta tramite cosiddetti *tag* (etichette) che vanno aperti e (obbligatoriamente) chiusi con la seguente sintassi:

```
<NOMETAG> ......... </NOMETAG>
```

[1]Per chi vuole approfondire il seguente link porta a un videocorso di storia della musica di Elia Bertolazzi, noto sul web come Mastro Elia. IN particolare gli episodi 22 e 33 parlano della notazione musicale. https://www.youtube.com/playlist?list=PLpONvz1ts9_4bN-nnPTDiq7Sz1sQMDidb .

Un tag può contenere altri tag al suo interno creando così una struttura gerarchica ad albero. Un tag aperto all'interno di un altro tag deve essere chiuso all'interno del tag in cui è stato aperto (altrimenti non si avrebbe la struttura ad albero).

Un tag può opzionalmente avere degli attributi che specificano informazioni addizionali. Gli attributi vengono specificati all'apertura del tag.

```
<NOMETAG attributo1="valore1" attributo2="valore2" ...>
```

Come si può notare anche da queste poche nozioni lo standard XML è molto generale. Un DTD (Document Type Definition) serve a definire la struttura di un documento XML tramite delle regole sintattiche: specifica quali sono gli elementi (tag), quale è la relazione gerarchica fra di essi, in che ordine devono apparire e quali elementi e quali attributi sono opzionali o obbligatori. Gli XSD (XML Schema Definition) sono una evoluzione dei DTD.

Un documento XML inizia sempre specificando, tramite un link web, lo schema (XSD) o il tipo (DTD) utilizzato. Le definizioni degli schemi e dei tipi sono forniti pubblicamente dal W3C tramite github, all'indirizzo

`https://github.com/w3c/musicxml/releases/tag/v4.0`

Il file compresso contiene sia i DTD che gli XSD. Si noti che dalla versione 4.0 i DTD sono obsoleti e quindi è necessairo usare gli XSD.

4.2.2 "*Ciao mondo*" in Music XML

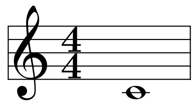
Quando si approccia un nuovo linguaggio di programmazione solitamente si parte da un esempio di programma che è il più semplice possibile e che serve solo a fornire in output un saluto ("*Ciao mondo*"). In modo analogo possiamo iniziare la descrizione del formato MusicXML dalla codifica più semplice possibile: una sola battuta con una sola nota. La Figura 4.1 mostra tale rappresentazione.

Come tutti i file XML inizia con una linea che specifica la versione di XML usata e il tipo di codifica per i caratteri. La seconda linea specifica il DTD.

La vera codifica inizia alla terza linea con `<score-partwise version="4.0">` che specifica l'elemento radice del file XML che in questo caso è uno `<score-partwise>` e anche la versione di MusicXML che viene utilizzata. Oltre a `<score-partwise>` esiste anche l'elemento `<score-timewise>`. Il file XML inizia con una lista, `<part-list>`, che elenca le varie parti componenti la partitura. In questo esempio la lista è quella minima possible: contiene un solo elemento `<score-part>` di cui viene specificato il nome con l'elemento `<part-name>`. Ogni parte deve avere un identificativo univoco che è una stringa che inizia con una lettera. Una scelta molto comune è quella di far iniziare l'identificativo con "P" e poi usare un intero progressivo. In questo esempio abbiamo una sola parte.

L'elemento `<attributes>` permette di speficare vari parametri relativi alla parte in cui è incluso. In particolare permette di specificare la chiave, l'armatura, il tempo e la granularità che si vuole raggiungere nella rappresentazione. L'elemento `<divisions>` indica la granularità: specifica l'elemento minimo in termini di divisione della seminimina. In questo caso non ha molta importanza in quanto c'è solo una semibreve quindi non scendiamo mai al di sotto di una semiminima, pertanto il valore specificato è 1. L'elemento `<key>` specifica l'armatura, cioè le alterazioni in chiahe indicando il numero di bemolle (<0) o di diesis (>0) da utilizzare con l'attributo `<fifth>` mentre l'elemento `<clef>` permette di specificare la chiave, in questo caso una chiave di violino (`sign=G`) sul secondo rigo (`line=2`). L'elemento `<time>` specifica il tempo, in questo caso un 4/4.

Infine ci sono le note, in questo caso una sola, specificate con l'elemento `<note>`. Per le note si specifica l'altezza, indicando la nota, in questo caso C, e l'ottava, in questo caso 4, la durata,



```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE score-partwise PUBLIC
    "-//Recordare//DTD MusicXML 4.0 Partwise//EN"
    "http://www.musicxml.org/dtds/partwise.dtd">
<score-partwise version="4.0">
  <part-list>
    <score-part id="P1">
      <part-name>Music</part-name>
    </score-part>
  </part-list>
  <part id="P1">
    <measure number="1">
      <attributes>
        <divisions>1</divisions>
        <key>
          <fifths>0</fifths>
        </key>
        <time>
          <beats>4</beats>
          <beat-type>4</beat-type>
        </time>
        <clef>
          <sign>G</sign>
          <line>2</line>
        </clef>
      </attributes>
      <note>
        <pitch>
          <step>C</step>
          <octave>4</octave>
        </pitch>
        <duration>4</duration>
        <type>whole</type>
      </note>
    </measure>
  </part>
</score-partwise>
```

Figura 4.1: "*Ciao mondo*" in Music XML

come multiplo dell'elemento unitario, in questo caso 4 in quanto come elemento unitario abbiamo usato la seminimima visto che `<divisions>` vale 1. La durata viene anche specificata in modo esplicito con l'attributo `<type>`, anche se può essere derivata dall'attributo `<duration>`.

4.2.3 partwise vs score-wise

I dati di un file XML sono organizzati in una struttura gerarchica. Una partitura musicale ha una struttura che può essere vista "verticalmente", se pensiamo ad una singola battuta per tutti i righi musicali, oppure "orizzontalmente" se pensiamo ad ogni rigo musicale composto di tante battute. Per questo motivo esistono 2 diversi schemi: `<score-partwise>` e `<score-timewise>`. Il primo dà priorità all'organizzazione gerarchica vista orizzontalmente, cioè le battute musicali sono contenute nella parte, mentre il secondo dà priorità all'organizzazione gerarchica vista verticalmente, cioà le parti, i singoli righi musicali, sono contenuti nella "battuta". In funzione della specifica applicazione un approccio può essere preferibile all'altro. Le due possibilità sono equivalenti ed è ovviamente possibile convertire da un formato all'altro.

La struttura generale di un file XML è, nel caso di un'organizzazione basata sulle parti, la seguente

```
<!ELEMENT score-partwise (%score-header;, part+)>
<!ELEMENT part (measure+)>
<!ELEMENT measure (%music-data;)>
```

mentre, nel caso di un'organizzazione basata sul tempo, è:

```
<!ELEMENT score-timewise (%score-header;, measure+)>
<!ELEMENT measure (part+)>
<!ELEMENT part (%music-data;)>
```

Come si può notare l'unica differenza è che nel primo caso l'elemento "part" contiene le battute, mentre nel secondo caso l'elemento "measure" (battuta) contiene la parti. In entrambi i casi l'elemento di livello più interno contiene i dati musicali veri e propri, definiti come:

```
<!ENTITY % music-data
"(note | backup | forward | direction | attributes |
 harmony | figured-bass | print | sound | barline |
 grouping | link | bookmark)*">
```

Inoltre c'è una sezione di intestazione che serve a specificare informazioni generali:

```
<!ENTITY % score-header
"(work?, movement-number?, movement-title?,
 identification?, defaults?, credit*, part-list)">
```

4.2.4 Un esempio più complesso

`https://www.w3.org/2021/06/musicxml40/tutorial/notation-basics/`

4.2.5 Documentazione Consorzio W3

`https://www.w3.org/2021/06/musicxml40/`

4.3 Libreria Python: music21

Music21 (`web.mit.edu/music21`) è una libreria Python che offre strumenti per la manipolazione della musica in formati simbolici. Oltre ad altre funzioni permette di gestire file in formato MusicXML. Music21 utilizza un approccio ad oggetti per gestire i dati musicali. Gli oggetti sono organizzati in moduli; i nomi dei moduli iniziano con una lettera minuscola mentre i nomi degli oggetti iniziano con una lettera maiuscola. Ad esempio il modulo `note` contiene l'oggetto `Note` che, come suggerisce il nome, rappresenta una nota musicale e quindi possiamo considerarlo come l'oggetto di base con il quale si costruiscono oggetti più complessi fino a rappresentare intere partiture. Il modulo `node` contiene vari altri oggetti basilari utili per rappresentare tutti gli elementi di una partitura, come, ad esempio `Lyric, Rest` o anche `Duration, Volume` che in realtà sono parte di sottomoduli (risp. `duration` e `volume`).

Nel seguito descriveremo brevemente le nozioni di base di Music21. Si faccia riferimento alla documentazione ufficiale, `web.mit.edu/music21/doc/`, per informazioni più dettagliate.

4.3.1 Oggetti `Note`, `Pitch`, `Duration`

Per creare un oggetto `Note` usiamo il costruttore, `note.Note("A4")`, e ovviamente possiamo salvare l'oggetto in una variabile: `a = note.Note("A4")`. Quindi possiamo accedere agli attributi dell'oggetto, ad esempio, `a.name` restituirà `A`, mentre `a.octave` restituirà `4`. In realtà entrambe queste informazioni sono memorizzate nell'oggetto `pitch.Pitch`, cioè l'oggetto `Pitch` del modulo `pitch`, al quale possiamo accedere con `a.pitch`. Nell'oggetto `Pitch` ci sono anche altre informazioni, come ad esempio la frequenza: `a.pitch.frequency` restituirà `440`. Ovviamente i singoli attributi possono anche essere cambiati dopo la creazione della nota. Ad esempio `a.note.octave=3` cambierà l'ottava della nota; di conseguenza la frequenza verrà cambiata da `440` a `220`. Analogamente, `a.pitch.frequency=329.6275569` cambierà la frequenza della nota e di conseguenza il "nome" simbolico, in `E4`. L'attributo `pitchClass` vale 0 per il Do, 1 per il Do$\sharp$ e così via fino a 11 per il Si. L'attributo `midi` invece fornisce il codice MIDI della nota. Per `a = note.Note("A4")` avremo `a.pitch.midi` pari a 69. L'oggetto `duration.Duration` memoriza la durata della nota. Le durate vengono specificate in multipli o sottomultipli (in entrmabi i casi anche non interi) di semiminime (cioè note di 1/4). Per le note dalla croma in su, si possono utilizzare i nomi ("maxima", "longa", "whole", "half", "quarter", "eight") mentre dalla semicroma in giù si usano nomi con in numeri ("16th", "32nd", "64th", ... fino addirittura a "2048th"). Oltre ai nomi si può usare il fattore moltiplicativo. Ad esempio `duration.Duration(1.5)` crea una durata corrispondente a un semiminima con il punto. Chiaramente usando l'opportuno moltiplicatore si possono rappresentare durate arbitrarie. L'attributo `quarterLength` fornisce la durata in termini di numero di semiminime (multipli non interi o frazioni). L'attributo `dots` specifica i "punti" musicali che allungano la durata della metà. Un oggetto `Note` di fatto è costituito da un oggetto `Pitch` che specifica l'altezza della nota e da un oggetto `Duration` che specifica la durata.

4.3.2 Liste e flussi di note

Le note possono essere raggruppate in liste e flussi per creare degli spartiti. Una lista di note è sostanzialmente un array con gli oggetti <music21.note.Note>. La gestione della lista ovviamente può essere fatta con gli strumenti che Python mette a disposizione. Ad esempio, possiamo iterare su tutte le note di una lista con `for n in noteList:  print(n)`. I flussi (stream) di note sono contenitore di oggetti musicale, come note, accordi, chiave, indicazini di tempo, etc. Sono molto simili alle liste. Gli oggetti contenuti in un flusso sono normalmente organizzati in funzione del tempo: per ogni oggetto viene specificato l'offset rispetto all'inizio del flusso, in termini di quarti. Ad esempio in una battuta di 4/4 con due minime, la prima avrà offset 0.0, mentre la seconda 2.0. Diversamente da un lista, un flusso può contenere altri flussi creando così un'organizzazione gerarchica. La classe `Stream` ha varie sottoclassi come ad esempio `Measure` per le battute, `Part` per le parti. Ad esempio se abbiamo due battute in un flusso `Part`, la prima avrà offset 0.0 e la seconda 4.0, sempre considerando un tempo di 4/4. Un'altra sottoclasse è `Score`. Quindi uno spartito potrebbe essere rappresentato da un flusso `Score`, organizzato in sottoflussi di tipo `Part`, ognuno dei quali corrispondente, ad esempio, ad uno strumento, e ogni parte fatta di un flusso di `Measure`.

4.3.3 `Show`

Il metodo `show` permette di "mostrare" un oggetto musicale. Il metodo sfrutta programmi esterni per visualizzare la musica o anche suonare il midi. Ad esempio `a=note.Note("C4"); a.show()` lancerà il programma predefinito per gestire file musicxml al quale passerà un file musicxml che descrive la nota C4. Analogamente `a.show('midi')` lancerà il porgramma predefinito per la gestione di file midi al quale verrà passato un file midi con la nota C4. Ovviamente il metodo show potrà essere usato su oggetti più complessi che descrivono intere partiture musicali.

4.4 Rappresentazioni specifiche

In alcuni casi può essere comodo usare delle rappresentazioni specifiche. Ad esempio una melodia potrebbe essere rappresentata tramite una sequenza di simboli alfanumerici. In questo modo la melodia intera sarebbe rappresentata da una stringa di caratteri e questo permetterebbe la sua manipolazione attraverso algoritmi che operano su stringhe. Due stringhe potrebbero essere "confrontate" per vedere quanto sono "diverse". Questo potrebbe essere utile ad esempio per individuare plagi musicali.

Un esempio di rappresentazione specifica è la `tinyNotation` fornita da Music21. L'istruzione `s = converter.parse('tinyNotation:  4/4 C4 D4 E4 F4 G4 A4 B4 c4')` rappresenta la par-

Figura 4.2: Tema di *"O sole mio"*, in Sol

Figura 4.3: Tema di *"O sole mio"*, in La

titura di una scala ascendente di Do maggiore, fatta di 8 seminimime. In questo caso la stringa "4/4 C4 D4 E4 F4 G4 A4 B4 c4" è la rappresentazione simbolica. In questa notazione la lettera "C" (maisucola) indica il Do centrale (C4); la lettera "c" (minuscola) invece indica il Do dell'ottava superiore (C5). Il 4 dopo la C invece non indica l'ottava, bensì la durata: 4 corrisponde ad una semiminima (8 a una croma, 16 a una semicroma, etc.). Le altre ottave sono CC e CCC verso il basso e c', c", c"' verso l'alto. La stringa "4/4 CCC8 CC8 C8 c8 c'8 c"8 c"'8 c""8" rappresenta 8 crome con gli 8 Do del pianoforte.

Chiaramente la rappresentazione simbolica può essere personalizzata in funzione della specifica applicazione.

4.4.1 Rappresentazione NOTE

In questa rappresentazione una melodia è rappresentata da una stringa di metacaratteri. Ogni metacarattere è a sua volta una stringa che specifica una nota o una pausa. La stringa che rappresenta una nota composta da una o o più caratteri che ci dicono il nome della nota e eventuali alterazioni (♯ e ♭) e un numero che indica l'ottava in cui si trova. Un carattere scecifico, la lettera "*p*". rappresenta la pausa.

Come esempio, consideriamo la melodia di *O sole mio* riportata nella Figura 4.2. La sua rappresentazione NOTE è: *p* D5 C5 B4 A4 G4 G4 A4 B5 G4 F♯4 E4 p F♯4 G4 A4 F♯4 E4 E4 *p* F♯4 G4 A4 E4 D4 D4 *p* D5 C5 B4 A4 G4 G4 A4 B5 G4 F♯4 E4 p D5 B4 A4 D5 B4 A4 G4 A4 B4 A4 B4 A4 G4.

La Figura 4.3 mostra la stessa melodia trasposta nella tonalità di La. La sua rappresentazione NOTE è: *p* E5 D5 C♯5 B4 A4 A4 B4 C♯5 A4 G♯4 F♯4 p G♯4 A4 B4 G♯4 F♯4 F♯4 *p* G♯4 A4 B4 F♯4 E4 E4 *p* E5 D5 C♯5 B4 A4 A4 B4 C♯5 A4 G♯4 F♯4 p E5 C♯5 B4 E5 C♯5 B4 A4 B4 C♯5 B4 C♯5 B4 A4.

4.4.2 Rappresentazione INTERVAL

In questa rappresentazione concentriamo l'attenzione sugli intervalli misurati in semitoni. Il metacarattere inizia con il simbolo "+" o con il simbolo "-" per indicare la direzione dell'intervallo. Dopo il segno c'è il numero di semitoni. Poichè in questa rappresentazione si indica un intervallo non possiamo specificare la nota di partenza. Inoltre nel caso di una pausa l'intervallo non ha senso. Quindi la prima nota verrà ignorata, sia all'inizio del brano, sia dopo una pausa. La presenza della pausa però verrà comuqnue indicata dalla lettera p.

Con la rappresentazione INTERVAL, la melodia di *O sole mio* riportata nella Figura 4.2 corrisponde alla stringa: -2 -1 -2 -2 +0 +2 +2 -4 -1 -2 p +1 +2 -3 -2 +0 p +1 +2 -5 -2 +0 p -2 -1 -2 -2 +0 +2 +2 -4 -1 -2 p -1 -2 +5 -3 -2 -2 +2 +2 -2 +2 -2 -2.

Con questa rappresentazione anche la melodia in La, Figura 4.3, sarà descritta dalla stessa stringa che rappresenta la melodia in Sol.

4.4.3 Rappresentazione DURATION

In questa rappresentazione concentriamo l'attenzione sulla durata delle note. Utilizzeremo la frazione che corrisponde alla durata della nota. Quindi una semibreve sarà rappresentata 1/1, una minima sarà rappresentata da 1/2, una semiminima sarà rappresentata da 1/4, una croma sarà rappresentata da 1/8, e così via. Si noti come questa rappresentazione permetta di rappresentare anche le durate delle note con un punto. Ad esempio una semimina con un punto sarà rappresentata da 3/8. Inoltre in questa rappresentazione non c'è differenza fra pause e note. D'altra parte le pause sono parte itnegrante della melodia.

Con la rappresentazione DURATION, la melodia di *O sole mio* riportata nella Figura 4.2 corrisponde alla stringa: 1/8 1/8 1/8 1/8 1/4 1/4 1/8 1/8 1/8 1/8 1/4 1/4 1/8 1/8 1/8 1/8 1/8 1/8 1/4 1/8 1/8 1/8 1/8 1/8 1/8 1/4 1/8 1/8 1/8 1/8 1/4 1/4 1/8 1/8 1/8 1/8 1/4 1/4 1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/4 1/4 1/16 1/16 1/16 3/8.

Anche in questo caso il cambio di tonalità non comporta differenze nella rappresentazione.

Esercizi

1. In un file MusicXML quale è la differenza fra lo schema `partwise` e quello `timewise`?
2. Che cosa è `music21`?
3. Descrivere brevemente la rappresentazione `tinyNotation` di `music21` e fornire un esempio musicale in tale rappresentazione.

Capitolo 5

Similarità e plagio musicale

In questo capitolo ci occuperemo del problema della similarità di brani musicali. Questo aspetto è alla base di un problema più complesso che è quello del plagio musicale. La tematica del plagio musicale è assai controversa perché coinvolge aspetti economici non di poco conto. Esistono vari esempi di cause legali relativi a plagi musicali di enorme impatto mediatico ed economico. Il principale ostacolo, che ritroviamo anche in altri problemi musicali, è quello della non esistenza di una metrica formale che permetta di dire quanto due composizioni musicali siano simili. Spesso un tale giudizio è soggettivo. Data la complessità del problema, focalizzaremo la nostra attenzione solo sull'aspetto melodico; tuttavia questo aspetto è quello che viene principalmente considerato anche nelle reali cause legali. Utilizzeremo delle rappresentazioni specifiche basate su stringhe di caratteri: nella prima parte del capitolo rivedremo alcune nozioni e algoritmi relativi alla manipolazione di stringhe.

5.1 Stringhe e distanze

Una stringa è una sequenza di caratteri presi da un dato alfabeto. Ad esempio, dato l'alfabeto $\Sigma = \{a, b, c, d, e\}$, un esempio di stringa è $s = abcbddead$. La lunghezza della stringa è il numero di caratteri di cui è costituita; nell'esempio la stringa s ha lunghezza 9. Con un opportuno alfabeto è possibile usare delle stringhe per rappresentare la musica. Un confronto delle stringhe che rappresentano due melodie può rivelare eventuali similarità.

5.1.1 Edit distance

Date due stringhe di simboli la edit distance è definita come il minimo numero di operazioni di inserimento, cancellazione e sostituzione che devono essere effettuate per trasformare una delle stringhe nell'altra. Più formalmente consideriamo un alfabeto Σ, ad esempio tutte le lettere. Una stringa definita su Σ è una sequenza di simboli appartenenti a Σ, ad esempio $\alpha = torta$, oppure $\beta = parlare$.

L'operazione di cancellazione elimina un simbolo: una stringa $\alpha = \alpha_1 x \alpha_2$ viene trasformata in $\alpha = \alpha_1\alpha_2$, eliminando il simbolo x. Ad esempio $\alpha = torta$, cancellando il carattere o, diventa $\alpha = trta$. Il simbolo da cancellare può trovarsi in qualsiasi punto della stringa.

L'operazione di inserimento inserisce un simbolo: la stringa $\alpha = \alpha_1\alpha_2$ viene trasformata in $\alpha = \alpha_1 x \alpha_2$, inserendo il simbolo x. Ad esempio $\alpha = torta$ diventa $\alpha = tortar$. Il simbolo può essere inserito in qualsiasi punto della stringa.

L'operazione di sostituzione cambia un simbolo: una stringa $\alpha = \alpha_1 x \alpha_2$ viene trasformata in $\alpha = \alpha_1 y \alpha_2$, cambiando x in y. Ad esempio $\alpha = torta$ diventa $\alpha = porta$. Anche in questo caso il simbolo da cambiare può trovarsi in qualsiasi punto della stringa.

Denotando con $ed(\alpha(i), \beta(j))$ la edit distance fra i primi i caratteri di α e i primi j caratteri β si ha che se una delle due stringhe è nulla allora la distanza è pari alla lunghezza della stringa non nulla altrimenti la edit distance (minima) è data dal minimo fra

$$ed(\alpha(i), \beta(j)) = \min \begin{cases} ed(\alpha(i-1), \beta(j) + 1, & \text{(ins. o canc. in } \alpha) \\ ed(\alpha(i), \beta(j-1) + 1, & \text{(ins. o canc. in } \beta) \\ \begin{cases} ed(\alpha(i-1), \beta(j-1) & \text{se } \alpha_i = \beta_i \text{ (nessuna)} \\ ed(\alpha(i-1), \beta(j-1) + 1 & \text{se } \alpha_i \neq \beta_i \text{ (sost.)} \end{cases} \end{cases} \tag{5.1}$$

È facile implementare un algoritmo che calcola la edit distance. Inoltre si può facilmente tenere traccia di quali operazioni sono state utilizzate. Basta usare due matrici di dimensione $(n+1) \times (m+1)$ dove n e m sono le lunghezze delle stringhe. La prima matrice conterrà il valore della distanza minima mentre la seconda servirà a memorizzare quale delle tre operazioni deve essere utilizzata per ottenere la distanza minima. Le righe e le colonne corrispondono ai singoli caratteri, nell'ordine in cui compaiono nelle strighe, e quindi corrispondono alle sottostringhe, inclusa la sottostringa vuota, a partire dall'inizio delle stringhe. Ad esempio per le parole usate prima avremmo delle matrici come quelle riportate sotto:

Distanze

	ϵ	p	a	r	l	a	r	e
ϵ	0	1	2	3	4	5	6	7
t	1							
o	2							
r	3							
t	4							
a	5							

Operazioni

	ϵ	p	a	r	l	a	r	e
ϵ	-	i_p	i_a	i_r	i_l	i_a	i_r	i_e
t	i_t							
o	i_o							
r	i_r							
t	i_t							
a	i_a							

La prima colonna e la prima riga delle matrici possono essere facilmente riempite in quanto corrispondono ai casi in cui una delle due stringhe è vuota, pertanto la distanza minima è semplicemente la lunghezza della stringa non vuota e le operazioni da utilizzare sono ovviamente tutte inserimenti, se si parte dalla stringa vuota, o tutte cancellazioni, se si parte dalla stringa non vuota (o anche un misto fra cancellazioni e inserimenti). Le restanti celle delle matrici possono essere calcolate sfruttando l'equazione 5.1. Nella matrice delle distanza si memorizza il valore della distanza mentre nella matrice delle operazioni si memorizza quale dei 3 casi dell'equazione ha determinato il minimo.

L'algoritmo 1 calcola le due matrici.

Algorithm 1 Edit distance

```
public static int computeED(String m1, String m2) {
  int distance = 0;
  int len1=m1.length();
  int len2=m2.length();
  int[][] d = new int[len1+1][len2+1];
  int[][] s = new int[len1+1][len2+1];
  for (int i=0; i<=len1; i++) {
    for (int j=0; j<=len2; j++) {
      d[i][j]=0;
      s[i][j]=0;
    }
  }
  s[0][0]='-';
  for (int i=1; i<=len1; i++) {
    d[i][0]=i;
    s[i][0]='D';
  }
  for (int j=1; j<=len2; j++) {
    d[0][j]=j;
    s[0][j]='D';
  }
  int i,j;
  for (i=1; i<=len1; i++) {
    for (j=1; j<=len2; j++) {
      if (m1.charAt(i).equals(m2.charAt(j))) {
      d[i][j]=d[i-1][j-1];
      s[i][j]='E';
  } else {
      if (d[i-1][j-1] <= d[i-1][j] && d[i-1][j-1] <= d[i][j-1]) {
        d[i][j]=1+d[i-1][j-1];
        s[i][j]='S';
      }
      else if (d[i-1][j] <= d[i][j-1]) {
        d[i][j]=1+d[i-1][j];
        s[i][j]='^';
      }
      else {
        d[i][j]=1+d[i][j-1];
        s[i][j]='<';
      }
      d[i][j]=1+Math.min(d[i-1][j-1], Math.min(d[i-1][j],d[i][j-1]));
    }
  }
}
```

5.1.2 Ricerca di sottostringhe (string matching)

Un algoritmo di string matching permette di vedere se determinati pattern di caratteri sono presenti in una stringa. Più formalmente il problema consiste nel determinare le occorrenze di un pattern di lunghezza p in un testo di lunghezza n, con $p < n$. La ricerca di una stringa all'interno di un'altra stringa può essere utilizzata come strumento per individuare frammenti melodici in comune. L'algoritmo 2 è l'algoritmo più semplice che usa la forza bruta controllando la presenza della sottostringa a partire da ogni possibile posizione.

Algorithm 2 Ricerca sottostringa, forza bruta

```
stringSearchNaive(String pattern, String str) {
  int p=pattern.length();
  int n=str.length();
  for (int i = 0; i <= n - p; i++) {
    int j;
    for (j = 0; j < p; j++)
       if (text[i+j] != pattern[j]) break;
    if (j == p) return i;  // pattern found at offset i.
  }
  return n; // pattern not found.
```

La complessità è $O(nm)$ ma il for interno, nel caso in cui non ci sia un match, si interrompe al primo carattere diverso e questo succede quasi subito se le stringhe sono distribuite uniformemente. Le prestazioni sono relativamente buone in pratica. Ovviamente il caso pessimo è $\Theta(nm)$. Ad esempio se la stringa $\alpha = aaaaaaaaaaaaaaaaaaaaaaab$ e la stringa $\beta = aaaaaaaaaaaa$.............$aaaaaaaaaa$ il for interno si accorgerà del mismtach solo all'ultimo carattere e il controllo verrà ripetuto per tutte le possibili posizioni iniziali. Esistono algoritmi più efficienti. Ad esempio l'algoritmo KMP (Knuth-Morris-Pratt) che opera una breve pre-computazione sulla stringa da cercare per poi sfruttare le informazioni dedotte per rendere più efficiente la ricerca. L'algoritmo KMP ha complessità $O(m)$.

La pre-computazione sul pattern si basa sulla seguente osservazione. Se un possibile match fallisce nella posizione q-esima del pattern sappiamo che i primi $q-1$ caratteri del pattern sono uguali a quelli della stringa (a partire dalla posizione corrente). Il pattern è sempre lo stesso quindi possiamo pre-calcolare la prossima posizione da analizzare possibilmente "saltando" delle posizioni. Facciamo un esempio concreto. Supponiamo che il pattern sia "abcde" e che stiamo controllando la sua occorrenza a partire dalla posizione 10 dalla quale la stringa ha i seguenti caratteri "abcda". Il mismatch verrà scoperto solo quando si controllerà il quinto carattere del pattern. Tuttavia possiamo ora evitare di controllare a partire dalle posizioni 11,12,13 e 14 visto che nessuna di esse contiene una "a". La prossima posizione "utile" per un possibile match è la 15. Un altro esempio. Pattern "abcabe", stringa "abcabc...". Il mismatch in questo caso verrà rilevato al sesto carattere. Inutile controllare le successive due posizione: il prossimo possibile match è dopo 3 posizioni. Dovrebbe essere chiaro che "il prossimo possibile match" dipende dai caratteri del pattern. La pre-computazione serve a stabilire l'indice del prossimo possibile match. Bisogna calcolare una tabella che per ogni indice i, $i = 1, ..., len(pattern)$, ci dice l'indice $next[i]$ del prossimo possibile match. Si noti che l'indicizzazione parte da 1 e non da 0 (scelta arbitraria).

In pratica la tabella *next* contiene informazioni su come il pattern si confronti con se stesso e più precisamente su come la parte iniziale del pattern si confronti con ogni suo prefisso. Sia q il numero di caratteri per i quali c'è stato un match con il testo. Nell'esempio precedente pattern=abcabe, $q = 6$ (al sesto carattere c'è stato un mismatch). Ci interessa sapere quale è il minimo spostamento (shift) del pattern tale che un suo prefisso corrisponda ad un suffisso dei primi $q-1$ caratteri del pattern. Tale scorrimento fornisce la prima posizione utile per un possibile occorrenza del pattern. E questa informazione ci serve per tutti i possibili q.

Algorithm 3 Algoritmo KMP, pre-computazione sul pattern

```
KMP-precomputation(String pattern) {
  m=pattern.length;
  int next[m];
  next[1]=0;
  k=0;
  for (q=2; q<m; i++) {
    while ((k>0) and (pattern[k+1] <> pattern[q])) {
      k=next[k];
    }
    if (pattern[k+1] == pattern[i]) then k=k+1;
    next[q]=k;
  }
  return next[];
}
```

Quindi la domanda alla quale dobbiamo dare una risposta con la pre-computazione è:

> Dato il prefisso *pattern*$[1..q]$, quale è il più piccolo intero s, tale che per un $k < q$ si ha che $P[1..k] = T[s+q-k..s+q]$, cioè quale è il più piccolo shift del pattern tale che un suo prefisso, di lunghezza $k < q$, sia uguale ad un suffisso che finisce nella posizione q?

L'algoritmo 3 calcola questa informazione per tutti i possibili q e la memorizza nella tabella *next*.

Algorithm 4 Algoritmo KMP

```
stringSearchKMP(String pattern, String str) {
  n = str.length;
  m = pattern.length;
  next = KMP-precomputation(pattern);
  q = 0;                                    //#chars match
  for (i=1; i<n; i++) {
    while (q>0) and (pattern[q+i] <> str[i])
      q = next[q];                          //next char does not match
    if (pattern[q+1] == str[i])
      q = q+1;                              //next char match
    if (q == m)
      print "Occorrenza alla posizione i-m";
      q = next[q];                          //Look for next match
  }
}
```

L'algoritmo 4 sfrutta la tabella *next* per evitare i confronti inutili: ogni qualvolta c'è un mismatch si evita di riconfrontare i simboli che già si conoscono.

5.1.3 Ricerca approssimata

Dato il pattern si potrebbe essere interessati a cercare occorrenze approssimate. Per fare questo si sfrutta la edit distance. Il problema dello string matching approssimato può essere definito nel seguente modo. Dato un pattern $p_1, \ldots, p_m$ e un testo $t_1, \ldots, t_n$, con $m < n$, trovare la sottostringa $t_i, \ldots, t_j$ tale che la edit distance $ed((p_1, \ldots, p_m), (t_i, \ldots, t_j))$ è quella minima fra tutte le possibili sottostringhe.

Quindi un approccio semplice sarebbe quello di considerare tutte le possibili sottostringhe, cioè tutti i possibili indici i, j, con $1 \leq i < j \leq n$, che sono $O(n^2)$ e per ognuno di essi calcolare la edit distance con il pattern. Quindi la complessità totale sarebbe $O(mn^3)$.

Esistono algoritmi più efficienti.

5.2 Similarità basata sulla distanza testuale

Vediamo adesso degli algoritmi per individuare la similarità fra due (o più) melodie. Mettendo insieme una rappresentazione testuale e la edit distance si può immediatamente avere un semplice algoritmo per la similarità: la edit distance è una misura di similarità. Se cambiano poche note nella melodia anche la stringa della rappresentazione cambierà di poco e conseguentemente l'edit distance sarà piccola. Se invece due melodie sono molto diverse fra loro lo saranno anche le stringhe che le rappresentano e pertanto l'edit distance sarà più grande.

Come facciamo a valutare la "bontà" di un tale algoritmo? Procederemo con degli esempi, valutando il comportamento dell'algoritmo.

Per poter utilizzare gli algoritmi di edit distance abbiamo bisogno di rappresentare la musica con stringhe di testo: una melodia sarà specificata da una stringa. Che cosa codifichiamo? La quantità di informazioni che codifichiamo nella stringa e anche il modo in cui le codifichiamo giocano un ruolo fondamentale. Per procedere consideriamo un caso concreto. La Figura 5.1.

Figura 5.1: Melodia di Fra Martino

Consideriamo una prima, semplice rappresentazione che codifica ogni nota con due o tre simboli, una o due lettere che corrispondono al nome della nota e all'eventuale alterazione musicale (♯ o ♭) e un numero che specifica l'ottava; inoltre useremo un carattere di separazione, −, per rendere la stringa più facilmente leggibile. La melodia della Figura 5.1 è raprresentata dalla seguente stringa: "C4 D4 E4 C4 C4 D4 E4 C4 E4 F4 G4 E4 F4 G4 G4 A4 G4 F4 E4 C4 G4 A4 G4 F4 E4 C4 D4 G4 C4 D4 G4 C4".

In questo esempio semplice non ci sono alterazioni e tutte le note sono nell'ottava numero 4 (quella centrale) della tastiera del pianoforte.

Figura 5.2: Melodia simile a Fra' Martino campanaro, in Do maggiore.

La Figura 5.2 mostra una variazione della melodia in Figura 5.1. La stringa che la descrive è "C4 D4 E4 C4 C4 D4 E4 E4 F4 G4 C4 D4 E4 F4 G4 G4 A4 G4 F4 E4 C4 G4 A4 G4 F4 E4 C4 D4 G4 C4 D4 E4 D4 C4". La edit distance fra queste due stringhe è 5. Una distanza così bassa, considerando la lunghezza delle stringhe confrontate, ci dice che sostanzialmente le due melodie sono uguali; infatti le modifiche fatte per ottenere la seconda melodia, a parte la trasposizione, sono minimali.

5.2.1 Problema della trasposizione

La Figura 5.3 mostra la melodia della Figura 5.2 trasposta nella tonalità di Re. La stringa che la descrive è "D4 E4 F♯4 D4 D4 E4 F♯4 F♯4 G4 A4 D4 E4 F♯4 G4 A4 A4 B4 A4 G4 F♯4 D4 A4 B4 A4 G4 F♯4 D4 E4 A4 D4 E4 F♯4 E4 D4". La edit distance fra le due stringhe è di 24. Considerando che le lunghezze delle 2 stringhe sono 31 e 34 caratteri, una edit distance così grande indicherebbe una differenza sostanziale fra le 2 melodie. Un valore così alto è però dovuto al fatto che le due melodie hanno tonalità diverse. Abbiamo visto prima che considerando la stessa tonalità la distanza fra le due stringhe è 5.

Figura 5.3: Melodia simile a Fra' Martino campanaro, in Re maggiore.

Vediamo un altro esempio, questa volta su un un caso reale di plagio, quello che ha visto contrapposto Al Bano a Michael Jackson.

Figura 5.4: Will you be there, M. Jackson

La Figura 5.4 mostra la partitura della melodia del brano *Will you be there* di Jackson, la cui tonalità originale è Re maggiore. La stringa che rappresenta tale melodia è "D4 D4 D4 E4 D4 E4 F♯4 F♯4 F♯4 G4 F♯4 G4 A4 A4 A4 A4 B4 A4 G4 F♯4 E4"

Figura 5.5: I cigni di Balaka, Al Bano

La Figura 5.5 mostra la partitura della melodia del brano *I cigni di Balaka* di Al Bano, la cui tonalità originale è La maggiore. La stringa che rappresenta tale melodia è "A4 A4 B4 A4 B4 A4 C♯5 C♯5 D5 C♯5 D5 C♯5 E5 E5 F♯5 E5 F♯5 E5 E5 D5 C♯5 D5 C♯5 C♯5 B4 A4 G♯4 A4". La distanza fra queste due stringhe è 27, pari alla lunghezza della stringa più lunga. Quindi queste 2 stringhe sono completamente diverse. Abbiamo già detto che occore confrontare i brani nella stessa tonalità.

La Figura 5.6 mostra il brano di Al Bano trasposto nella stessa tonalità, Re maggiore, del brano di Jackson. La stringa che rappresenta tale melodia del brano trasposto è: "D4 D4 E4 D4 E4 D4 F♯4 F♯4 G4 F♯4 G4 F♯4 A4 A4 B4 A4 B4 A4 A4 G4 F♯4 G4 F♯4 F♯4 E4 D4 C♯4 D4". La distanza fra le due stringhe ora è 11.

In modo analogo si potrebbe usare una rappresentazione testuale che codifica solo gli accordi. Il brano di Jackson diventerebbe "D Em/D D Em/D D Em/D D Em/D" mentre quello di Al Bano "D Em/D D Em/D D Em/D D Bm D G A". La distanza testuale fra queste 2 stringhe è

Figura 5.6: I cigni di Balaka trasposto in Re

4. Se consideriamo che gli ultimi 2 accordi del secondo brano sono solo di collegamento e quindi potremmo anche eliminarli, la seconda stringa diventerebbe "D Em/D D Em/D D Em/D D Bm D" e la distanza testuale sarebbe 2. Se i brani fossero stati considerati nelle tonalità originali la distanza sarebbe stata quella massima.

5.2.2 Altre rappresentazioni

La rappresentazione testuale che abbiamo considerato è forse la più semplice possibile in quanto l'unica informazione che viene inserita nella stringa è la sequenza delle note. Tutto ciò che va oltre tale sequenza, come ad esempio la durata di ogni singola nota, viene ignorato. Non dovrebbe sorprendere che una codifica così semplice funziona bene in alcuni casi ma non in tutti.

Ad esempio, una codifica più complessa potrebbe inserire anche informazioni riguardo la durata delle note oppure codificare altri aspetti della melodia. Vediamo alcune possibilità. Un problema comune a tutte le rappresentazioni è quello di poter usare più caratteri per rappresentare una singolo elemento della partitura mentre per usare gli algoritmi su stringhe dovremmo usare un solo carattere. In realtà allargando l'alfabeto è possibile usare un solo carattere ma si perde in leggibilità (umana, non per il computer). Dunque per comododità i nostri "caratteri" saranno in realtà delle stringhe loro stessi, stringhe che chiameremo *meta*caratteri. Dal punto di vista logico però ogni sottostringa che rappresenta una singola nota deve essere considerata un solo carattere. Quindi, ad esempio, "A♭4", che rappresenta il La bemolle nella quarta ottava è un solo "(meta)carattere". Inoltre useremo lo spazio per seperare questi metacaratteri.

Rappresentazione NOTE. Questa è la rappresentazione che abbiamo già visto: ogni metacarattere è composta da una o o più lettere che ci dicono il nome della nota e eventuali alterazioni (♯ e ♭) e un numero che indica l'ottava in cui si trova. Magari possiamo aggiungere un carattere per rappresentare la pausa. Useremo la lettera "p". Come esempio, consideriamo la melodia riportata nella Figura 5.7. La sua rappresentazione NOTE è: p C5 B4 D5 C5 B4 Bb4 G♯4 A4 A4 A4 G♯4 B♭4 A4 A♭4 G4 E4 F4 F4 E4 F4 G4 A4 B♭4 D5 C5 B4 D5 C5 G4 A4 B4 C5 D5 C♯5 D5 E5 D5 B4 A4 G4 C5 B4 D5 C5 B4 B♭4 G♯4 A4 A4 A4 G♯4 B♭4 A4 G♭4 G4.

Rappresentazione PITCH. La rappresentazione NOTE funziona molto male quando due melodie simili sono presentate in tonalità diverse. Per aggirare questo problema possiamo pensare alla seguente rappresentazione che chiameremo PITCH e che da importanza agli intervalli fra note successive. Al posto di rappresentare le singole nota rappresentermo gli intervalli fra due noye successive misurandolo in semitoni. Anche in questa caso potremmo facilmente includere le pause rappresentadole con la lettera p La melodia della Figura 5.7 sarebbe rappresentata da: p -1 3 -2 -1 -1 -2 1 0 0 -1 2 -1 -1 -1 -3 1 0 -1 1 2 2 1 4 -2 -1 3 -2 -5 2 2 1 2 -1 1 2 -2 -4 -1 -2 5 -1 3 -2 -1 -1 -2 1 0 0 -1 2 -1 -3 1.

Rappresentazione PITCH-DA. Le precedenti rappresentazioni ignorano completamente le durate delle note. Vediamo ora una rappresentazione che invece include anche la durata. Definiamo l'alfabeto delle durate come $D = \{ w, h, q, e, s\}$, nel quale abbiamo messo dei simboli per

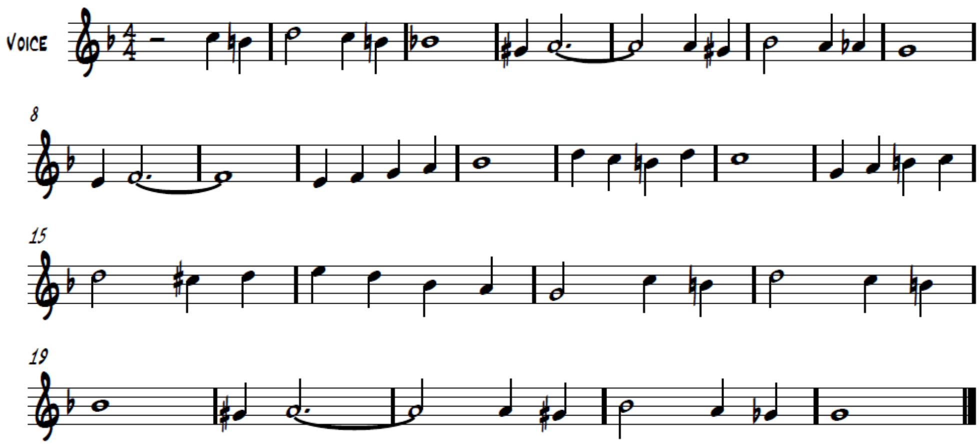


Figura 5.7: Frammento della melodia del brano *"Near you"*, di Francis Craig.

rappresentare le durate: w = *whole* (4/4), h = *half* (2/4), q = *quarter* (1/4), e = *eight* (1/8), e s = *sixteenth* (1/16). Possiamo aggiungere tale simbolo subito dopo l'intervallo, unendo così tale informazione con quella della rappresentazione PITCH. La rappresentazione PITCH-DA per la melodia della Figura 5.7 è: ph q -1q +3h -2q -1q -1w -2q +1h. 0h 0q -1q +2h -1q -1q -1w -3q +1h. 0w -1q +1q +2q +2q +1w +4q -2q -1q +3q -2w -5q +2q +2q +1q +2h -1q +1q +2q -2q -4q -1q -2h +5q -1q +3h -2q -1q -1w -2q +1h. 0h 0q -1q +2h -1q -3q +1w.

Rappresentazione PITCH-ND**.** L'utilizzo di durate "assolute" può creare un problema simile a quello della traspozione. Un brano che è semplicemente scritto con le note che durano tutte il doppio (o la metà) di un altro brano verrà considerato completamente diverso mentre in realtà sono uguali. Per ovviare a questo problema possiamo pensare a rappresentare le dirate in termine di un "massimo comun divisore". Più in dettaglio, possiamo pensare di avere un simbolo, "b", che rappresenta una durata di base e poi rappresentare le durate in termini di ripetizioni di questo simbolo in proporzione alla reale durata. Per stabilire il valore del simbolo di ripetizione (cioè quale durata rappresenta) possiamo analizzare preliminarmente la melodia da rappresentare e scegliere la durata più grande che permetta di rappresentare tutte le durate delle note e della pause della melodia come multipli di tale durata (appunto un massim comun divisore).

Per la melodia della Figura 5.7 il simbolo di ripetizione rappresenterebbe la durata di una semiminima e la stringa è: pbb b -1b +3bb -2b -1b -1bbbb -2b +1bbbbb 0b -1b +2bb -1b -1b -1bbbb -3b -2bbbbbbb -1b +1b +2b +2b +1bbbb +4b -2b -1b +3b -2bbbb -5b +2b +2b +1b +2bb -1b +1b +2b -2b -4b -1b -2bb +5b -1b +3bb -2b -1b -1bbbb -2b -1bbbbb 0b -1b +2bb -1b -3b +1bbbb.

Rappresentazione PITCH-NDT La rappresentazione PITCH-ND presenta però un problema: se nella melodia è presente una nota o pausa di una durata molto breve (si pensi ad esempio a una acciaccatura) o comunque se il "massimo comun divisore" risulta essere estremamente piccolo rispetto alla quasi totalità delle durte da rappresentare, la stringa risultante diventerebbe eccessivamente lunga. Per risolvere questo problema si potrebbe scegliere un "massimo comun divisore" più grande ignorando note di durata poco frequente.

Esercizi

1. Quale è la edit distance fra `ababca` e `cbaca`? Indicare anche la sequenza di operazioni che permettono di trasformare una stringa nell'altra usando un numero di operazioni pari alla edit distance.

2. La edit distance può essere usata come metrica di similitudine fra due stringhe. Rappresentando la musica con delle stringhe si può sfruttare la edit distance per fornire una metrica di similitudine fra due melodie. Tuttavia per rappresentare la musica è comodo usare più di un carattere per ogni nota. Come si può adattare la edit distance per gestire questo aspetto?

3. Definire una rappresentazione testuale che permetta di descrivere una melodia specificando le note; in particolare la rappresentazione deve considerare l'altezza delle note e la durata permettendo di rappresentare le durate di semibrevi, minime, semiminime, ... fino alle semibiscrome.

4. Si consideri la seguente rappresentazione testuale per descrivere delle melodie: una stringa di metacaratteri in cui ogni metacarattere ha una lettera per il nome della nota e un numero per l'ottava. Qualsi sono i vantaggi e gli svantaggi di una tale rappresentazione?

5. Definire una rappresentazione testuale che evita il problema della trasposizione.

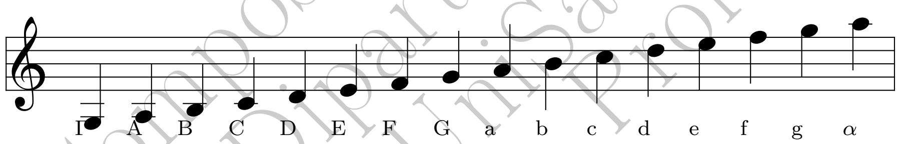
Capitolo 6

Composizione automatica semplice

In questo capitolo parleremo di alcune tecniche di composizione automatica relativamente semplici. La semplicità è dovuta al fatto che si basano su procedimenti che non hanno nulla a che fare con la musica (in alcuni casi studiati in altri contesti) sui quali si innestano delle regole semplici per produrre musica.

6.1 Metodo di Guido

Guido d'Arezzo, importante teorico musicale vissuto intorno all'anno 1000, fra i tanti contributi alla teoria musicale, aveva inventato un compositore "algoritmico" che partendo da un testo produceva una melodia. L'algoritmo in realtà è randomizzato in quanto prevede delle scelte non deterministiche. Il metodo prima associa un simbolo ad ognuna delle note in un intervallo di 2 ottave, tipico dell'epoca, come mostrato di seguito:



Quindi utilizza una ulteriore associazione di questi simboli con le vocali a, e, i, o, u. Avendo solo 5 vocali, Guido crea un'associazione ciclica per la quale alla stessa vocale corrispondono più note.

Γ	A	B	C	D	E	F	G	a	b	c	d	e	f	g	α
a	*e*	*i*	*o*	*u*	*a*	*e*	*i*	*o*	*u*	*a*	*e*	*i*	*o*	*u*	*a*

Partendo da un testo si estraggono le vocali e da queste si crea una sequenza di note. Per ogni vocale in realtà c'è da scegliere una fra 3 (o 4) note e la scelta viene fatta in modo casuale.

Ad esempio partendo dal testo

"dalle parole si creano le note"

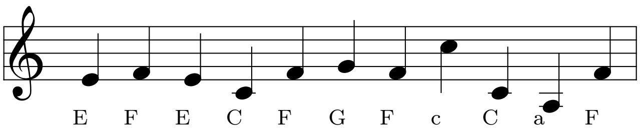
si estraggono le vocali

a e a o e i e a o e o e

e da queste si ottiene una lista di (multi)note

[ΓEcα] [AFd] [ΓEcα] [Caf] [AFd] [BGe] [AFd] [ΓEcα] [Caf] [AFd] [Caf] [AFd]

ed infine, scegliendo una sola nota per ognuna delle possibilità si ottiene, ad esempio:



6.2 Il gioco dei dadi

Il cosiddetto "gioco dei dadi" per la creazione di composizioni musicali è un sistema di composizione basato sulla generazione casuale di un numero fra 2 e 12 (lancio di 2 dadi) ripetuta per un prefissato numero di volte. Ogni numero casuale determina la selezione di una battuta musicale fra 11 disponibili. Il tutto si basa sulla creazione a priori delle battute musicali disponibili per la selezione.

Per capire meglio usiamo un analogo con le parole. Supponiamo di voler creare un "compositore di frasi". Ogni possibile frase generata dal compositore consta di 6 pezzi (singole parole o anche più parole). Ogni pezzo viene scelto fra 3 possibilità, stabilite a priori. La prima parola può essere scelta fra (1) "*Il*", (2) "*Un*" e (3) "*Quel*". La seconda parola può essere scelta fra (1) "*cane*", (2) "*gatto*" e (3) "*leone*". La seguente tabella riporta le 3 possibilità per ognuna delle 6 parole.

	1	2	3	4	5	6
1	Il	cane	sta	correndo	nel	cortile
2	Un	gatto	va	saltando	in quel	bosco
3	Quel	leone	stava	camminando	fuori dal	parco

Per creare una frase si sceglie per 6 volte un numero fra 1 e 3. Per ogni scelta si seleziona la corrispondente parola. Ad esempio alla sequenza di scelte 1,2,2,3,2,1 corrisponde la frase "Il gatto va camminando nel cortile" Alla sequenza 1,1,1,1,1,1 corrisponde la frase "Il cane sta correndo nel cortile". Alla sequenza 2,1,3,3,2,3 corrisponde la frase "Un cane stava camminando in quel parco". Chiaramente esistono 3^6 possibili frasi, che in realtà sono tutte molto simili. Il trucco è quello di scegliere per ognuna delle 6 parole/pezzi 3 alternative che funzionano bene qualunque siano le altre scelte.

Il gioco dei dadi per la composizione musicale si basa sullo stesso principio: si decide la lunghezza in battute della composizione, ad esempio 16 battute. Per ognuna delle 16 battute si compongono 11 possibilità: ogni singola possibilità deve legarsi bene ad ogni possibile battuta precedente e ogni possibile battuta successiva.

Vari compositori del 18^o secolo hanno composto (programmato?) dei compositori "automatici" di questo genere. Uno dei più conosciuti è quello di Mozart (o quanto meno a egli attribuito) che prevede 16 battute. Ogni battuta viene selezionata casualmente fra 11 possibilità. Quindi Mozart ha previsto $16 \times 11 = 176$ battute, in 16 gruppi di 11 battute. Le 176 battute sono state numerate (in modo arbitrario e non successivo, probabilmente per creare un effetto più "magico") e la Tabella nella Figura 6.1 riporta i gruppi di battute.

	Battute															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	96	22	141	41	105	122	11	30	70	121	26	9	112	49	109	14
3	32	6	128	63	146	46	134	81	117	39	126	56	174	18	116	83
4	69	95	158	13	153	55	110	24	66	139	15	132	73	58	145	79
5	40	17	113	85	161	2	159	100	90	176	7	34	67	160	52	170
6	148	74	163	45	80	97	36	107	25	143	64	125	76	136	1	93
7	104	157	27	167	154	68	118	91	138	71	150	29	101	162	23	151
8	152	60	171	53	99	133	21	127	16	155	57	175	43	168	89	172
9	119	84	114	50	140	86	169	94	120	88	48	166	51	115	72	111
10	98	142	42	156	75	129	62	123	65	77	19	82	137	38	149	8
11	3	87	165	61	135	47	147	33	102	4	31	164	144	59	173	78
12	54	130	10	103	28	37	106	5	35	20	108	92	12	124	44	131

Figura 6.1: Tabella gioco dei dadi di Mozart

La Figura 6.2 mostra, in miniatura, le 176 battute mettendole nella sequenza della numerazione, cioè dalla battuta 1 alla battuta 176.

Figura 6.2: Le 176 battute del gioco dei dadi di Mozart.

La Figura 6.3 mostra, in modo leggibile, le 11 possibilità previste per le prime 4 battute. La prima battuta sarà selezionata fra quelle numerate 96, 32, 69, 40, 148, 104, 152, 119, 98, 3 e 54. L'armonia è sempre quella di Do maggiore; ognuna di queste battute è diversa dalle altre

Figura 6.3: Le prime 4 battute (11 possibilità per ognuna)

per l'aspetto ritmico, per quello melodico o per entrambi. La seconda battuta sarà selezionata fra quelle numerate 22, 6, 95, 17, 74, 157, 60, 84, 142, 87 e 130. Anche per la seconda battuta l'armonia è quella di Do maggiore e anche in questo caso le battute sono diverse fra loro. Si può notare come però alcune di queste battute abbiano lo stesso contenuto di battute del gruppo precedente. La battuta 22 è uguale alla 96 e la battuta 6 è uguale alla 32, e ce ne sono altre. Forse la numerazione arbitraria serve anche a nascondere queste uguaglianze. Il gruppo di alternative per la terza battuta utilizza l'armonia di Sol maggiore, mentre quello della quarta battuta ritorna nell'armonia di Do. Dunque qualunque sia il responso dei dadi, la composizione creata avrà nelle prime 4 battute sempre l'armonia di Do-Do-Sol-Do, cioè I-I-V-I. E questa cosa è vera per tutti i gruppi; dunque l'aspetto armonico di tutto il brano in realtà è fissato a priori dal compositore e non dipende dalle scelte casuali.

Una implementazione disponibile in rete (anche se la musica non corrisponde esattamente a quella mostrata nelle figure precedenti) è:

`http://www.playonlinedicegames.com/mozart`

6.3 Compositori basati su Automi Cellulari

Gli automi cellulari, introdotti da John von Neumann [?] sono stati studiati tempo fa (negli anni 1960) come modello di auto-riproduzione.

6.3.1 Automi cellulari

Un automa cellulare è costituito da 4 componenti: un insieme di cellule, una relazione di "vicinato" fra le cellule, un insieme di stati in cui si possono trovare le cellule e delle regole per l'evoluzione delle cellule. Il nome deriva dal fatto che gli automi cellulari sono stati inizialmente utilizzati per studiare il processo biologico della riproduzione. Spesso le cellule vengono rappresentate in uno spazio bidimensionale su una griglia di celle. Pertanto, almeno in questo contesto, utilizzaremo i termini *cella* e *cellula* come sinonimi.

L'universo di un automa cellulare è costitutio dall'insieme delle celle. La Figura 6.4 mostra un universo bidimensionale. In generale l'universo di un automa cellulare è infinito; tuttavia poichè una qualsiasi implementazione deve necessariamente usare una rappresentaione finita, solitamente si sceglie di rappresentare l'universo come uno spazio toroidale bidimensionale in cui un estremo diventa continuo con l'estremo opposto, come mostrato nella figura.

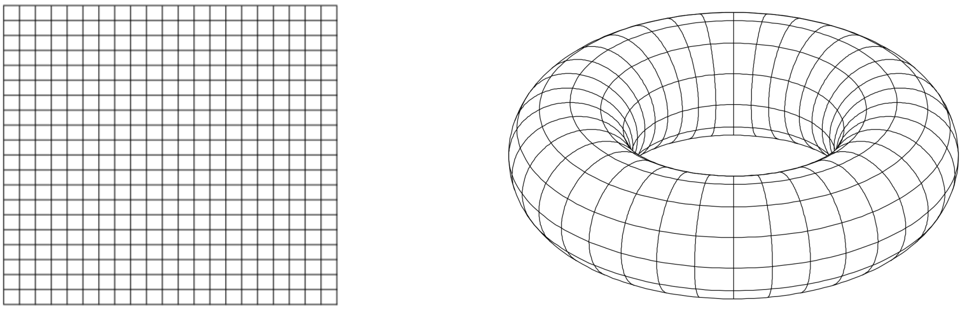


Figura 6.4: Universo bidimensionale e spazio toroidale

La relazione di vicinato può essere arbitraria ma tipicamente rispecchia la struttura fisica dell'universo dell'automa. Con un universo monodimensionale ogni cella ha come vicini le due celle adiacenti. Con universi bidimensionali ogni cella avrà come vicini le 4 celle con le quali condivide un lato oppure le 8 celle con le quali condivide o un lato on uno spigolo. Nulla vieta la definizione di una qualsiasi altra (arbitraria) relazione di vicinato, come mostrato nella Figura 6.5.

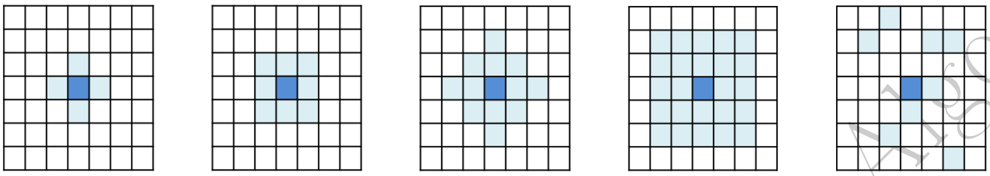


Figura 6.5: Vicinati

Ogni cella dell'automa, in un dato istante di tempo, si trova in un determinato stato. Lo stato dell'automa è dato dall'insieme degli stati di tutte le celle dell'automa. Consideriamo un esempio semplice in cui ogni cella può trovarsi in soli due stati, che possiamo decsrivere graficamente con una cella vuota e una cella piena. La Figura 6.6 mostra uno specifico stato dell'automa.

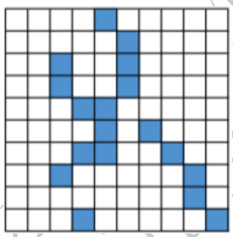


Figura 6.6: Stato

Infine ci sono le regole di evoluzione. Un automa cellulare evolve partendo da una situazione iniziale in cui ogni cella dell'universo è in un particolare stato e a ogni istante discreto di tempo aggiorna lo stato di ogni cella dell'universo secondo determinate regole che sono fisse, cioè fanno parte della definizione dell'autoname cellulare. Le regole determinao lo stato di una particolare cella per il successivo sitante di tempo in funzione degli stati del vinicato della cella, incluso lo stato attuale della cella stessa.

6.3.2 Un esempio semplice

Consideriamo il seguente automa cellulare costituito da uno spazio toroidale monodimensionale di $n = 10$ celle e supponiamo che l'insieme degli stati di una cella contenga solo $s = 2$ stati, che interpreteremo come "spento" e "acceso".

Figura 6.7: Spazio toroidale monodimensionale

Il vicinato di ogni cella è costituito dalla cella stessa e dalle due celle ad essa adiacente. La regola di evoluzione è la seguente: una cella c è accesa al tempo $t + 1$ se e solo se al tempo t le tre celle del suo vicinato, cioè la cella alla sua sinistra, la stessa cella c e la cella alla destra di c, sono nella configurazione (acceso, spento, spento). Se la configurazione iniziale ha una sola cella accesa, l'effetto dell'evoluzione è uno spostamento verso destra della cella accesa ad ogni istante di tempo. La Figura 6.8 mostra una tale evoluzione.

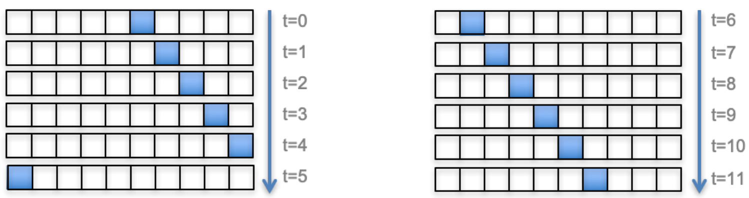


Figura 6.8: Esempio evoluzione

6.3.3 Game of Life

L'automa cellulare conosciuto come *Game of Life* è stato ideato da John Conway. Nel Gioco della Vita, l'universo delle celle è bidimensionale, il vicinato di una cella è costituito dalle 8 celle che condividono o un angolo o un lato con la cella stessa, gli stati sono due, cell(ul)a viva e cell(ul)a morta, e le regole di evoluzione sono le seguenti:

1. Una cella morta che al tempo t ha esattamente 3 vicini vivi diventa viva (nascita).

2. Una cella viva che al tempo t ha 2 o 3 vicini vivi rimane viva (sopravvivenza).

3. In tutti gli altri casi la cella muore o rimane morta (sovraffollamento o solitudine).

Data la popolarità di tale automa cellulare è molto facile trovare implementazioni sul World Wide Web. Una disponibile (almeno al tempo di scrittura) e facilmente utilizzabile è la seguente: `https://playgameoflife.com`. Si sperimentino delle configurazioni iniziali per capire come procede l'evoluzione.

6.3.4 Cyclic Cellular Automaton

Un altro automa cellulare semplice è quello proposto da Griffeath et al. [?]: l'universo è bidimensionale come per il gioco della vita, ma gli stati non sono più solo 2 bensì p, dove p è un parametro, e sono numerati da 0 a $p-1$. C'è una sola regola di evoluzione che utilizza un altro parametro, una soglia t. La regola è la seguente: se per una cella (i,j) che si trova nello stato k ci sono almeno t celle adiacenti nello stato $k+1 \mod p$ allora la cella passa anch'essa nello stato $k+1 \mod p$. La Figura 6.9 mostra lo stato di un automa cellulare ciclico, con 12 colori, soglia 1, dopo 10.000 iterazioni dell'evoluzione partita da una configurazione iniziale casuale.

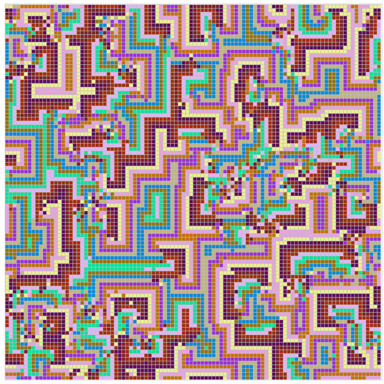


Figura 6.9: Esempio stato CCA

6.3.5 Compositore automatico basto su automi cellulari

Vari compositori automatici basati su automai cellulari sono stati proposti (es. CAMus). In questa sezione descriveremo un compositore automatico basato su automi cellulare simile a quelli proposti in letteratura.

Utilizzeremo sia il gioco della vita e che l'automa cellulare ciclico. Entrambi gli automi vengono usati in spazi bidimensionali aventi le stesse dimensioni e l'evoluzione procede in parallelo. A ogni passo, per ogni cellula viva nel gioco della vita, il colore della corrispondente cellula nell'automa ciclico determina una nota di partenza, mentre le coordinate delle cellula determinano delle note da suonare. Più in dettaglio se la cellula con coordinate (i,j) è viva, viene generata una tripla di note a partire da una nota di riferimento: l'intero i rappresenta il numero di semitoni di distanza della seconda nota dalla nota di riferimento mentre j specifica il numero di semitoni di distanza della terza nota dalla seconda, come mostrato nella Figura 6.10. La nota di riferimento è generata a partire dal colore della cellula nell'automa ciclico.

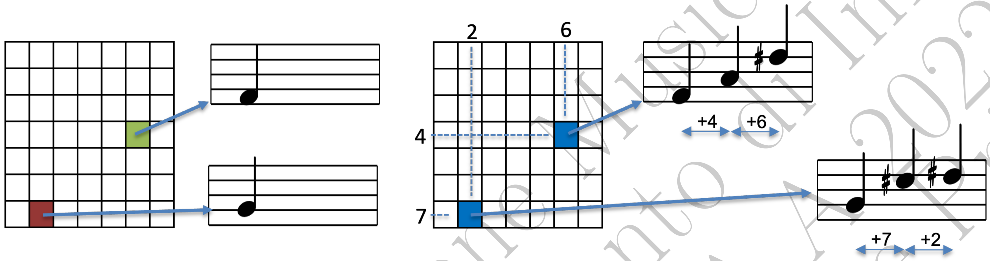


Figura 6.10: Generazione di una tripla di note

Inoltre le 3 note generate possono essere suonate con diversi pattern ritmici che prevedono sia l'esecuzione in sequenza delle note, sia in simultanea di 2 o anche tutte le 3 note e determina anche l'ordine delle 3 note. Per scegliere il pattern ritmico possiamo usare lo stato delle 8 celle adiacenti la cella che ha generato le note. Usando lo stato di 8 cellule possiamo codificare ben 256 diversi pattern ritmici. La Figura 6.11 mostra alcune possibilità.

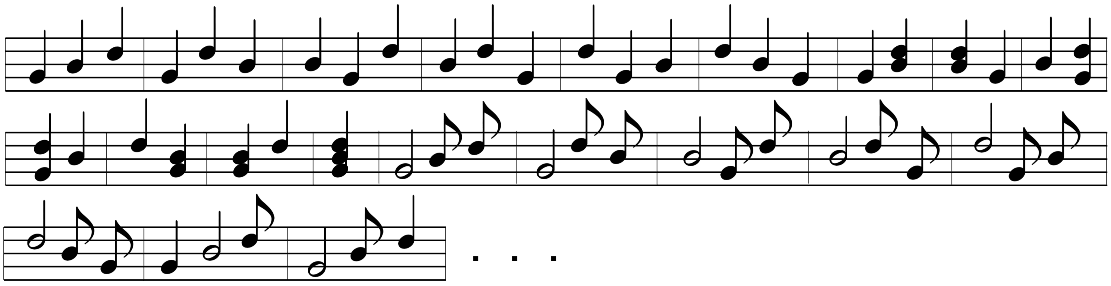


Figura 6.11: Alcuni pattern ritmici

Ad ogni ciclo dell'evoluzione vengono esaminate tutte le celle, per colonne dall'alto verso il basso, partendo dalla colonna più a sinistra e per ogni cella viva si applica il procedimento di generazione delle note.

6.3.6 Web links relativi a CA music

Una spiegazione: `https://www.youtube.com/watch?v=LrAftDGDimE`

Esempi video:

- `https://www.youtube.com/watch?v=ZZu5LQ56T18`
- `https://www.youtube.com/watch?v=RQnmL-OIznc`
- `https://www.youtube.com/watch?v=-HXRrpIYYOU`
- `https://www.youtube.com/watch?v=NvXKd-jCrnY`
- Atari: `https://www.youtube.com/watch?v=8LP7ZfwT3KA`
- `https://www.youtube.com/watch?v=iwC3eZCzowk`

6.4 Fractal Music

6.4.1 Frattale

Un frattale è un oggetto geometrico la cui forma è definita in forma ricorsiva in termine di se stessa. Per questo motivo se guardiamo un pezzo di un frattale ritroviamo la forma del frattale stesso e se usiamo una lente di ingrandimento, a qualsiasi livello, ritroveremo sempre la stessa forma. Diversamente da un forma geometrica euclidea che può essere descritta da una funzione, la costruzione di un frattale necessita di un procedimento che descrive la natura ricorsiva dell'oggetto. Inoltre il procedimento di costruzione è teoricamente infinito; ovviamente in pratica ci si dovrà fermare dopo un certo numero di iterazioni, sufficienti a descrivere in modo dettagliato la parte dell'oggetto che può essere vista dall'occhio umano. Il termine frattale è stato coniato dal matematico Benoît Mandelbrot che è considerato il fondatore della geometria frattale.

Un esempio semplice di oggetto frattale è la cosiddetta curva di Von Kock, studiata nel 1904 dal matematico svedese Helge von Koch. La forma iniziale è un segmento. Il procedimento per la costruzione dell'oggetto frattale è il seguente: il segmento viene diviso in tre parti uguali e la parte centrale viene sostituita da due segmenti uguali alla parte centrale dispositi in modo da formare un triangolo equilatero con la parte centrale tolta. La Figura **??** illustra i primi 2 passi della costruzione.

La Figura 6.13 illustra il risultato del procedimento applicato ai 3 lati di un triangolo equilatero.

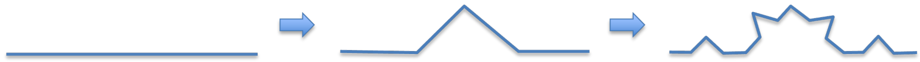


Figura 6.12: Curva di von Cock: procedura ricorsiva.

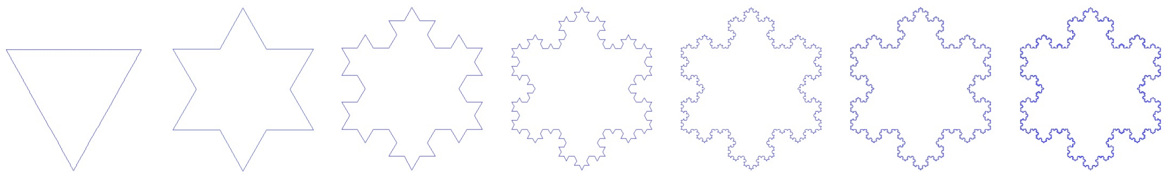


Figura 6.13: Curva di von Cock

6.4.2 Musica frattale

Possiamo usare il principio di base dei frattali anche per la musica, introducendo una struttura ricorsiva nella composizione. Un esempio semplice è il seguente. Partiamo da un elemento melodico, anche molto semplice, ad esempio di sole 2 note. Da questo elemento di base potremmo crearne uno nuovo in cui ogni nota diventa un nuovo elemento di 2 note con la stessa "struttura" dell'elemento da cui viene creato. Ad esempio se le due note sono un Do4 e un Fa4, che creano un intervallo di quarta, ognuna delle due note genera una coppia di note con lo stesso intervallo. Quindi il Do4 genererebbe la stessa coppia di note originali, mentre il Fa4 generebbe la coppia F4 e Si♭. E il procedimento potrebbe continuare all'infinito, magari definendo dei limiti superiori e inferiori entro i quali devono rimanere le note e considerando uno spazio toroidale. Ad esempio, rimanendo fra il G3 e A5, avremmo la generazione mostrata nella Figura 6.14.

Figura 6.14: Elemento melodico e "derivati ricorsivi"

Gli elementi derivati potrebbero anche essere sovrapposti temporalmente, magari aggiungendoli uno per volta, come mostrato nella Figura 6.15.

La trasformazione ricosiva sull'intervallo, o sugli intervalli nel caso di più note, non è l'unica posisbile. Ad esempio possiamo considerare la durata delle note. Prendiamo l'elemento melodico riportato in Figura 6.16, e consideriamo i suoi "derivati ricorsivi" ottenuti dividendo per 2 la durata delle note.

Abbiamo visto dunque due elementi sui quali basare l'aspetto frattale della musica. Ovviamente si possono immaginare anche altre trasformazioni ispirate ai frattali. Inoltre è possibile fare delle considerazioni musicali riguardo le sequenze di note che magari funzionano meglio quando manipolate in questo modo. Ad esempio usando la scala pentatonica si ottengono delle consonanze migliori.

6.4.3 Web links per musica frattale

- `https://www.youtube.com/watch?v=GVK5N7HQf8Y` (Musica frattale)
- `https://www.youtube.com/watch?v=6XTe5EsPE20` (Fractals in Music)
- `https://www.youtube.com/watch?v=GiAj9WW1OfQ` (Sounds of the Mandelbrot set)

Figura 6.15: Elementi sovrapposti temporalmente

Figura 6.16: Elemento melodico e "derivati ricorsivi"

- `https://www.youtube.com/watch?v=cYBAjnMCmO8` (Fractal Symphony)
- `https://www.youtube.com/watch?v=1HqmOdYKUx4` (Music and Music Theory)
- `https://www.youtube.com/watch?v=V5tUM5aLHPA` (Music Symmetry)

6.5 Fibonacci music

La sequenza di Fibonacci è una successione di numeri interi è definita da $F_0 = 0$, $F_1 = 1$ e per $n \geq 2$, $F_n = F_{n-1} + F_{n-2}$:

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, ...$$

I numeri F_n sono anche detti numeri di Fibonacci. Il matematico pisano Leonardo Fibonacci ha introdotto questa sequenza nel 1200 (circa) come soluzione ad un problema matematico che riguardava la crescita di una popolazione di conigli per la quale F_n era il numero di coppie di conigli della popolazione. Il problema posto da Fibonacci assumeva di partire da una singola coppia di conigli appena nati; quindi $F_1 = 1$. Ogni coppia di conigli diventa fertile dopo il primo mese e riproduce una nuova coppia ogni mese a partire dal secondo. Quindi in principio c'è una coppia di conigli. Dopo un mese c'è ancora una coppia di conigli diventata fertile; quindi $F_2 = 1$. Alla fine del secondo mese ci sono 2 coppie di conigli: quella iniziale e quella appena nata. Al terzo mese la coppia iniziale genererà una nuova coppia, mentre la seconda coppia è diventata fertile ma ancora non si è riprodotta. Quindi ci saranno 3 coppie di conigli; quindi $F_3 = 3$. Al mese seguente sono 2 le nuove coppie e quindi ci saranno 5 coppie in totale; quindi $F_4 = 5$. La Figura 6.17 raffigura l'evoluzione della popolazione di coppie di conigli.

Il problema posto e risolto da Fibonacci è quello di trovare una formula esplicita per F_n. Si può notare che aggiungendo $F_0 = 0$, la formula continua a valere.

Sorprendentemente, la sequenza di Fibonacci ha delle proprietà che nulla hanno a che fare con i conigli. Il rapporto fra un qualsiasi numero di Fibonacci e il precedente è una approssimazione del rapporto aureo, e l'approssizione diventa esatta all'infinito, cioè si ha che

$$\lim_{n\to\infty} \frac{F_n}{F_{n-1}} = \frac{1+\sqrt{5}}{2} \simeq 1.61803398874...$$

Il rapporto aureo definisce una proporzione che si trova spessso in natura. Tale proporzione è ben esplicitata dalla spirale aurea che si può facilmente costruire tramite la sequenza di Fibonacci.

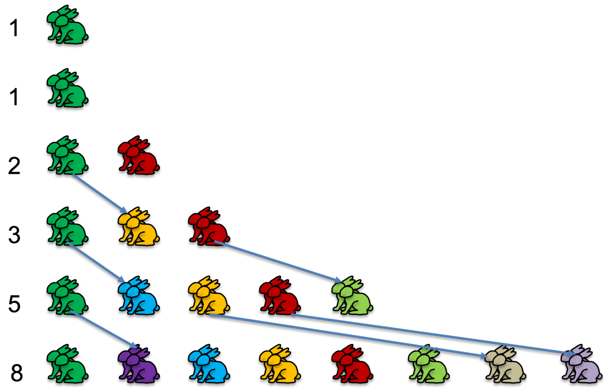


Figura 6.17: Evoluzione popolazione di coppie di conigli

Infatti i quadrati costruiti su lati di lunghezza pari ai numeri di Fibonacci possono essere posizionati uno a fianco all'altro in modo tale che unendo con dei quarti di cerchio gli estremi opposti, si ottiene la spirale, come mostrato nell Figura 6.18.

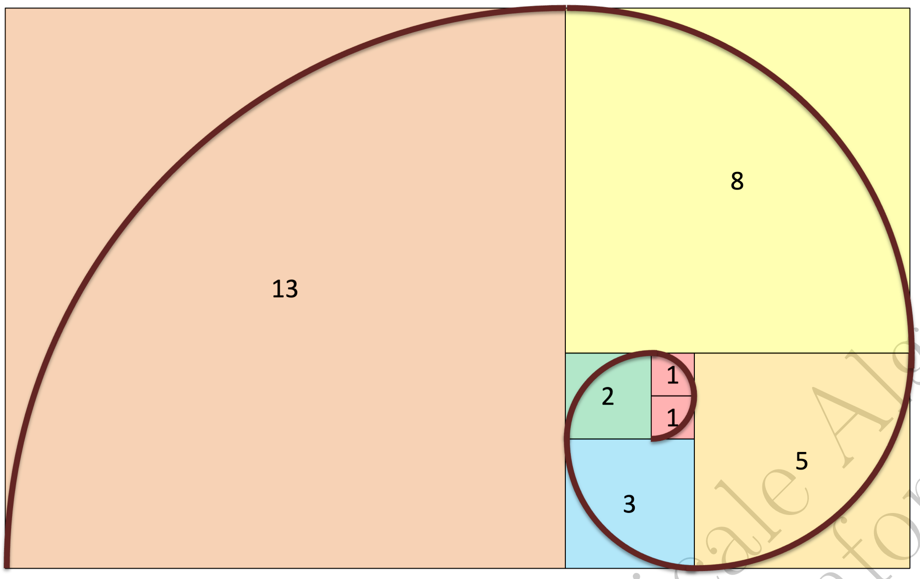


Figura 6.18: Spirale aurea costruita sui quadrati di Fibonacci

I numeri di Fibonacci e la spirale aurea appaiono spesso in natura. Ad esempio moltissi fiori hanno un numero di petali pari a un numero di Fibonacci. Ad esempio le calle hanno un petalo, i gigli ne hanno 3, le rose canine 5, il camedrio alpino 8, l'anemona hortensis ne ha 13. La classica margherita dei prati ha normalmente 21 petali.

(https://it.pearson.com/aree-disciplinari/scienze-matematica/articoli/fiori-fibonacci.html)

Moltissime conchiglie hanno forme che ricalcano la spirale aurea, come mostrato nell Figura 6.19.

E questi sono solo pochissimi esempi dell'apparizione dei numeri di Fibonacci e della sezione aurea in natura. Questa pervasività ha suscitato da sempre l'attenzione verso la successione di Fibonacci e il rapporto aureo. Pittori e architetti hanno usato, consapevolemnte o intuitivamente, la proporzione aurea. E anche i musicisti.

Figura 6.19: Spirale aurea delle conchiglie

Vediamo ora degli approcci per costruire musica basandoci sulla sequenza di Fibonacci. Se consideriamo le singole cifre nell'intera sequenza avremmo

$$0, 1, 1, 2, 3, 5, 8, 1, 3, 2, 1, 3, 4, 5, 5, 8, 9, 1, 4, 4, 2, 3, 3, 3, 7, 7, 6, 1, 0, 9, 8, 7, ...$$

Possiamo far corrispondere ogni cifra a una nota di una scala. Prendiamo per esempio la scala maggiore. Avendo 10 cifre ma solo 8 note nella scala (considerando l'ottava come un'altra nota) dobbiamo aggiungere ulteriori due note e potremmo prendere quella prima e quella dopo. Ad esempio nella figura 6.20 le 10 cifre sono messe in corrispondenza con la scala di Do maggiore, partendo da un Si per arrivare a un Re.

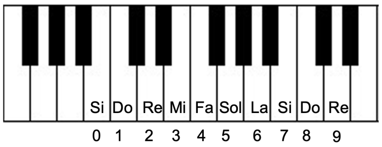


Figura 6.20: Corrispondenza cifre e note

Questa associazione produrrebbe la sequenza di note: $B3, C4, C4, D4, E4, G4, C5, C4, E4, D4, C4, E4, F4, G4, G4, C5, D5, C4, F4, F4, D4, E4, E4, E4, B4, B4, A4, C4, B4, D5, C4, B4, ...$

- `https://www.youtube.com/watch?v=IGJeGOw8TzQ` (Coding the Fibonacci Seqwuence into Music - piano aSongScout)
- `https://www.youtube.com/watch?v=z7vovDiPjW4` (Pi Greco - piano aSongScout)
- `https://www.youtube.com/watch?v=7bod8x0LgJs` (Creating Music Using the Fibonacci sequence - guitar)

Esercizi

1. Descrivere il metodo di composizione automatico ideato da Guido d'Arezzo.

2. Descrivere il gioco dei dadi per la creazione automatica di musica.

3. Si consideri il gioco dei dadi per una composizione lunga 10 battute e con il lancio di un solo dado. Quante varianti occorre pre-comporre per ogni battuta? Quante sono le possibili diverse composizione che possono essere composte lanciando il dado?

4. Nel gioco dei dadi per la composizione automatica quale è l'elemento chiave (dovuto alla bravura di chi pre-compone le battute) per il quale la musica composta risulta gradevole?

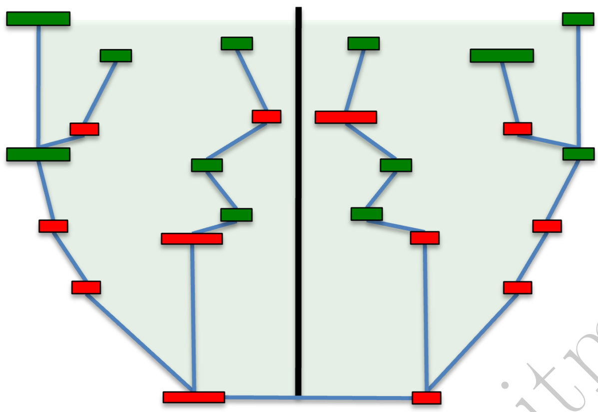
5. Descrivere l'automa cellulare denominato *Game of life*.

6. Descrivere l'automa cellulare denominato *Cyclic Cellular Automaton*.

7. Si faccia un esempio di un procedimento di composizione automatica che può essere considerato *frattale*.

Capitolo 7

Musica e simmetria



In questo capitolo parleremo della relazione fra simmetria e musica.

7.1 Simmetria

Un oggetto si dice simmetrico quando è possibile applicare una trasformazione dell'oggetto che in realtà mantiene l'oggetto inalterato. La Figura 7.1 mostra una figura simmetrica: l'operazione di trasformazione consiste in una rotazione rispetto all'asse verticale centrale; una rotazione di 180^o fornisce una figura che è identica a quella iniziale. Ovviamente altre trasformazioni, inclusa la stessa rotazione con angoli diversi, cambiano l'immagine (stiamo implicitamente assumendo che il punto di osservazione non cambia). Quindi questa immagine è simmetrica solo rispetto alla rotazione di 180^o sull'asse centrale verticale.

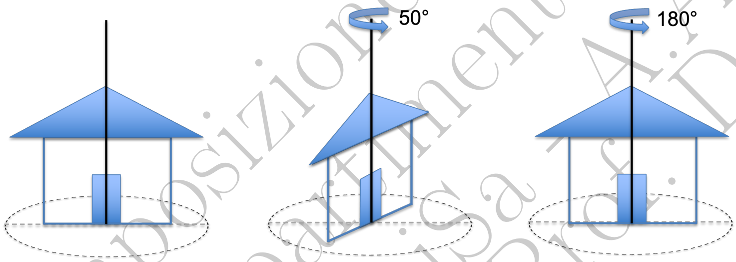


Figura 7.1: Figura simmetrica rispetto a rotazione su un asse verticale

La Figura 7.2 mostra una stella marina che è simmetrica rispetto alla rotazione di 360^o sull'asse orizzontale. Con una certa approssimazione, la stella marina è simmetrica anche rispetto a rotazioni che sono multipli di un quinto di 360^o.

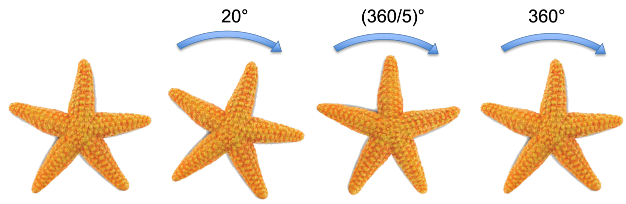


Figura 7.2: Figura simmetrica rispetto a rotazione su un asse orizzontale

La Figura 7.3 mostra un mosaico moresco del palazzo-fortezza dell'Alhambra a Granada in Spagna che è simmetrico sia rispetto alla rotazione di 360^o che alle traslazioni verticali, orizzontali e oblique che portano una stella di un determinato colore nella stessa posizione della successiva stella dello stesso colore.

Figura 7.3: Un mosaico moresco

Il nostro stesso volto, con una certa approssimazione, è simmetrico rispetto all'operazione di riflessione sull'asse centrale.

7.2 Simmetria in musica

La musica è piena di aspetti simmetrici, o quanto meno quasi-simmetrici. In una partitura di musica classica, è facile trovare degli elementi musicali che si ripetono. Come esempio consideriamo il preludio n. 1, in Do maggiore, dal primo libro del clavicembalo ben temperato di Bach (BWV 846). La Figura 7.4 mostra le prima 2 battute. Ogni battuta è formata da 2 parti di cui la seconda è una ripetizione della prima. Quindi una traslazione temporale, di mezza battua, della prima parte fa sovrappore in maniera perfetta la prima parte della battuta con la seconda parte.

Figura 7.4: Bach, Preludio in Do, BWV 846

La traslazione su altre battute non combacia perfettamente in quanto in ogni battuta c'è un cambio di armonia, ma la struttura ritmica resta inalterata. Quindi focalizzando l'attenzione solo sulla struttura ritmica avremmo una simmetria traslazionale che copre (quasi) tutta la composizione.

Figura 7.5: Beethoven, battute iniziali della 5^a sinfonia

Il famoso tema inizale della 5^a sinfonia di Beethoven presenta varie simmetrie. L'elemento melodico dell'incipit, presentato nelle prime 2 battute, viene ripetuto con una traslazione verso il basso nelle due battute successive. Poi lo stesso elemento melodico viene riproposto seguito da 2 quasi-traslazioni verso l'alto con ogni ripetizione parzialmente sovrapposta alla precedente. Ne scaturisce una cellula musicale di 4 battute, dalla 6 alla 9 che viene ripetuta quasi identica (con un armonia diversa) dalla battuta 10 alla 13.

Quindi prosegue con un elemento melodico che ha la stessa struttura ritmica, 3 crome seguite da una minima, alle battute 14 e 15 che viene poi riproposto con una simmetria orizzontale nelle battute 15 e 16 e la cosa si ripete nelle battute successive.

Figura 7.6: Esempio di inverso e retrogrado

In moltissime composizioni è facile trovare frammenti melodici riproposti con una trasformazione simmetrica. Dato un frammento melodico, si può considerare il suo simmetrico *retrogrado*, quello *inverso* e e quello *retrogrado-inverso*. Il retrogrado è ottenuto semplicemente ripetendo le note dalla prima all'ultima, quindi invertendo l'ordine temporale; dqnue è un simmetria rispetto a riflessione su un asse verticale centrale. L'inverso è ottenuto invertendo gli intervalli e quindi è una riflessione orizzontale. Il retrogrado-inverso è ottenuto facendo entrambe le cose ed è dunque un riflessione sia orizzontale che verticale. La Figura 7.6 mostra un esempio. Volendo descrivere la melodia con una stringa di meta-caratteri, come abbiamo visto nei precedenti capitoli, avremmo "C5s G4c E5c D5s F5c E5c D5s B4s E4m". La stringa che rappresenta il retrogrado è ottenuta invertendo i meta-caratteri: "E4m B4s D5s E5c F5c D5s E5c G4c C5s". Per l'inversione possiamo usare una rappresentazione che considera gli intervalli sui gradi della scala. La melodia sarebbe "-4 +6 -2 +3 -2 -2 -3 +2". L'inverso inverte il segno: "+4 -6 +2 -3 +2 +2 +3 -2". Si noti che abbiamo considerato gli intervalli misurati sui gradi della scala. Se usiamo i semitoni abbiamo qualcosa di diverso; invertire gli intervalli in termini di semitoni porterebbe a note fuori dalla scala. Ovviamente è possibile considerare anche inversioni di questo tipo.

7.2.1 Simmetria negli accordi

Sappiamo che un accordo è formato da intervalli. La grandezza delgi intervalli dipende dagli accordi. Per un accordo maggiore gli intervalli sono +4 e +3. Volendo aggiungere l'ottava, avremmo 4+, +3 e ovviamente +5 in quanto la somma totale deve essere 12. L'intervallo di 12 semitoni può essere diviso in intervalli di grandezza uguale, ad esempio di 3 o di 4, formado così accordi simetrici rispetto aggli intervalli. L'accordo +3,+3,+3,+3, ad esempio è l'accordi di settiman diminuita. L'accordo +4,+4,+4 invece è l'accordo di quinta aumentata. Sono possibili anche altre suddivisioni, come ad esempio +2,+2,+2,+2,+2,+2 che genera un "accordo" non classificato nella musica classica; in realtà questo "accordo" è formato da tutte le note della scala esatonale, in quanto le 6 note sono tutte a distanza di un tono. Come pure +6,+6 che però non genera un accordo in quanto ci sono solo 2 suoni.

7.2.2 Simmetria della forma

La forma musicale fa riferimento alla struttura di una composizione. Speeso una composizione segue una ben specifica forma. Ad esempio, una tipica forma musicale è quella detta "trepartita", cioè composta da 3 parti, con la prima e la terza uguali. Solitamente si usa descriveremo una tale forma con A-B-A per indicare che ci sono appunti 3 parti con la prima e la terza uguali. La forma tripartita rappresenta una "meta" simmetria della composizione: la parte centrale funge da "asse" per una simmetria di inversione orizzontale.

7.2.3 Simmetria nel canone

Il canone musicale è un forma musicale che usa una melodia, o delle sue trasformazioni, che si sovrappone a se stessa. La sovrapposizione è fatta in modo tale che le armonie che vengono create dalla sovrapposizioni sia gradevoli. Un esempio semplice è la melodia di Fra Martino. Traslando la stessa melodia di 2 battute e sovrapponendola a quella originale si ottiene un canone, come mostrato nella Figura 7.7.

Figura 7.7: Canone con la melodia di Fra Martino

7.2.4 Duetto dello specchio

7.3 Dodecafonia

Nel Capitolo 2 abbiamo studiato la costruzione delle scale e di conseguenza delle tonalità. Le scale e le tonalità, per costruzione, si basano sulla consonanza. Le dissonanze sono comunque possibili, ovviamente, e spesso sono utilizzate per effetti particolari. A cavallo fra l'ottocento e il novecento molti compositori ne hanno fatto abbondante uso. L'utilizzo delle dissonanze è arrivato al suo estremo con la teoria della dodecafonia di Arnlod Schönberg. Tale teoria è basata sull'equivalenza di tutte le 12 note di un'ottava e quindi abbandona completamente il concetto di scala e di tonalità. Non ci sono più le funzioni tonali e quindi gli accordi non hanno più lo stesso significato.

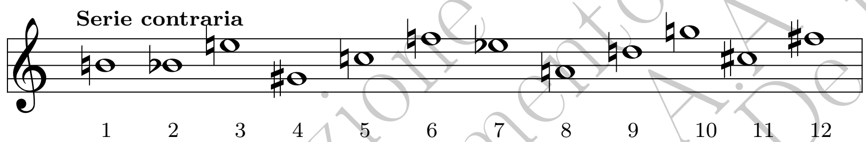
L'elemento fondamentale della musica dodecafonica diventa una sequenza di 12 note che comprende tutte le 12 note della scala cromatica. Tale sequenza viene detta *serie* nel gergo dodecafonico.

Figura 7.8: Spartito del duetto dello specchio

Una composizione dodecafonica parte da una specifica serie, che viene detta fondamentale, e si sviluppa tramite riproposizioni della serie in forme simmetriche. Dalla fondamentale si possono derivare altre 3 serie usando il moto retrogrado, il moto contrario e un moto retrogrado e contrario.

Nel moto retrogrado la serie viene riproposta a ritroso partendo dalla 12^a nota della serie fondamentale per arrivare alla prima:

Nel moto contrario la serie viene riproposta invertendo la direzione degli intervalli fra le note adiacenti: gli intervalli ascendenti diventeranno discendenti e quelli discendenti diventeranno ascendenti. Ad esempio, la serie fondamentale vista in precedenza inizia con un intervallo ascendente di seconda minore. Nella serie contraria diventerà un intervallo discedente di seconda minore.



Infine la serie retrogrado-contraria è la retrograda della serie contraria.

Come esempio di composizione dodecafonica consideriamo le variazioni op. 27 di Anton Webern

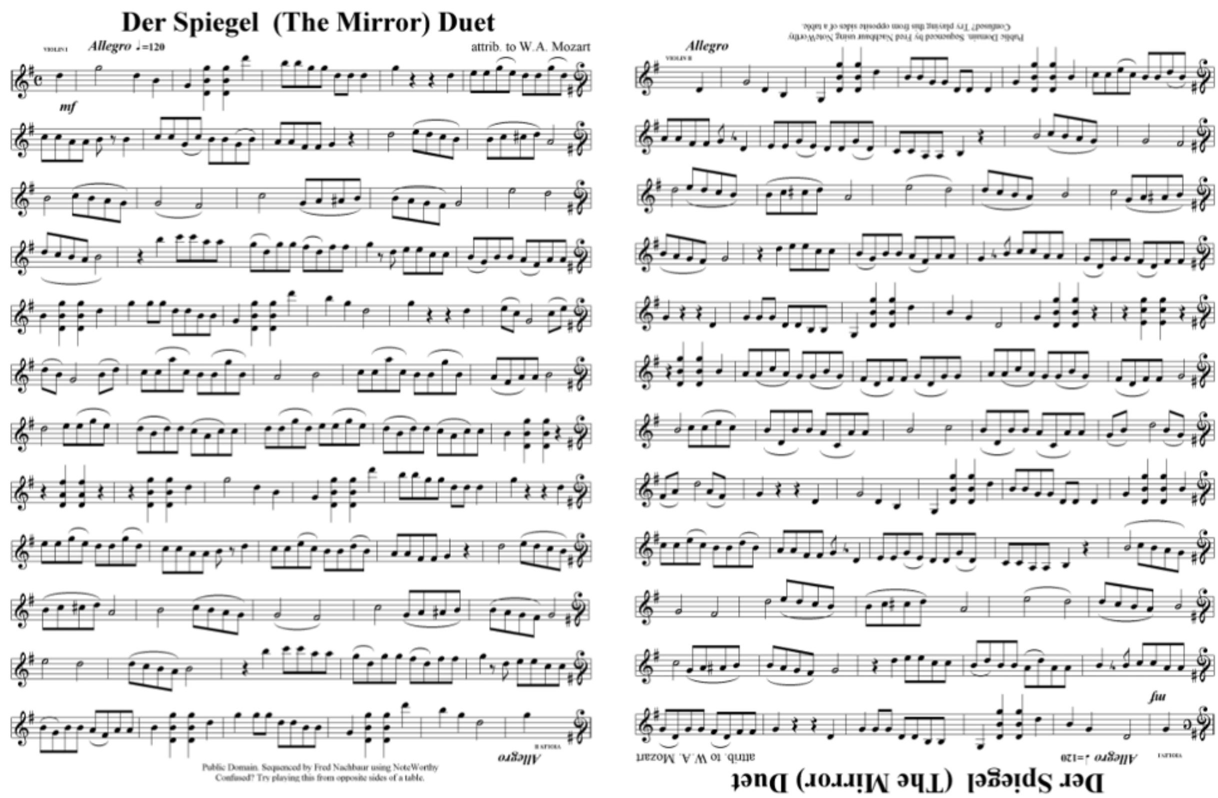


Figura 7.9: Spartito del duetto dello specchio, normale e capovolto

(1936). La Figura 7.10 mostra la prima parte della partitura originale.

La simmetria dovuta all'uso delle serie e delle loro trasformazioni può essere apprezzata più facilmente se usiamo una diversa rappresentazione grafica, come mostrato nelle Figure 7.11 Figure 7.12

7.4 Web links

- https://www.youtube.com/watch?v=YOqBTojB1Ag (Symmetry in Music - Two minut music theory)
- https://music.tutsplus.com/the-rule-of-three-and-music-audio-1389t
- https://www.youtube.com/watch?v=QcMYjR0vN_0 (mirror duet - Margareta Benkova)
- https://www.youtube.com/watch?v=nhBc96b2Yms (mirror duet)
- https://mathcs.holycross.edu/ tilde groberts/Courses/pastcourses.html
- https://www.youtube.com/watch?v=Y0_DeHSTLHU (Bach - Crab Canon)
- http://www.josleys.com/show_gallery.php?galid=349
- https://www.youtube.com/watch?v=A41CITk85jk&t=36s (Bach - Neverending canon)
- https://www.youtube.com/watch?v=JvNQLJ1_HQ0 (Pachelbel - Canone)

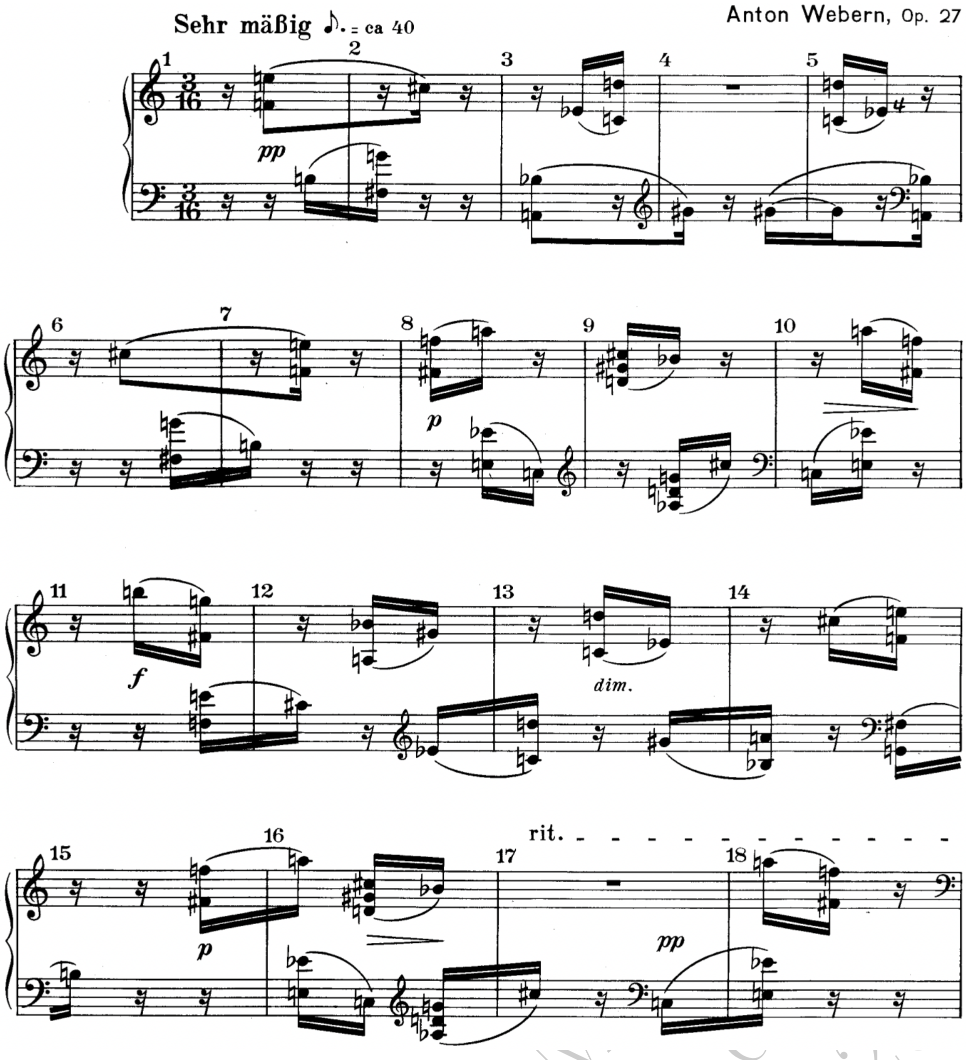


Figura 7.10: Prime battute della variazione n.1, op. 27 di Anton Webern

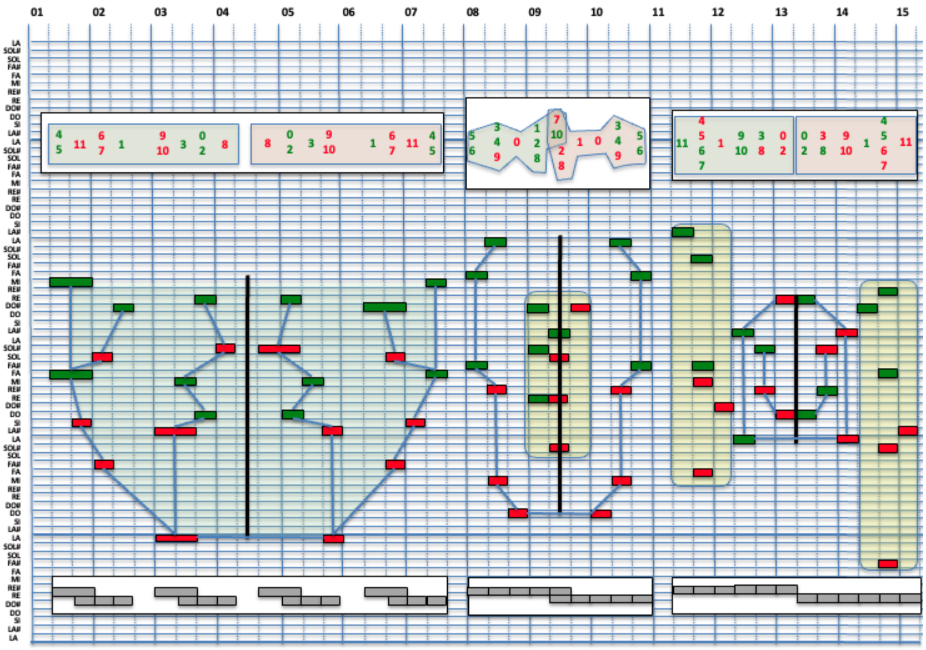


Figura 7.11: Rappresentazione alternativa delle prime 15 battute

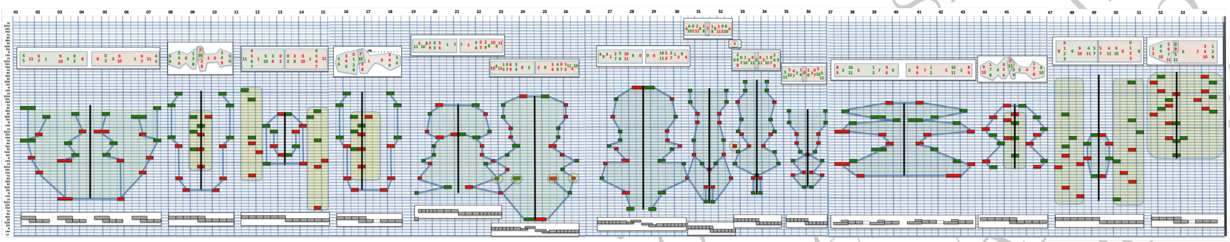


Figura 7.12: Rappresentazione alternativa intera variazione

Capitolo 8

Musica evolutiva

Gli algoritmi evolutivi sono algoritmi ispirati al processo evolutivo formalizzato da Darwin: una popolazione si evolve facendo sopravvivere gli individui che più si adattano al mondo in cui vivono. In questo contesto le (possibili) soluzioni a un problema sono equiparate a individui di una popolazione e l'algoritmo simula, in base a opportune regole, la loro evoluzione con l'obiettivo di cercare la soluzione (individuo) migliore. Tipicamente sono euristiche di ricerca in spazi molto grandi. In generale un algoritmo evolutivo parte da un popolazione iniziale, spesso generata casualmente, e simula l'evoluzione generando nuovi individui da quelli esistenti e operando un processo di selezione degli individui migliori; il processo termina quando si raggiunge un individuo/soluzione che soddisfa i criteri stabiliti oppure dopo un numero massimo di generazione. Esistono vari approcci che differiscono nel modo in cui le soluzioni/individui vengono rappresentate e nei dettagli dell'implementazione della simulazione dell'evoluzione. L'approccio più diffuso è quello degli *algoritmi genetici*.

8.1 Algoritimi genetici

Un algoritmo genetico è un particolare tipo di algoritmo evolutivo in cui gli individui/soluzioni sono rappresentati da stringhe (di simboli o più genericamente di bit) e l'evoluzione è simulata con operazioni di *ricombinazione* (corssover) e *mutazione*. La ricombinazione (corssover) è un'operazione di evoluzione che genera un nuovo individuo a partire da 2 individui esistenti. La mutazione, invece, è un'operazione che genera un nuovo individuo modificando un individuo esistente. La bontà di una soluzione/individuo viene misurata con una funzione detta di *fitness*; tale funzione fornisce una valutazione numerica.

Il processo evolutivo umano si basa sul trasferimento alla successive generazioni di informazioni relative al funzionamento e alla strutture delle cellule. Tale informazioni sono contenute nel DNA che è formato da segmenti detti *cromosomi*. Ogni cromosoma è costitutito da *geni* che sono le l'unità funzionali dell'eriditarietà, cioè sono i "mattoncini" che determinano il corredo eriditario di ogni individuo.

In analogia a tale processo evolutivo, nel gergo degli algoritmi genetici, si usa il termine *cromosoma* per indicare un individuo (o più precisamente la sua rappresentazione) e il termine *gene* per indicare i "pezzettini" che compongono il cromosoma. L'esatta definizione di geni e cromosomi, ovviamente, dipende dal problema: essi devono modellare le caratteristiche salienti che l'algorimo deve considerare per poter arrivare alla soluzione cercata.

Quindi il primo passo da fare è proprio questo: trovare una rappresentazione simbolica delle soluzioni. Tale rappresentazione sarà una stringa di bit o più genericamente di caratteri che nella sua interezza è il cromosoma. Il cromosoma è costituito da vari pezzetti, che sono i geni, come mostrato nella Figura 8.1.

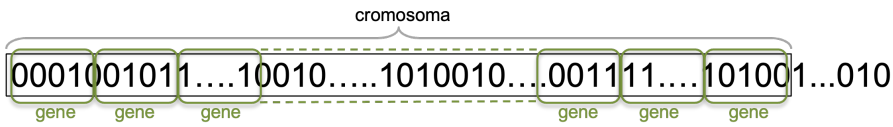


Figura 8.1: Rappresentazione generica di un cromomosa.

Il secondo passo è quello di definire le operazioni di ricombinazione e mutazione che operano sui cromosomi. Una tipica operazione di crossover spezza a metà i due cromosomi di input e ne crea un altro mettendo insieme la prima metà di un cromosoma con la seconda metà dell'altro cromosoma. Poichè ci sono 2 modi per fare questa cosa, l'operazione potrebbe o sceglierne uno a caso o creare due nuovi cromosomi. La Figura 8.2 mostra graficamente il funzionamento del crossover.

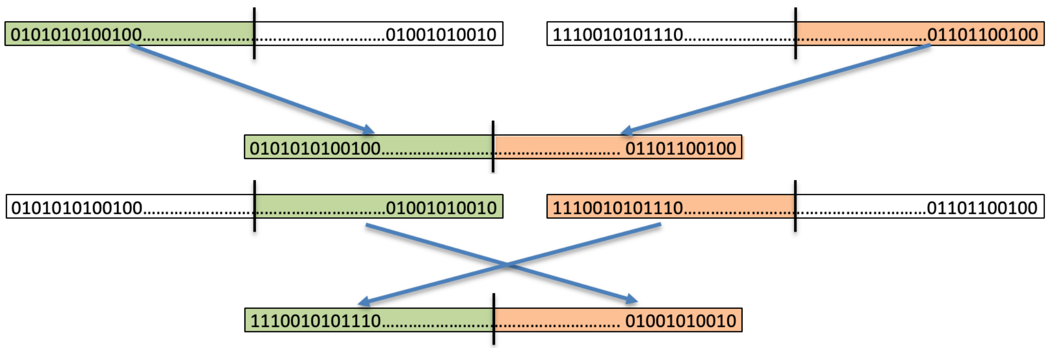


Figura 8.2: Crossover.

La mutazione, come dice il nome stesso, crea un nuovo cromosoma da quello di partenza, mutando alcuni geni, come illustrato nella Figura 8.3.

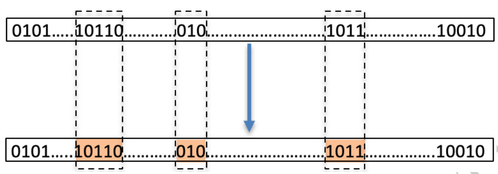


Figura 8.3: Mutazione.

Il terzo elemento che occorre è quello di una funzione di valutazione (fitness) che ci permette di selezionare i cromosomi da portare nelle nuove generazioni. Nella sua forma più semplice la funzione di fitness prende in input un cromosoma e fornisce un numero reale.

La generica struttura di un algoritmo genetico è mostrata nell'algoritmo GenericGA. Si parte da un popolazione iniziale di una detrminata grandezza fissata a priori. Quindi si ripete il processo evolutivo per un numero fissato di volte e ad ogni iterazione si generano nuovi cromosomi con le operazioni di crossover e di mutazione e si selezionano solo i migliori individui da portare nella prossima generazione.

Algorithm 5 GenericGA(MAX,SIZE,f)

```
P = setInitialPopulation()
for (i=1 to MAX)
    C = Applica il Crossover a tutte le coppie di P
    M = P Applica la Mutazione a tutti gli elementi di P
    P = i migliori SIZE elementi di P+C+M in funzione di f
endfor
Fornisci in output il miglior elemento di P
```

8.2 Music GA

Vediamo ora come usare gli algoritmi genetici per la creazione di musica. Stabiliamo prima di tutto uno specifico problema musicale. Ad esempio vogliamo creare dei brani di musica pop, composti da una linea melodica e un accompagnamento armonico (accordi). Supponiamo, per semplicità, di voler creare solo la una frase di 4 battute, con il tempo di 4/4. Dunque gli individui della popolazione sono frasi musicali di 4 battute in cui vengono specificati linea melodica e accordi.

Un gene potrebbe essere una frazione di una battuta, per semplicità possiamo ad esempio usare 2/4, cioè metà battuta. Il gene specifica sia l'accordo da usare sia il pezzettino di linea melodica all'interno del gene. La Figura 8.4 mostra un esempio.

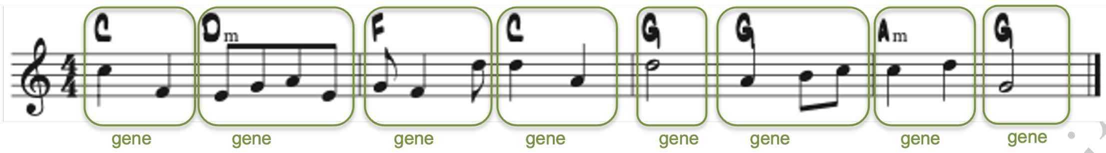


Figura 8.4: Cromosoma 1

La funzione di fitness può valutare la bontà musicale sia dei singoli geni, sia del modo in cui siincastrano fra di loro. E possiamo anche aggiungere valutazioni sull'intero cromosoma, o su parti specifiche. Ad esempio si può valutare se l'accordo di un gene è appropriato per le note della melodia inlcusa nel gene. Nella melodia si può valutare i salti melodici, sia all'interno di un singolo gene sia fra due geni successivi. All'interno di un gene si può valutare la proporzionalità delle durate delle note della melodia. Sull'aspetto armonico si può valutare la relazione fra 2 geni successivi (o anche più di 2) favorendo successioni armoniche più tipiche. Si può valutare la presenza di cadenze nella conclusione come pure l'accordo iniziale e quallo finale che tipicamente sono l'accordo di tonica.

Ricordandoci di quanto detto nel Capitolo 1, costruiamo una semplice funzione di fitness f. Valuteremo i seguenti aspetti.

- Siano $n_1, n_2, \ldots, n_k$ e C, rispettivamente, le note e l'accordo di un gene. Per ogni $i = 1, 2, \ldots, k$, se n_i appartiene a C la funzione f aumenta di 1 il punteggio di fitness mentre lo diminusice di 1 se non appartiene, a meno che la nota non sia preceduta e seguita da altre note con gradi congiunti (nel qual caso +1).
- Siano $n_1, n_2, \ldots, n_k$ le note di *tutta* la linea melodica. Per ogni salto melodico vietato la funzione f diminuisce di 3 il punteggio di fitness.
- Se due (o più accordi) corrispondono a una delle successioni tipiche allora la funzione f aumento di 3 il punteggio di fitness se la sequenza ricade nella categoria "spesso", 2 per la categoria "a volte", 0 per la categoria "raramente" e -3 per "mai" (vedi Tabella 1.4).

- Se il primo accordo è la tonica o la dominante il punteggio aumenta di 5.
- se l'ultimo accordo è la tonica il punteggio aumenta di 5.
- se gli ultimi 2 o 3 accordi formano una cadenza finale il punteggio aumenta di 10

Riprendendo l'esempio della Figura 8.4, la funzione di fitness assgenerebbe un punteggio di 6, derivante da:

- Gene 1: +1-1=0; gene 2: -1-1+2-1=-3; gene 3:-1+1-1=-1; gene 4: -1-1=-2; gene 5:+1; gene 6: -1+1+1=1; gene 7: +1-1=0; gene 8: +1. Totale: -3.
- 0 (non ci sono salti melodici vietati)
- I-ii: 0, ii-IV:+2; IV-I:+2; I-V:+3; V-vi:+2: vi-V:+2. Totale: 9.
- : 0
- : 0
- : 0

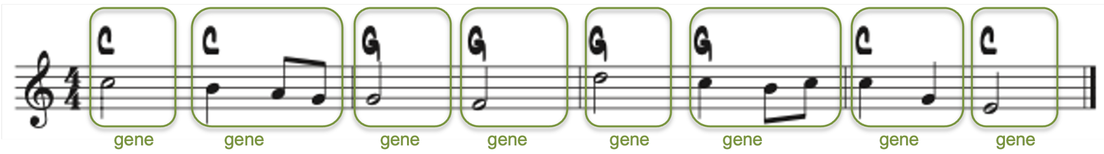


Figura 8.5: Cromosoma 2

La Figura 8.4 mostra un cromosoma con un fitness migliore, pari a 27.

- Gene 1: +1; gene 2: +1+1+1=+3 (gradi congiunti); gene 3:+1=+1; gene 4: -1=-1; gene 5:+1; gene 6: +1+1+1=3 (gradi congunti); gene 7: +1+1=2; gene 8: +1. Totale: 11.
- 0 (non ci sono salti melodici vietati)
- I-V:+3, V-I:+3; Totale: 6.
- : 5
- : 5
- : 0