# TECNICHE DI ELABORAZIONE SU STRINGHE APPLICATE A FRAMMENTI MUSICALI

Patrick Zanon
Donatella Imbrogno

24 settembre 2002

## 1 Introduzione

Nella maggior parte della musica che ascoltiamo, dalla musica tonale occidentale, barocca e classica, al jazz, circa dal "New Orleans" fino al BeBop, possiamo trovare ripetizioni e "regolarità" di varia natura. Il concetto di pattern, usato principalmente nell'ambito dello string matching, si adatta naturalmente all'analisi di queste regolarità in ambito musicale e può descrivere strutture ricorrenti a diversi livelli di astrazione, dalle altezze e le durate, alla dinamica e al timbro delle note, ai contorni e i movimenti e così via.
Gli ambiti in cui è necessario usare questa struttura vanno dalla composizione di musica al computer, all'analisi musicale, alla rielaborazione in tempo reale di materiale musicale. Esempi di pattern musicali sono i pattern melodici nella musica classica, sequenze mono o polifoniche di altezze e durate di note, i pattern armonici nel jazz tonale, sequenze di accordi, e pattern ritmici. Nella creazione di musica al computer questi pattern vengono usati come materiale di base per costruire parti dell'output del sistema di generazione e vengono perciò detti *pattern − base*.
È importante stabilire alcune caratteristiche che avranno poi conseguenze sugli schemi e le tecniche dell'estrazione dei pattern: il *contenuto musicale*, la *rappresentazione* e l'*organizzazione* in un pattern-base, e la *selezione*, l'*adattamento* e l'*uso* dei pattern. Il contenuto musicale varia da un sistema ad un'altro: le informazioni contenute nei pattern possono essere sequenze di intervalli, come frasi di note in formato MIDI. La natura di queste informazioni si riflette nella rappresentazione dei pattern (liste di note in formato MIDI, o di interi rappresentanti intervalli di semitoni) e nella organizzazione del pattern-base (lineare, gerarchica, ecc.). Per selezionare dei pattern dal materiale di base si possono usare diversi paradigmi, l'uso dei quali cambia la capacità del sistema di reagire ad un contesto in evoluzione, come nel caso di analisi di esecuzione in tempo reale. L'adattamento permette di riutilizzare i pattern in altri contesti, di solito con trasposizioni in semitoni. Infine l'uso finale dei pattern varia da sistema a sistema.
Uno degli scopi dell'analisi musicale fatta con l'uso di tecniche di string matching è capire e caratterizzare lo stile di compositori e improvvisatori. Un esempio è lo studio di Ruwet [26] su Debussy, tramite la ricerca dei "motivi", e della loro ripetizione con semplici variazioni. Sono stati fatti anche molti studi sull'improvvisazione di musicisti jazz, ad es. quello di Owens [17] su Charlie Parker. Egli estrasse da un corpus di 250 pezzi dell'autore, un lessico organizzato gerarchicamente di circa 200 pattern che vedeva come i blocchi costituenti gli assolo del sassofonista (vedi figura 1).
Alcuni sistemi di analisi musicale tramite pattern (EMI) [10], identificano pattern stilistici, "signatures", in campioni di date composizioni tonali. Altri cercano ripetizioni di pattern automaticamente,

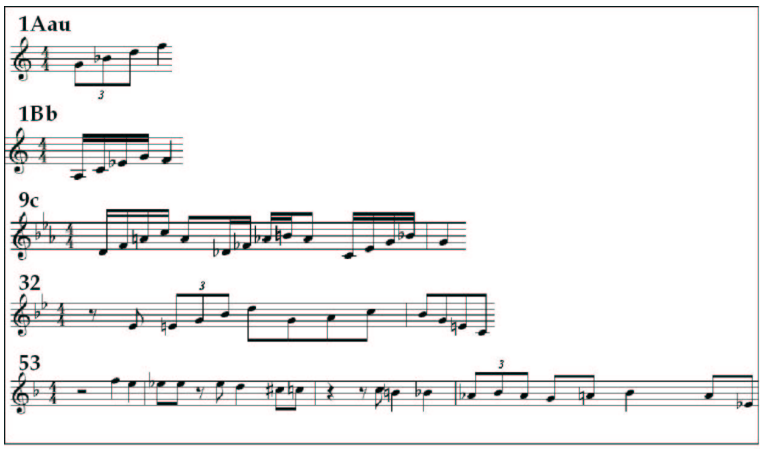


Figura 1: Alcuni dei pattern di Charlie Parker tratti dal lessico di Owens (1974)

dando come predefinite un certo numero di variazioni ritmiche o melodiche (Stech) [29], infine Imprology [20], [21] studia l'improvvisazione nel jazz.
L'origine degli studi sul "pattern matching" si colloca agli inizi degli anni '60, per affrontare le problematiche connesse alla progettazione di compilatori. Da allora le applicazioni si sono espanse in un gran numero di campi di ricerca, dalla biologia fino ad arrivare alla musica. Una bibliografia completa potrebbe contenere più di 500 articoli [1].
Qui parleremo dei problemi connessi alla rappresentazione musicale, in particolare dell'altezza e del ritmo. In seguito del modo in cui si portano avanti i confronti tra frammenti musicali, con le relative implementazioni del matching esatto, $\delta$-approssimato e dell'editing dei pattern. Poi vedremo come si può affrontare la rappresentazione delle opere polifoniche, infine tratteremo l'induzione, o estrazione di pattern.

## 2 Particolarità musicali

Nell'applicare algoritmi di pattern matching a stringhe musicali sorgono questioni collegate alla particolare natura degli elementi musicali. Per esempio, a un livello più basso di rappresentazione una melodia potrebbe essere data con stringhe di coppie di valori (altezza e durata) oppure i differenti parametri potrebbero essere trattati in stringhe separate.
La "melodic surface", cioè l'insieme dei simboli che rappresentano una esecuzione (eventi nota nel caso del MIDI), potrebbe differentemente essere rappresentata come una stringa di pitch assoluti, o classi di pitch, o pitch in relazione ad un centro tonale o ancora come intervalli di pitch. Le stringhe relative al ritmo potrebbero contenere durate, o rapporti fra durate.
Un esempio dei diversi possibili approcci nell'uso degli algoritmi su stringhe è illustrato sinteticamente nella tavola degli algoritmi nella sezione 7, in cui si vedono contemporaneamente le rappresentazioni dell'altezza e del ritmo scelte per l'applicazione di un particolare algoritmo.
Sono molte le rappresentazioni possibili, sia per quanto riguarda l'altezza che il ritmo. Queste dovrebbero essere assimilate a stringhe, pur tenendo conto delle specifiche necessità della rappresentazione musicale. In particolare l'operazione di trasposizione di pezzi dovrebbe essere una trasformazione invariante, ovvero due brani uguali suonati in due diverse tonalità dovrebbero risultare

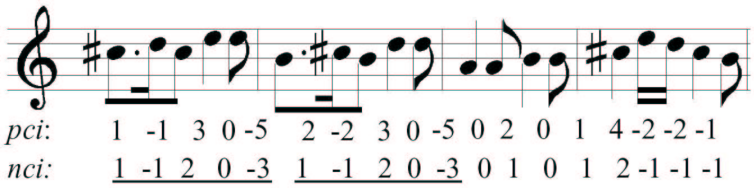


Figura 2: Inizio del tema della Sonata in La maggiore di Mozart. Si nota come il "pci" non riesce ad individuare il pattern comune tra le battute 1 e 2. L'uso dell' "nci" consente di rilevare un pattern comune. Tuttavia la rappresentazione in termini di pitch relativo riesce ad individuare due sequenze che però a livello di pitch assoluto risultano essere sovrapposte per una nota (la prima della seconda battuta).

"in un certo senso" uguali agli occhi dell'algoritmo di matching. Ciò induce una sorte di classe di equivalenza per i confronti tra stringhe. Inoltre, un ascoltatore è in grado di riconoscere un pezzo, anche se viene suonato con tempi differenti, e perciò anche il cambio di tempo globale dovrebbe essere una trasformazione invariante.
Proprio queste trasformazioni invarianti vengono usate nella Musical Information Retrieval, in cui si ha a che fare con un gran numero di performances registrate con tempi e tonalità diversi. Il sistema MIR dovrebbe essere capace di riconoscere tutte queste performances come un unico pezzo e darne una descrizione univoca. In altri ambiti (analisi dell'espressività) questi tempi differenti non possono essere trascurati, anzi vengono usati insieme ad altre informazioni, quali ad esempio le micro-deviazioni, per analizzare ed eventualmente classificare performances secondo differenti tipi di intenzioni espressive.

## Rappresentazione del pitch

Nella tradizione occidentale l'altezza è di solito rappresentata o col nome della nota nel sistema tradizionale (es.: Fa♯-Sol♯-La), o come altezza assoluta (es. nel MIDI: 66, 68, 69). Gran parte delle applicazioni musicali assistite dal computer adottano la rappresentazione col pitch assoluto. Questa però non è adatta per le applicazioni nella musica tonale, dato che ignora le gerarchie della scala diatonica. Infatti la discretizzazione nel campo delle altezze che opera il MIDI, con la divisione di un'ottava in 12 intervalli identici, annulla la sensazione di tonalità , rendendo equivalenti toni inarmonici che hanno qualità tonali differenti. In altre parole, note la cui distanza armonica non è grande (es. la tonica e la dominante) verrebbero trattate come note molto lontane nel contesto armonico (es. la tonica e la sensibile). Inoltre la ciclicità dell'armonia verrebbe trascurata, per cui note la cui distanza è multipla di un'ottava non verrebbero considerate così uguali quanto appaiono ad un ascoltatore.
Un altro rilevante problema che si ha applicando algoritmi di pattern processing su stringhe con pitch assoluto è che non riesce a riconoscere la trasposizione, mentre vorremmo che la valutazione di similarità fosse insensibile alle trasposizioni (vedi fig.9 tra i pattern $d$ ed $e$).
Infine, dovremmo porre attenzione su come considerare le trasformazioni di "cancellazione" (o "inserimento") usate di solito nelle tecniche di matching approssimato di pattern: la cancellazione di un pitch assoluto ha effetti molto simili alla cancellazione di un simbolo in una stringa standard, mentre altri tipi di rappresentazione complicano di molto la situazione. Ci piacerebbe mantenere questo tipo di similarità, o, pur introducendo altre rappresentazioni, non portare troppi cambia-

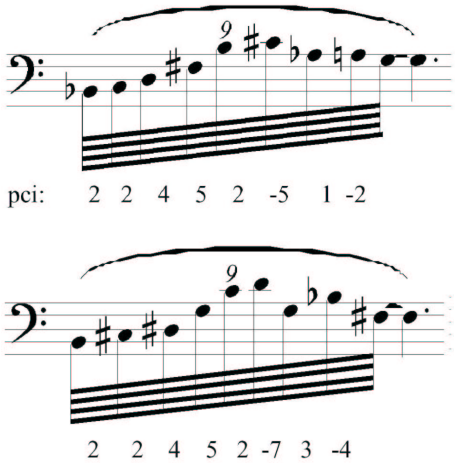


Figura 3: Da un motivo di Messiaen *Vingt Regards sur l'Enfant Jesus* (L'echange). Se rappresentiamo i pattern con l'altezza assoluta non si osserva nessuna coincidenza, invece col "pci" i primi cinque intervalli coincidono.

menti.
La soluzione ovvia a questi problemi è l'uso di un'altezza relativa, soprattutto con la derivazione di intervalli di pitch dalla "surface" del pitch assoluto. Comunque ci sono differenti soluzioni. Se definiamo come "pci" (pitch class intervals) il numero di semitoni fra la prima e la seconda nota, e come "nci" (name-class intervals) gli intervalli nella scala diatonica corrispondente, si vede nella figura 2 che col "pci" non si riesce a rilevare la ripetizione che ha luogo nelle prime due battute, mentre l' "nci" è più adeguato. Poiché nella musica tonale ci sono diversi tipi di scale (non solo la diatonica), Cambouropoulos [7] ha proposto una rappresentazione a intervallo di pitch generale (GPIR), con cui si possono mettere in codice intervalli secondo i più significativi tipi di scale di un dato idioma musicale.
Con queste rappresentazioni relative sorgono altri problemi. Nel caso di elaborazioni approssimate di pattern, la cancellazione di un pitch o di un intervallo tra pitch ha effetti piuttosto differenti sulla sequenza musicale risultante. Per esempio, consideriamo il frammento in figura 2: se dalla stringa che rappresenta la melodia secondo il pitcht assoluto viene cancellata la seconda nota della prima battuta, si ottiene un pattern non molto diverso: Do♯-Do♯-Mi-Mi; se invece dalla stringa di intervalli tra picht viene cancellato il secondo intervallo si ha un cambiamento molto più forte: Do♯-Re-Fa♯-Fa♯.
Nel caso di tecniche di induzione di pattern (sez. 6) sorge questo problema: se in una stringa di intervalli di pitch i successivi pattern contigui non si sovrappongono, nella corrispondente stringa di pitch invece si sovrappongono (per un singolo pitch). Per esempio se applichiamo alla stringa "nci" di figura 2 un algoritmo di induzione di pattern che cerchi di trovare una soluzione senza sovrapposizioni - per esempio usando metodi a minima lunghezza di descrizione - allora compare il pattern sottolineato; tuttavia se consideriamo i pitch assoluti questi due pattern si sovrappongono per una sola nota. Questa rappresentazione a pattern sovrapposti non è plausibile negli algoritmi di induzione, perciò se volessimo descrivere una melodia con pattern di pitch assoluti non sovrapponentesi, dovremmo imporre il vincolo aggiuntivo che i pattern di intervalli trovati debbano essere disgiunti, cioè separati da almeno un altro intervallo.
Molto spesso, data la complessità degli algoritmi per il matching approssimato (vedi sezione 3), si

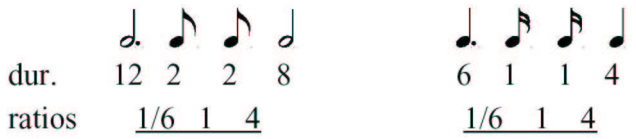


Figura 4: I pattern ritmici coincidono a livello dei rapporti di durate.

cerca di utilizzare gli algoritmi di matching in modo diverso: si classificano gli intervalli in un certo numero di classi a seconda della loro lunghezza (es. $repeat : nci = 0, step : nci = 1, leap : nci > 1$ ) e si costruisce una stringa che rappresenta la melodia a partire dall'alfabeto {-l,-s,r,+s,+l}, oppure usando un criterio ancora più ristretto contenente i soli segni degli intervalli: in questo caso il profilo, o contorno melodico, viene inteso come direzione ascendente, discendente, o orizzontale dell'intervallo, ed è rappresentato da una stringa i cui simboli appartengono all'alfabeto {-,+,=}.
Nell'esempio di figura 3, se i pattern vengono rappresentati con l'altezza assoluta, non c'è nessuna interessante coincidenza; se invece vengono codificati con intervalli di pitch di semitoni, allora si vede come i primi cinque intervalli coincidano. Se codificati come stringhe "step-leap" del primo alfabeto, allora le prime sette note coincidono. Infine usando solo l'informazione della direzione dell'intervallo, come nel caso del secondo alfabeto, allora tutte le note coincidono.

### Rappresentazione del ritmo

La rappresentazione del tempo nelle stringhe musicali presenta gli stessi problemi della rappresentazione dell'altezza. L'idea è che un ascoltatore può riconoscere il ritmo anche se un pezzo viene suonato con un tempo diverso: questo significa che siamo sensibili ai rapporti tra durate, e non alle durate assolute di tempo (vedi figura 4). In questo caso, anzichè indicare la durata effettiva della nota, si usa il rapporto della durata della nota corrente con la precedente. Questa soluzione è affetta dallo stesso problema riscontrato per la rappresentazione relativa del pitch, ovvero si ha la sovrapposizione dei pattern a livello di durata assoluta.

## 3 Paragoni fra sequenze musicali

Nel confronto tra sequenze musicali siamo interessati a dare una valutazione della similitudine tra due sequenze prese nella loro interezza. A questo scopo vi sono tre aspetti importanti da considerare: la possibilità di permettere delle differenze, *globali* e/o *locali*, il paradigma di valutazione, e le tecniche di ricerca.
Con degli esempi mostreremo come si possono confrontare sequenze piuttosto diverse, predefinendo delle differenze, che possono essere viste in termini di *ridescrizione* o *trasformazione* di una sequenza in un'altra. Abbiamo poi bisogno di sapere quanto le due sequenze si somigliano, cioè di conoscere il paradigma di valutazione, e questa rappresentazione di similarità può essere data in forma quantitativa, (in forma Booleana, o come misura graduale tra 0 e 1), o qualitativamente, come con la *trace* (vedi figura 6) che dà una rappresentazione grafica delle trasformazioni necessarie per confrontare le due sequenze. Infine è necessario stabilire le tecniche di ricerca più appropriate, quando si devono considerare diverse possibilità per stabilire la somiglianza fra due sequenze: la più nota e più documentata è la programmazione dinamica.

Il confronto esatto consiste nel determinare quando due sequenze sono strettamente identiche a livello della "surface" musicale, e richiede che esse abbiano la stessa lunghezza. Il paradigma

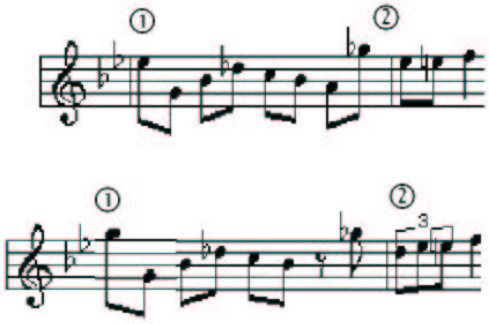

Figura 5: Due frammenti simili di improvvisazione jazz.

di valutazione è Booleano. Per le applicazioni musicali però sono più utili approcci tolleranti alle differenze. Come abbiamo visto nella precedente sezione possiamo confrontare, tramite delle *ridescrizioni*, sequenze musicali che non siano esattamente uguali, per esempio tramite l'uso di successioni di intervalli di semitoni o successioni di direzioni di intervalli (ascendenti, discendenti o orizzontali). Un altro modo per confrontare sequenze *globalmente* differenti, è introdurre delle *trasformazioni*, come le tradizionali retrograde e le inversioni applicate alle dimensioni dell'altezza e del tempo (un esempio di retrogrado si ha nella figura 7, e si può intendere come una trasformazione simmetrica rispetto ad un asse verticale, mentre l'inversione è la trasformazione simmetrica rispetto ad un asse orizzontale). In entrambi i casi lo schema di valutazione è Booleano, e rimane il vincolo che le sequenze debbano avere la stessa lunghezza. Questi approcci però non riescono a tener conto delle variazioni musicali o di ornamenti, che consistono di solito in "distorsioni" locali. La figura 5 mostra un esempio di frammenti di improvvisazione jazz, la cui somiglianza non si può vedere in termini di corrispondenze globali. Il confronto fra i due frammenti può essere fatto in termini di trasformazioni o "edits": si omette il La, si traspone il Mi di una terza maggiore, si cambiano le durate dei Mi e Mib della seconda battuta, si aggiunge Re nella seconda battuta.
Le trasformazioni usate per andare dal primo al secondo frammento sono: *inserzioni*, *cancellazioni* e *sostituzioni*. In figura 6 la *trace* dà una rappresentazione grafica delle trasformazioni fatte per confrontare le due sequenze: la cancellazione del La con una croce, l'inserzione del Re nella seconda battuta con un'apice, le sostituzioni con la linea più chiara e le trasformazioni d'identità con la linea nera. Da questo esempio si vede come il numero di elementi delle sequenze in questo tipo di approccio può essere diverso.
La cosa più importante è che possiamo assegnare a ciascuna trasformazione operata una *funzione* di *contributo* numerica, così da sapere quanto ognuna contribuisce alla similarità complessiva delle due sequenze. Lo schema di valutazione è dato dalla *somma* dei contributi individuali di tutte le trasformazioni. Per esempio i contributi delle cancellazioni prendono valori negativi, mentre le trasformazioni di identità (un elemento viene sostituito nella seconda sequenza da uno identico), assumono valori strettamente positivi. Per quanto riguarda la tecnica di ricerca, è data dalla programmazione dinamica, che ottimizza la valutazione di similarità con una ricerca ricorsiva.

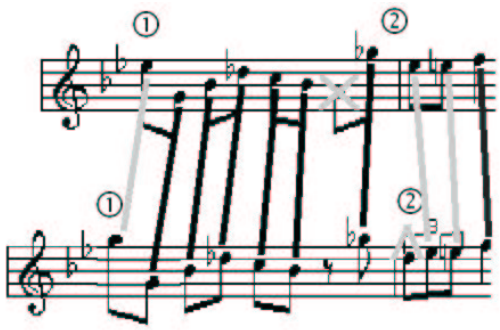

Figura 6: Rappresentazione grafica della similitudine (*trace*) fra i due frammenti.

### 3.1 Alcune definizioni per il matching approssimato

Un modo per formalizzare la similitudine fra stringhe musicali strettamente collegate ma non identiche, è fare uso della definizione di matching $\delta$-approssimato. In questo tipo di confronto, pattern di eguale lunghezza consistenti di interi vengono considerati coincidenti se ciascun paio di interi corrispondenti differiscono per non più di un $\delta$. Per esempio un arpeggio maggiore ascendente [+4,+3,+5] ed una sequenza di arpeggio minore [+3,+4,+5] possono essere considerati coincidenti se nel processo di matching è consentita una tolleranza $\delta = 1$. In particolare, sia $\Sigma$ un alfabeto di interi. Diciamo che due simboli $a, b$ sono $\delta$-approssimati,

$$a =_{\delta} b \tag{1}$$

se e solo se

$$|a - b| \leq \delta \tag{2}$$

dove $\delta$ è un intero dato. Diciamo che due stringhe $x, y$ sono $\delta$-approssimate,

$$x \stackrel{\delta}{=} y, \tag{3}$$

se e solo se

$$|x| = |y|, \text{e } x_i =_{\delta} y_i, \forall i \in \{1...|x|\} \tag{4}$$

per un dato $\delta$, dove con $|x|$ si indica la lunghezza (cardinalità) della stringa.
La tolleranza totale $\delta$ consentita per il confronto tra stringhe può essere vincolata da un ulteriore parametro di tolleranza $\gamma$, dando così luogo ad un matching ($\delta$,$\gamma$)-approssimato. Diciamo che due stringhe $x, y$ sono $\gamma$-approssimate,

$$x \stackrel{\gamma}{=} y, \tag{5}$$

se e solo se

$$|x| = |y|, \text{ e } \sum_{1}^{|x|} |x_i - y_i| < \gamma \tag{6}$$

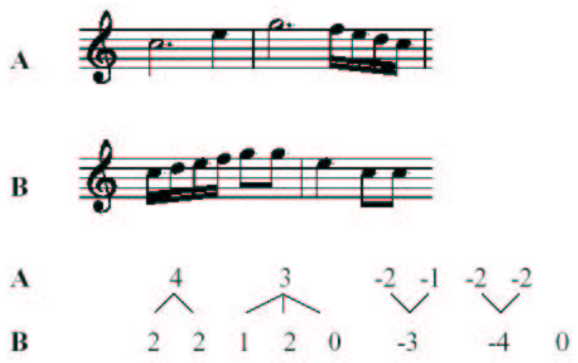


Figura 7: Dalla *Prima Sinfonia* di Beethoven: tema dell'Allegro (A) e tema del Finale (B). Il pattern B è anche il retrogrado del pattern (A).

per un dato numero $\gamma$.
Infine diciamo che due stringhe $x, y$ sono $\{\gamma,\delta\}$ -approssimate,

$$x \overset{\gamma,\delta}{=} y, \tag{7}$$

se e solo se $x$ e $y$ soddisfano alle condizioni 4 e 6. Sono stati sviluppati efficienti algoritmi per questi diversi casi da E. Cambouropoulos [8].

### 3.2 Un esempio di pattern editing

Gli algoritmi di matching $\delta$-approssimato vanno bene solo per pattern di eguale lunghezza. Una tecnica di composizione musicale consiste nell'operazione di inserzione e nell'operazione di cancellazione di singole note in un pattern (vedi figura 6). Aggiungere una nota tra due può essere

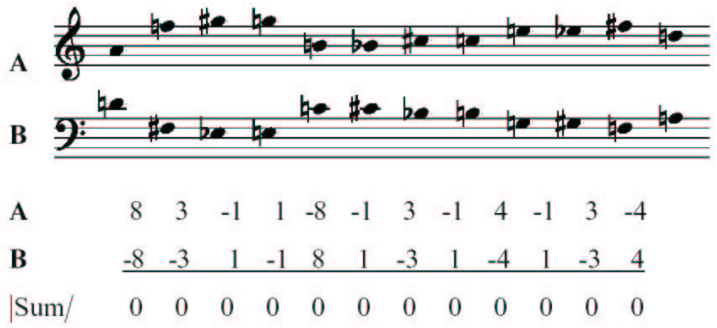


Figura 8: Originale ed Inversione di una serie dodecafonica della *Cantata No.*1, *Op.*29 di Webern.

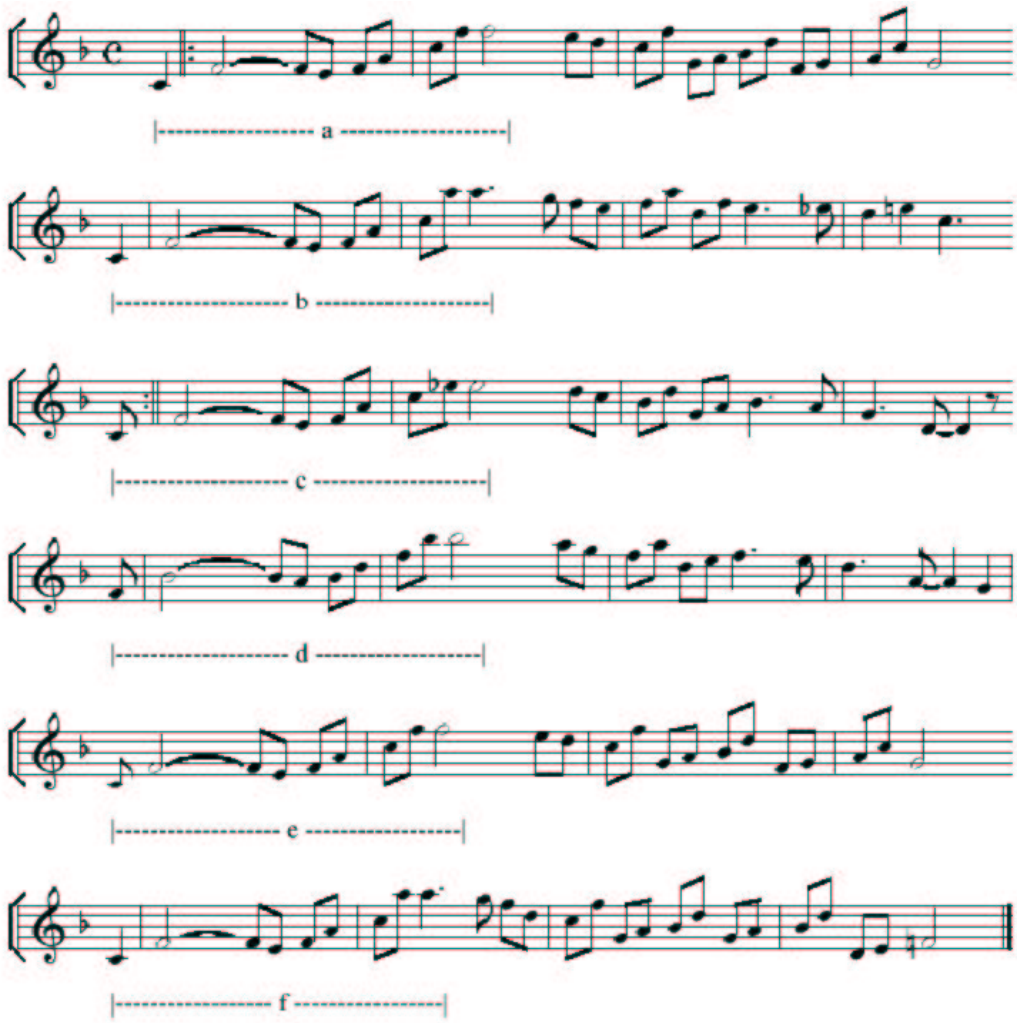


Figura 9: Dal *Traümerei* di Schumann. Questa melodia può essere rappresentata con una stringa di intervalli di pitcht (in numero di semitoni). Pitch Interval Pattern:{5,-1,1,4,3,5,0}. Pitch Interval String: {$\underline{5,-1,1,4,3,5,0}$ -1,-2,-2,5,-10,2,1,4,-9,2,2,3,-5,-7,$\underline{5,-1,1,4,3,9,0}$ -2,-2,-1,1,4,-7,3,-1,-1,-1,2,-4,-12,$\underline{5,-1,1,4,3,3,0,}$1,-1,-2,4,-7,2,1,-1,-2,-5,3,$\underline{5,-1,1,4,3,5,0,}$1,-2,-2,4,-7,2,1,-1,-2,-5,-2,-7,$\underline{5,-1,1,4,3,5,0,}$1,-2,-2,5,-10,2,1,4,-9,2,2,3,-5,-7,$\underline{5,-1,1,4,3,9,0}$ -2,-2,-3,-2,5,-10,2,1,4,-7,2,1,4,-12,2,1}. Se si usa il matching esatto, vengono trovate tre istanze identiche nel pattern di ricerca (i pattern $a$, $d$ ed $e$); le altre tre istanze non coincidono. Se si impiega il matching $\delta$-approssimato per $\delta$=2, vengono trovate altre quattro istanze ($a$, $c$, $d$ ed $e$); per $\delta$=4 tutte le sei istanze evidenziate coincidono. Si può usare il vincolo aggiuntivo del matching $\gamma$-approssimato per valori elevati di $\delta$ e per corpi melodici ampi.

interpretato come dividere l'intervallo d'altezza iniziale in due successivi intervalli la cui somma è uguale all'intervallo iniziale. Ad esempio, se la sequenza iniziale è data da 60, 62 (intervallo:+2) e la sequenza con una nota aggiunta sia 60, 67, 62 (intervalli:+7, -5), allora la somma dei due intervalli risultanti deve essere uguale all'intervallo iniziale.
Questa proprietà può essere usata per confrontare sequenze di diversa lunghezza in questo modo: un intervallo di una stringa può essere messo in relazione con due o più intervalli successivi dell'altra, la cui somma è uguale (o $\delta$-approssimata) all'intervallo iniziale (vedi fig.7 e fig.9).

# 4 La polifonia

Molto spesso, nella Computer Music e nelle applicazioni MIR, si ha a che fare con dei brani polifonici. Questi possono essere rappresentati in due modi, sia con un grafico in due dimensioni (pitch contro il tempo), sia come un insieme di stringhe monodimensionali. Nel primo caso l'informazione su quali siano e come siano composte le varie linee melodiche e di accompagnamento è assente, perciò bisogna usare algoritmi particolari per estrarre pattern in uno spazio bidimensionale. Questi algoritmi sono molto utili in quanto i più comuni database musicali contengono semplici file MIDI non strutturati. Inoltre essi consentono la ricerca di strutture polifoniche piuttosto che di semplici pattern melodici. Un problema potenziale è che se la "search query", per esempio melodica, non è lunga abbastanza o contiene salti d'altezza ampi, qualsiasi algoritmo darà delle risposte musicalmente o percettivamente implausibili.
La rappresentazione in stringhe monodimensionali richiede sofisticati algoritmi di "streaming", vale a dire algoritmi che possono dividere l'opera polifonica in stream indipendenti (linee melodiche) che siano significativi in sé. Questo non è un compito banale. D'altra parte lo sviluppo di tali algoritmi può essere molto utile per la preparazione di dati musicali per la successiva elaborazione dei pattern. Cambouropoulos ha proposto un algoritmo di questo tipo [5]. L'algoritmo di "streaming" è basato principio Gestalt della prossimità, e cerca semplicemente di trovare il percorso più breve che connette tutti gli onset con una battuta (vedi figura 10). Non è permesso l'incrociarsi dei flussi. Il numero di flussi è sempre uguale al numero di note nell'accordo più ampio. In questo caso è possibile usare algoritmi di matching standard.

# 5 Strutture gerarchiche e matching

Il confronto esatto tra pattern è diretto a cercare istanze in pattern dati (o per indurre pattern identici), ma può essere usato ugualmente per rivelare o stabilire somiglianze tra pattern differenti. Ci chiediamo dunque quale sia il metodo di confronto tra pattern più adeguato quando cerchiamo di stabilire similarità fra entità complesse quali dei passaggi melodici.
Semplificando, sono possibili principalmente due approcci: a) il pattern-matching approssimato applicato sulla superficie musicale non strutturata e, b) il pattern-matching esatto applicato sulla superficie musicale e su un numero di versioni ridotte di essa, che contengono componenti strutturalmente più importanti.
Il presupposto del primo approccio è che dei segmenti musicali, essendo simili, abbiano alcuni dei loro elementi componenti uguali. Ad esempio, due istanze di un motivo melodico hanno una significativa quantità di note uguali. Il secondo approccio è basato sull'ipotesi che segmenti musicali simili siano necessariamente identici in almeno un profilo parametrico della riduzione della sua superficie. Per poter applicare il secondo metodo di pattern matching, è richiesta una parte rilevante di pre-elaborazione, per esempio i segmenti melodici non vengono esaminati solo a livello della superficie,

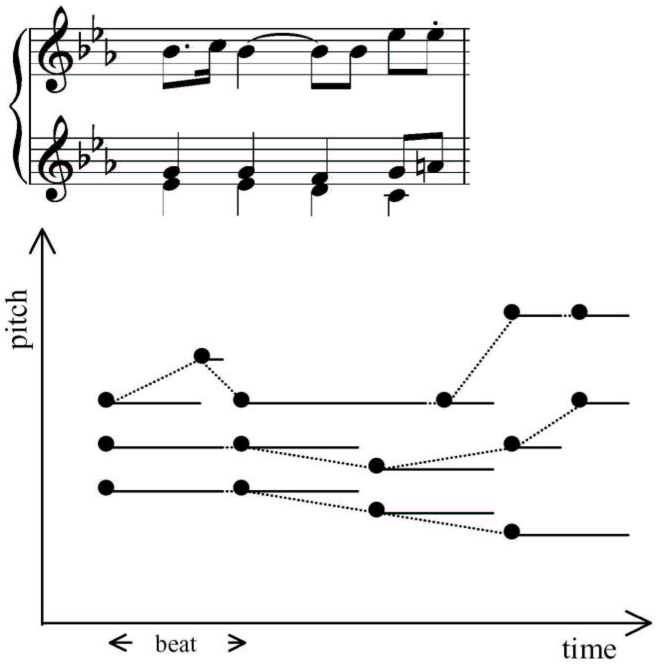


Figura 10: Applicazione dell'algoritmo di "streaming" all'inizio della Sonata KV282 di Mozart. I punti nel grafico rappresentano gli onset delle note del segmento musicale; le linee punteggiate mostrano i tre flussi individuati dall'algoritmo; le linee orizzontali indicano gli intervalli tra gli onset per ciascun flusso.

ma vengono costruiti vari livelli astratti di rappresentazione che riflettano le proprietà strutturali dei segmenti melodici (per es. note più lunghe, note metricamente più forti, note tonalmente importanti, ecc.). In questo modo è possibile costruire una gerarchia strutturata che consente di distinguere, alla stessa stregua del discorso verbale, le frasi musicali, le "parole" che le costituiscono e le "sillabe" di tali parole con i relativi accenti.
I risultati confermano che le tecniche di confronto approssimato non riescono a discriminare situazioni in cui un musicista riconoscerebbe immediatamente le note più importanti.

## 6 Induzione di pattern

L'estrazione o induzione di pattern è una forma di sapere non assistito dal punto di vista dell'intelligenza artificiale. Più in particolare l'estrazione di pattern può essere definita come la costruzione (o l'arricchimento) di un insieme di pattern costruito a partire da materiale musicale dato.
Una delle differenze tra le tecniche di confronto tra pattern e quelle di estrazione di pattern, è che quest'ultima richiede una nozione di "rilevanza" del pattern. Nel confronto tra pattern non si incontra questo problema perché la "search query" è data; l'utilizzatore decide "a priori" che un certo pattern è importante, e dopo vengono individuati tutti i confronti in una stringa o in un set di stringhe.
Invece nell'estrazione di pattern bisogna decidere che tipo di pattern l'algoritmo dovrebbe cercare, nel senso che non è utile cercare qualsiasi tipo di pattern. Selezionare pattern "significativi" può essere fatto o dopo aver cercato tutti i pattern (che non è di solito l'approccio più efficiente), oppure

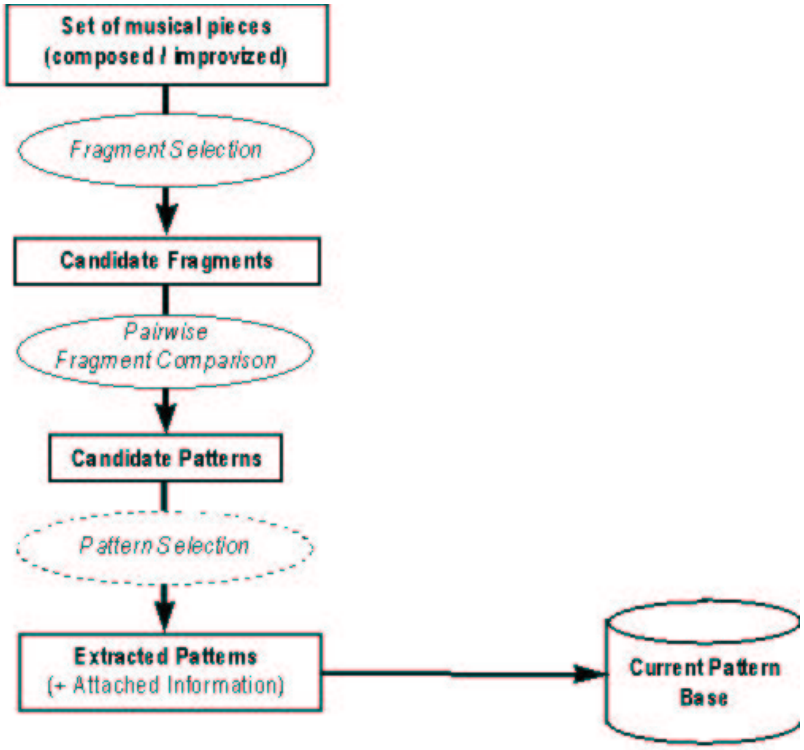


Figura 11: Schema di elaborazione di un estrattore off-line di pattern

prima, forzando l'algoritmo a fermarsi quando viene trovato lo specifico tipo di pattern.
Nel primo approccio vengono estratti tutti i pattern, attaccando un valore di rilievo a ciascuno dei pattern trovati, basandosi su questi fattori: a) si preferiscono pattern più lunghi, b) si preferiscono i pattern che ricorrono con più frequenza, c) si evitano sovrapposizioni. Una *funzione* di *selezione* che calcola un indice di importanza numerica per ciascun pattern secondo questi principi, potrebbe essere per esempio:

$$f(L, F, DOL) = F^a \cdot L^b / 10^{c \cdot DOL} \tag{8}$$

dove L è la lunghezza del pattern, F è la frequenza con cui ricorre, DOL è il grado di sovrapposizione e $a, b, c$ sono pesi relativi ai principi di cui sopra. Per ciascun pattern scoperto con l'algoritmo esatto di induzione, viene calcolato un valore per la funzione di selezione. I pattern che ottengono i valori più elevati dovrebbero essere i più importanti.
Nel secondo approccio, la determinazione del pattern più significativo viene fatta in anticipo, così da consentire all'algoritmo di fermarsi quando si verifica l'appropriato "criterio di importanza". Che tipo di pattern sono "significativi" per l'estrazione musicale? Un tipo interessante di pattern può essere collegato alle ripetizioni immediate (almeno 2, "square", o più ripetizioni consecutive). Il tipo di pattern più appropriato per cercare queste ripetizioni è il *periodo*, che è una ripetizione successiva all'interno di una stringa (ad esempio *abc* in *abcabcabcab*). Questo viene usato nel dominio del pitch assoluto, e nel dominio degli intervalli di tempo tra una nota e la successiva (intervalli tra gli onset). Nel dominio degli intervalli d'altezza (e in quello dei rapporti tra intervalli fra onset), bisogna trovare altri tipi di pattern per la ricerca delle ripetizioni immediate. Chiamiamo questo tipo di pattern periodo disgiunto, che è un pattern che si ripete, separato da singoli simboli. Per esempio *abc* è un periodo disgiunto di *abcdabcaabcbabc*. Questi simboli di separazione (intervalli)

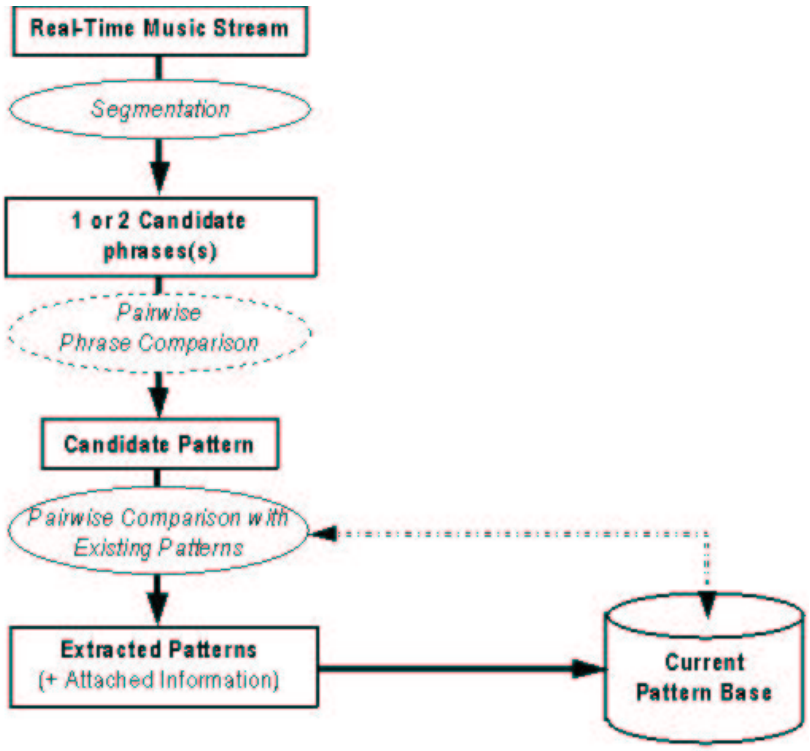


Figura 12: Schema di elaborazione per un estrattore di pattern online

sono necessari se ci si aspetta che i pattern consecutivi d'altezza non si sovrappongano (vedi sezione 2).
Gli algoritmi di induzione dei pattern vengono anche divisi in algoritmi $off - line$ ed algoritmi *online*. Ci sono alcune differenze in questi due approcci nelle varie fasi dei procedimenti di estrazione: il primo viene usato quando il sistema ha l'intero materiale già a disposizione, come nel caso della Musical Information Retrieval, mentre il secondo approccio viene usato in un contesto in tempo reale, quando il sistema deve analizzare il materiale musicale nello stesso momento in cui viene realizzata la performance. Essenzialmente nel primo approccio (off-line) i pattern vengono estratti selezionandoli da un insieme ricavato dalla registrazione in analisi. Nel secondo approccio si cerca di far uso delle elaborazioni già fatte in precedenza, per cui da una prima selezione fatta sui pattern in ingresso viene fatto un confronto con il contenuto del database di pattern già estratti.

Un esempio tipico di un sistema di estrazione di pattern off-line è EMI [10]. Allo scopo di identificare la 'firma' stilistica di un compositore, si cerca prima all'interno di ciascuna opera del corpus i pattern ricorrenti, successivamente quelli relativi alle diverse opere vengono confrontati tra loro. Solo i pattern che ricorrono in diverse composizioni possono essere considerati 'firme' stilistiche dell'autore. Lo schema generale del processo di estrazione è presentato in fig.11. L'input di un sistema di estrazione è di solito una rappresentazione Midi di un set di pezzi musicali. La selezione dei frammenti consiste nel selezionare parti delle sequenze sorgenti che verranno confrontate tra di loro. Se frammenti confrontati vengono considerati simili, essi formano una classe di equivalenza ad ognuna delle quali corrisponde un pattern candidato. La selezione dei pattern viene fatta in base ad un principio di 'importanza', dato dal numero di frammenti all'interno delle corrispondenti classi di

equivalenza. I pattern così estratti vengono inseriti in un pattern-base corrente.
Nei sistemi di estrazione di pattern online (vedi fig. 12), come nei sistemi interattivi in tempo reale, lo schema è differente. L'input è uno stream di note in formato Midi che viene segmentato in frasi. Nel sistema Cypher [23] ogni frase viene ridescritta come una serie di intervalli di semitoni, successivamente le ultime due frasi segmentate vengono confrontate, e se coincidono per almeno quattro note si trova un pattern candidato. Questo viene poi confrontato con quelli già esistenti e se non c' è coincidenza viene considerato unico ed aggiunto al pattern base, mentre se concide con quelli già selezionati non viene aggiunto al pattern-base, ma viene incrementato un parametro di forza del pattern esistente.

# 7 Tavola degli algoritmi

In questa sezione viene presentata una lista di alcune pubblicazioni in cui algoritmi per lo string matching venivano usati nella musica.

| | **Pitch representation** | **Rhythm representation** | **Other structural factors** | **Pre-segmentation required** | **Type of pattern processing** | **Type of matching** | **String matching algorithm** |
|---|---|---|---|---|---|---|---|
| **Mongeau & Sankoff 1990 [16].** | pitch/ degree difference from tonal centre | durations | weights based on degree of consonance | | comparison | approximate | dynamic programming |
| **Stammen & Pennycook 1993 [28].** | intervals in semitones | duration ratios | no | yes | recognition | approximate | dynamic programming |
| **Smith, McNab & Witten 1998 [27].** | intervals in semitones & contour | durations | no | no | recognition | exact & approximate | dynamic programming |
| **Rowe 1995 [24]; Rowe & Li 1995 [25]** | intervals in semitones | durations | no | yes | recognition & induction | approximate | dynamic programming |
| **Rolland & Ganascia 1996a,b [20]; Rolland 1998 [18].** | absolute pitch & intervals | durations & ratios | contribution weights (e.g. long dur.) | no | induction | approximate | dynamic programming |
| **Bakhmutova, Gusev & Titkova 1997 [2].** | scale-step intervals | | metric position | no | induction | approximate | dynamic programming |
| **Cope 1990 [9].** | intervals in semitones | durations | elimination of very short notes | no | m-length pattern induction | near-exact | brute-force algorithm |
| **McGettrick 1997 [15].** | abs. pitch, intervals in semitones | duration ratios | accented notes | no | recognition | exact | Boyer-Moore algorithm |
| **Coyle & Shmulevich 1998 [12].** | intervals in semitones (+error) | duration ratios (+error) | key-finding algorithm | | comparison | exact (+error absolute and perceptual) | equal length comparison |
| **Hsu, Liu & Chen 1998 [14].** | absolute pitch | durations | elementary chords and meter | no | induction | exact | dynamic programming (only exact) |
| **Hiraga 1997 [13].** | interv: sem, scale-steps, step-leap, contour | durations: exact, log-ratio, shorter-longer-equal | reduction of surface | yes (tentative) | induction | exact (emphasis in immediate repetition) | not described |
| **Cambouropoulos 1998 [3] [4].** | interv: sem, scale-steps, step-leap, contour | durations: exact, ratio, shorter-longer-equal | reduction of surface | no | induction | exact | Chrochemore (1981) [11]. |

La colonna a sinistra indica vari metodi di elaborazione di pattern musicali, mentre la riga in alto indica alcuni aspetti utili di questi metodi.

## Riferimenti bibliografici


[1] Apostolico A. and Galil Z. 1985. "Combinatorial Algorithms on Word". Springer-Verlag, NATO ASI Series.

[2] Bakhmutova I.V., Gusev V.D. and Titkova T.N. 1997. "The Search for Adaptations in Song Melodies". *Computer Music Journal*, 12(1):5867.

[3] Cambouropoulos E. 1998a. "Towards a General Computational Theory of Musical Structure". Ph.D. Thesis, University of Edinburgh.

[4] Cambouropoulos E. 1998b. "Musical Parallelism and Melodic Segmentation". In Proc. of the *XII Colloquio di Informatica Musicale*, Gorizia, Italy.

[5] Cambouropoulos E. 2000. "Extracting 'Significant' Patterns from Musical Strings: Some Interesting Problems". Presented at the *London String Days 2000* Workshop, 3-4 April 2000, King's College London and City University.

[6] Cambouropoulos E. 2000. "From MIDI to Traditional Musical Notation". In Proc. of the "AAAI Workshop on Artificial Intelligence and Music", Austin, Texas.

[7] Cambouropoulos E., Crawford T. and Iliopoulos C.S. 2000. "Pattern Processing in Melodic Sequences: Challenges, Caveats and Prospects". *Computers and the Humanities*, 34:4.

[8] Cambouropoulos E., Crochemore M., Iliopoulos C.S., Mouchard L. and Pinzon Y.J. 1999. "Algorithms for Computing Approximate Repetitions in Musical Sequences". In Proc. of the *AWOCA '99 Workshop (Australasian Workshop on Combinatorial Algorithms)*, Perth.

[9] Cope D. 1990. "PatternMatching as an Engine for the Computer Simulation of Musical Style". In Proc. of the *International Computer Music Conference*, Glasgow.

[10] Cope D. 1991. "Computers and Musical Style". Oxford: Oxford University Press.

[11] Crochemore M. 1981. "An Optimal Algorithm for Computing the Repetitions in a Word". *Information Processing Letters*, 12(5):244250.

[12] Coyle E.J. and Shmulevich 1998. "A System for Machine Recognition of Musical Patterns". In Proc. of *IEEE ICASSP '98*.

[13] Hiraga Y. 1997. "Structural Recognition of Music by Pattern Matching". In Proc. of the *International Computer Music Conference*, Thessaloniki, Greece.

[14] Hsu J.L., Liu C.C. and Chen A.L.P. 1998. "Efficient Repeating Pattern Finding in Music Databases". In Proc. of the *Conference in Information and Knowledge Management (CIKM'98)*, Bethesda, Maryland.

[15] McGettrick P. 1997. "MIDIMatch: Musical Pattern Matching in Real Time". Msc Dissertation, York University, U.K.

[16] Mongeau M. and Sankoff D. 1990. "Comparison of Musical Sequences". *Computer and the Humanities*, 24:161175.

[17] Owens T. 1974 "Charlie Parker-Techniques of improvisation" Ph.D.Thesis, University of California at Los Angeles, Department of Music.

[18] Rolland P.Y. 1998. "FlExPat: A Novel Algorithm for Musical Pattern Discovery". In Proc. of the *XII Colloquium in Musical Informatics*, Gorizia, Italy.

[19] Rolland P.Y. 2001. "Music Information Retrieval: a brief Overview of Current and Forthcoming Research", Proc. of the *Human Supervision and Control in Engineering and Music.*

[20] Rolland P.Y. and Ganascia J.G. 1996. "Automated MotiveOriented Analysis of Musical Corpuses: a Jazz Case Study". In Proc. of the *International Computer Music Conference*, HongKong.

[21] Rolland P.Y. and Ganascia J.G. 1996. "Automated Extraction of Prominent Motives in Jazz Solo Corpuses". In Proc of the *4th International Conference on Music Perception and Cognition (ICMPC'96)*, Montreal.

[22] Rolland P.Y. and Ganascia J.G. 1999. "Musical Pattern Extraction and Similarity Assessment". In *Readings in Music and Artificial Intelligence.* E. Miranda. (ed.). Harwood Academic Publishers.

[23] Rowe R. 1993. "Interactive Music Systems". Cambridge: MIT Press.

[24] Rowe R. 1995. "Artificial Intelligence and Musical Interaction". In Proc. of the *International Congress in Music and AI*, University of Edinburgh, Edinburgh.

[25] Rowe R. and Li T.C. 1995. "Pattern Processing in Music". In Proc of the *Fifth Biennial Symposium for Arts and Technology.* Connecticut College, New London, Connecticut.

[26] Ruwet N. 1972. "Language, musique, poesie". Paris: Editions Le Seuil.

[27] Smith L.A., McNab R.J. and Witten I.H. 1998. "Sequencebased Melodic Comparison: A DynamicProgramming Approach". *Computing in Musicology*, 11:101117.

[28] Stammen D.R. and Pennycook B. 1993. "Realtime Recognition of Melodic Fragments Using the Dynamic Timewarp Algorithm". In Proc of the *International Computer Music Conference (ICMC '93).*

[29] Stech David A. 1981. "A computer assisted approach to micro analysis of melodic lines." *Computer and the humanities* 15, pp.211-221. North Holland Pub.Co.

[30] Widmer G. 2001. "Inductive Learning of General and Robust Local Expression Principles", Proc. *International Computer Music Conference.*