Gabriele Petrillo

Problematiche della composizione musicale elettroacustica

Ciò che segue è stato sviluppato a partire dalla dispensa di composizione del 17/02/2020 del prof. N.Bernardini presso il conservatorio S.Cecilia di Roma che possono essere consultati pressohttps://git.smerm.org/SMERM/TR-2019-2020/src/branch/master/COME-02/A.A.2019-2020/20200217. Questa dispensa tratta delle problematiche principali della composizione elettroacustica e sviluppa un primo approccio a Csound.

Csound

Csound è un *software* per la sintesi digitale diretta del suono realizzato da Barry Vercoe al M.I.T. (*Massachusetts Institute of Tecnology*)

Per ottenere qualsiasi tipo di suono in Csound è necessaria la scrittura di due testi differenti:

- Orchestra
- Partitura

Su questi due testi noi scriveremo rispettivamente le informazioni sugli strumenti di cui abbiamo bisogno e le operazioni che questi devono svolgere, dopo di che il programma creerà un file audio con i suoni che noi abbiamo richiesto.

Orchestra

L'orchestra è composta da due sezioni:

- Header: dove inseriamo le informazioni di base che devono seguire tutti gli strumenti;
- Strumenti: dove costruiamo i nostri strumenti virtuali.

In generale un Header contiene sempre queste informazioni:

- **sr** frequenza di campionamento dei segnali audio (sample rate)
- **ksmps** rapporto tra sr e kr (frequenza di campionamento dei segnali di controllo)
- **nchnls** numero dei canali di uscita

Un esempio di Header del nostro primo programma Csound è:

```
sr = 44100
ksmps = 32
nchnls = 1
0dbfs = 1
```

Dove è stato aggiunta l'opzione 0dbfs = 1 che serve a normalizzare i valori delle ampiezze tra -1 e 1.

Gli strumenti

Gli strumenti sono molto più vari perché dipendono da ciò che vogliamo creare. La prima cosa da scrivere è il numero dello strumento, con l'istruzione *instr* seguita da un numero. L'ultima parola è *endin* con il quale si termina lo strumento. All0'interno dello strumento dobbiamo assegnare alle variabili (che possiamo immaginare come dei cassetti dove vengono riposti tutti i risultati di determinate operazioni) i vari opcode (codici operativi di funzioni di Csound, come ad esempio gli oscillatori o i filtri ecc.) e i vari argomenti che necessita l'opcode utilizzato. Nella nostra orchestra abbiamo utilizzato 4 opcode differenti:

- **soundin** permette di leggere un file audio
- **diskin** come soundin permette di leggere un file audio ma consente di variare la ve-

locità di lettura, la direzione e realizzare dei loop.

- **oscil** è un oscillatore.
- **loscil** legge un file audio importato in una tabella e consente di eseguire le stesse operazioni di un campionatore come legger eil file audio cambiando ampiezza, frequenza e punti di loop.

```
instr 1
a1 soundin "sample_spoon.wav"
out a1
endin

instr 2
a1 diskin "sample_spoon.wav", p4
out a1
endin

instr 3
kfreq oscil 50, 0.8, 2
kfreq = kfreq + 261.6
a1 loscil 1, kfreq, 1, 261.6, 2, 5000, 35000
out a1
endin
```

Bisogna fare una piccola precisazione sul nome delle variabili, infatti con *a* si intendono le variabili audio (quindi ogni variabile che tratterà segnali audio inizierà per a) mentre con *k* tutte le variabili che contengono segnali di controllo (come ad esempio LFO o inviluppi ecc.). Per ultimo, alla fine di tutta la catena avremo una variabile in uscita definita dal comando *out* più il nome della variabile alla fine di ogni strumento.

Parametri degli opcode

- **soundin** nome del file audio.
- **diskin** nome del file audio, velocità di lettura (1=normale, 2=doppia, ecc.)
- **oscil** frequenza, ampiezza, numero della tabella che contiene la forma d'onda.
- **loscil** ampiezza, velocità di lettura (1=normale, 2=doppia, ecc.), numero della tabella che contiene la forma d'onda, frequenza del file audio (261.6 di default), modo del loop (0=nessun loop; 1= loop semplice, 2=loop avanti e indietro), numero del campione da cui far iniziare il loop, numero del campione da cui far terminare il loop.

Spiegazione degli strumenti

lo strumento 1 semplicemente legge il file audio contenuto nella stessa cartella del file Csound.

Il secondo strumento legge lo stesso file audio, ma variando la velocità di lettura del file secondo il parametro p4 specificato nella partitura.

Il terzo strumento è più complicato e contiene un oscillatore di controllo che genera una sinusoide che va a 50 Hz che va da -0.8 a 0.8. La seconda variabile aggiunge un Offset a *kfreq* di 261.2 in quanto dovrà modulare la frequenza in output del file audio e 0.8 sarebbe eccessivamente bassa (calcolando che 261.6 è la frequenza originale). A questo punto l'opcode *loscil* leggerà il file audio contenuto nella tabella modulando la frequenza in uscita. In output avremo quindi *a1*.

La partitura

Anche la partitura è formata da due tipi di istruzioni:

- **funzioni**: servono a creare forme d'onda di cui possiamo scegliere le caratteristiche.
- **note**: sono tutti gli eventi sonori dei vari strumenti e la loro posizione nel tempo.

La lettera iniziale di ogni riga della partitura indica il tipo di istruzione: *f* per le funzioni e *i* per le note.

```
f1 0 0 1 "sample_spoon.wav"000
f204096101

i1 0 0.1
i1 0.15 0.1
i1 0.35 0.1

i2 1 1 1
i2 2 2 0.25

i3 4 8
```

Vediamo come si scrive una funzione direttamente dal nostro esempio. Le funzioni vengono numerate, quindi *f1, f2* ecc. il primo parametro è il momento di creazione della funzione in secondi (0 all'inizio, 3 al terzo secondo ecc.). Poi dobbiamo indicare il numero di punti che costituiscono la nostra tabella (normalmente è 4096, o comunque una potenza di 2). Il terzo parametro è l'indicazione del metodo di generazione (GEN) ovvero il metodo utilizzzto per la generazione delle forme d'onda, nel primo caso è 1 perchè l'opcode *loscil* richiede la GEN01, mentre la seconda funzione è GEN10 (che serve a creare sinusoidi) i numeri successivi dipendo dalla GEN utilizzata, nel caso di GEN10 rappresentano l'ampiezza delle sinusoidi che compongono la forma d'onda, nel nostro coso solo una. Nel caso di GEN01 abbiamo il nome del file, lo skiptime, il formato e il numero di canale che se impostati a 0 indicano lo stesso dell'header.

Per gli strumenti viene identificato il numero dello strumento che deve suonare, e dai vari parametri denominati p1, p2 ecc. che possono essere utilizzati anche per modificare dei parametri negli strumenti. Il primo parametro è sempre il numero dello strumento, il secondo il momento di inizio della nota e il terzo la sua durata. Tutti gli altri parametri si rifanno a valori negli strumenti, come nel caso di *diskin* in cui la frequenza di lettura è cambiata dal parametro p4.

Tag e Flag

Come possiamo vedere nel nostro programma completo all'inizio e alla fine sono inserite delle diciture racchiuse tra < e > che ancora non abbiamo decifrato. Questi sono i tag, ovvero delle parole chiave che contengo delle informazioni, nel nostro caso abbiamo la prima che identifica il nostro programma e si apre e si chiude all'inizio e fine dell'intero codice (<CsoundSynthesizer>). Poi abbiamo le opzioni (<CsOptions>) che identificano il tipo di output che deve avere il nostro programma. Qui possiamo inserire molti FLAG, ovvero dei codici che servono a gestire il tipo di file compilato, in questo caso avremo un file Wave di nome *primo*. L'ultimo tag è la partitura (<CsScore>).

```
<CsoundSynthesizer>
<CsOptions>
-o primo_test.wav W
</CsOptions>
<CsInstruments>
;
; orchestra
;
sr = 44100
ksmps = 32
nchnls = 1
0dbfs = 1

instr 1
a1 soundin "sample_spoon.wav"
out a1
endin

instr 2
a1 diskin "sample_spoon.wav", p4
out a1
endin

instr 3
kfreq oscil 50, 0.8, 2
kfreq = kfreq + 261.6
a1 loscil 1, kfreq, 1, 261.6, 2, 5000, 35000
out a1
endin
</CsInstruments>
<CsScore>
;
; partitura
;
f1 0 0 1 "sample_spoon.wav" 0 0 0
f2 0 4096 10 1

i1 0 0.1
i1 0.15 0.1
i1 0.35 0.1

i2 1 1 1
i2 2 2 0.25

i3 4 8
</CsScore>
</CsoundSynthesizer>
```

BIBLIOGRAFIA

•Bianchini, Riccardo and Cipriani, Alessandro, *Il suono virtuale*, Contemponet 2001