Stria, di John Chowning

Analisi del Processo Compositivo

Padova, 19/12/2002
(revisione del 16/02/2012)

Matteo Meneghini,
Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Padova
e-mail: matteo.meneghini@dei.unipd.it
tel. +39 049 827 7664

Sommario

Queste pagine costituiscono il risultato di un lavoro di **analisi di Stria**, un brano di musica elettronica composto da John Chowning nel 1976, eseguita a partire dalla lettura dei listati dei programmi con cui la composizione è stata generata.

L'ascolto della composizione ed il continuo riferimento alla letteratura sono stati di aiuto, sebbene non siano molte le informazioni specifiche reperibili nei testi.

Il lavoro è diviso in tre sezioni:
- Nella prima sezione sono richiamati alcuni concetti-chiave utili nell'analisi del brano.
- La seconda parte raggruppa un insieme di considerazioni globali sulla composizione, della quale dà una caratterizzazione, esaminando le frequenze e le durate dei suoni generati.
- Nell'ultima sezione sono invece analizzati gli algoritmi che generano i suoni, entrando quindi nel dettaglio del codice.

Gli algoritmi sono scritti in "SAIL", un linguaggio che ha avuto una breve storia negli anni '70 ed adesso non è più utilizzato. Le nozioni usate per l'analisi degli algoritmi sono state reperite in rete, dove, con qualche difficoltà si riescono a trovare alcuni tutorial e guide dell'epoca.

Alcune informazioni sono state ottenute da Chowning stesso, che ha gentilmente chiarito alcuni dubbi sorti durante l'analisi, in relazione ad alcuni parametri usati nella definizione delle caratteristiche dei suoni.

Ulteriori informazioni possono essere reperite nelle pubblicazioni

http://www.dei.unipd.it/~menego/CIM2003_Meneghini.pdf
http://www.mitpressjournals.org/doi/pdf/10.1162/comj.2007.31.3.26

1. Introduzione

Stria fu composta nel 1976 da John Chowning, professore dell'Università di Stanford (USA), pochi anni dopo che questi aveva pensato di applicare alla sintesi del suono la tecnica della modulazione di frequenza, nota all'epoca soprattutto nell'ambito delle telecomunicazioni.
Insieme a "Turenas" (1972) e alla più tarda "Phonee" (1981), quest'opera rappresenta il punto di incontro tra la ricerca di tecniche di sintesi innovative e la grande esperienza in acustica e psicoacustica raggiunta da Chowning.

Il brano, che in tutto dura poco più di 17 minuti, è stato generato interamente a partire da una **serie di algoritmi**, che, ricevendo in ingresso i **parametri** dei suoni e degli eventi sonori da generare, restituivano in uscita una **serie di file**, contenenti le informazioni sui processi così creati.
Uno di loro, in particolare, veniva utilizzato per la generazione di suoni da parte di un ulteriore software.

Prima di analizzare nel dettaglio questi algoritmi, è importante fare qualche breve premessa riguardante alcuni concetti-chiave, indispensabili per la lettura di quest'opera sia dal punto di vista artistico-compositivo, sia dal punto di vista strutturale.
Tra questi è importante ricordare la **sezione aurea** e le sue proprietà, nonché la tecnica di **sintesi per modulazione di frequenza**.

1.1 La sezione aurea

Vista l'origine geometrica di questa grandezza, possiamo descriverla considerando un **segmento**, di lunghezza unitaria l=1.
Cerchiamo quindi la lunghezza x di una sua parte tale che il rapporto tra la lunghezza totale del segmento e x sia uguale al rapporto tra x e la restante parte di segmento, cioè consideriamo l'uguaglianza

$$\frac{1}{x} = \frac{x}{1-x}$$

Risolvendo l'equazione nell'incognita x troviamo che

$$x = \frac{1}{2}\left(-1+\sqrt{5}\right) = 0.618$$

Possiamo quindi allargare questo rapporto alla **proporzione continua**

$\frac{1-x}{x} = \frac{x}{1} = \frac{1}{1/x} = \ldots$ che numericamente porta a $\frac{0.382}{0.618} = \frac{0.618}{1} = \frac{1}{1.618} = \ldots.$

L'importante rapporto ottenuto, la sezione aurea, è considerato fin dall'antichità un canone di perfezione fisica, e è facilmente riconoscibile sia in moltissime opere (es. architettoniche), sia in natura.

In ambito musicale il rapporto aureo rappresenta in buona approssimazione una **sesta minore**, secondo la notazione occidentale. Infatti, un'ottava (intervallo tra due frequenze una doppia dell'altra) può essere divisa in 12 semitoni, disposti secondo una successione geometrica di ragione $\sqrt[12]{2}$; il rapporto tra un semitono e la sua sesta minore (otto semitoni) corrisponde quindi a $\sqrt[12]{2^8} \cong 1.6$.

Un'altra proprietà da mettere in luce riguardo la sezione aurea è legata alla **successione di Fibonacci**: questa successione, a partire dai termini 0,1,2 genera ogni termine come somma dei due precedenti.

In particolare, si può notare che il rapporto tra due termini successivi di questa successione tende velocemente alla sezione aurea: da ciò si ricava facilmente che **le potenze di G=1.618 sono ordinate secondo una successione di Fibonacci,** cioè che vale

$$G^n = G^{n-1} + G^{n-2}$$

Queste proprietà rappresentano parte della chiave di lettura di Stria, è quindi importante ricordarle nell'analisi della composizione.
Una trattazione approfondita di questo argomento esula dagli scopi di questa analisi, ma in letteratura si trovano molti spunti.

1.2 La modulazione di frequenza

La sintesi per modulazione di frequenza è una sintesi **non lineare**: l'applicazione di una trasformazione non lineare al segnale in ingresso, ne arricchisce lo spettro e lo traspone, semplicemente variando alcuni parametri.
Fu proprio **Chowning**, nel 1973, a proporre una trattazione dettagliata di questo tipo di sintesi.
In generale, questa tecnica consiste in una **modulazione di argomento** (cioè una modulazione che varia l'angolo, argomento di una sinusoide), che può essere intesa sia come modulazione di fase, sia come modulazione di frequenza (FM).
L'idea alla base della modulazione di frequenza consiste nel variare la fase (modulazione di fase) o la frequenza istantanea (modulazione di frequenza) di un oscillatore di solito sinusoidale (portante) utilizzando un altro oscillatore sinusoidale (modulante).

Consideriamo intanto di modulare la fase, sommando all'argomento del seno una sinusoide di frequenza f_m e di ampiezza I: il segnale modulato è del tipo

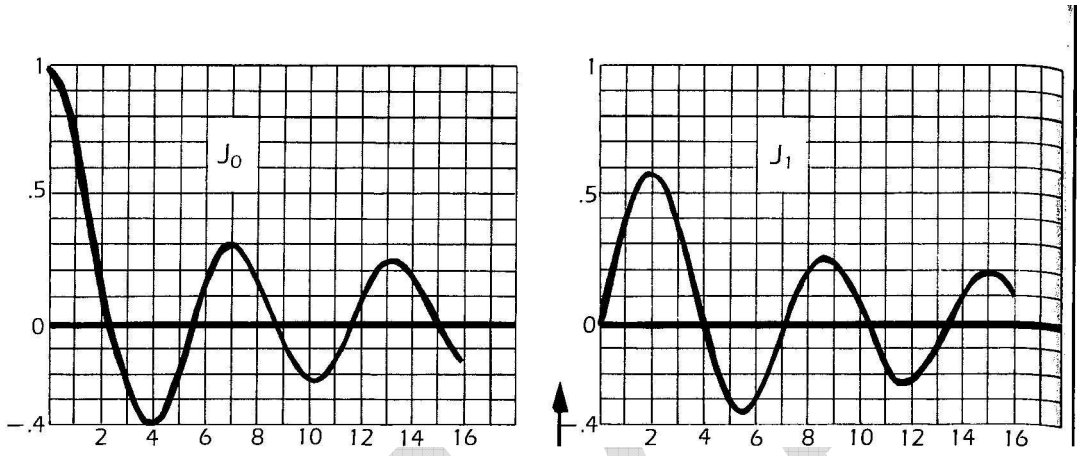
$$s(t) = sin[2\pi f_c t + I sin(2\pi f_m t)]$$

e usando gli sviluppi in serie può essere espresso come

$$s(t) = \sum_{k=-\infty}^{+\infty} J_k(I) sin\,|\,2\pi(f_c + k f_m)t\,|\,.$$

Lo spettro risultante, quindi, presenta una serie di righe alle frequenze $|\,f_c \pm k f_m\,|$, con k=0,1,2,… le cui ampiezze sono definite dai valori delle funzioni J_k di Bessel di ordine k, calcolate nel punto I; in figura sono

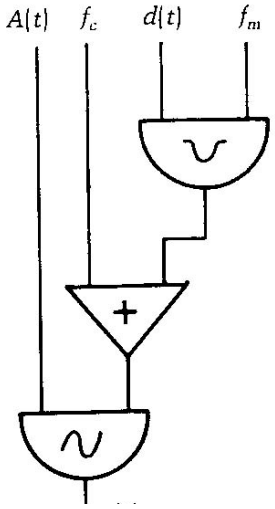
riportate le funzioni di Bessel di ordini 0 e 1. Queste funzioni sono tabulate in molti testi e non si ritiene opportuno approfondire qui l'argomento.
È comunque importante notare come al variare di I, detto indice di modulazione, non varino le righe spettrali, ma solo le loro ampiezze; questo parametro quindi definisce un utile strumento di controllo del timbro dei suoni generati mediante FM.



A parità di modulante, variando la portante si ottiene una traslazione dello spettro intorno alla nuova portante, mentre, variando la modulante a portante fissa, si ha uno spettro sempre centrato intorno a f_c, ma costituito da righe più larghe o più strette, a seconda della variazione di f_m: le frequenze di portante e modulante, quindi, costituiscono il parametro di controllo dei movimenti delle righe spettrali.

La larghezza di banda di uno spettro FM è infinita, perchè le funzioni di Bessel non diventano mai nulle per k>0. Per definire la larghezza di banda dello spettro FM, quindi, si utilizza una definizione basata sull'ampiezza delle componenti, secondo cui si considerano solo le componenti di ampiezza maggiore di un centesimo del segnale non modulato.
Il numero M di bande laterali considerate significative è dato dalla formula $M = I + 2.4I^{0.27}$, o in modo approssimato (ma più rapido) da $M = I + 1$.

Rappresentando lo stesso tipo di modulazione come **una modulazione di frequenza,** invece, si considerano le variazioni della frequenza istantanea intorno a un valore di riferimento, detto portante: la frequenza istantanea, definita come derivata nel tempo dell'argomento (calcolata sul segnale modulato in fase) vale $f_i = f_c + If_m \cos(2\pi f_m t)$, e ciò significa che modulare la fase sommandole una sinusoide di ampiezza I equivale a modulare la frequenza istantanea sommandole una sinusoide di ampiezza $d=If_m$ (a parte il cambio di fase introdotto nella derivazione).
In ogni caso sia la modulazione di fase sia la modulazione di frequenza sono due tipi di modulazione di argomento.



Nella figura a fianco viene rappresentata la modulazione di frequenza come variazione della frequenza istantanea dell'oscillatore portante per sovrapposizione a f_c di una modulante sinusoidale.

Il rapporto tra la portante e la modulante (f_c/f_m) è indice dell'armoniosità del suono: se è un numero semplice (es. 1,2,…) le componenti generate saranno in rapporto armonico tra loro e potranno essere intese come parziali di un suono avente un pitch riconoscibile; se invece il rapporto è più complesso (es. irrazionale) le righe spettrali risultanti non saranno distribuite uniformemente sullo spettro, e si genererà un suono inarmonico.

Un'evoluzione rispetto a questo schema si ottiene sommando all'argomento della sinusoide portante **due sinusoidi modulanti**: si ha così il segnale modulato $s(t) = sin[2\pi f_c t + I_1 sin(2\pi f_1 t) + I_2 sin(2\pi f_2 t)]$.

Seguendo lo stesso ragionamento del caso a modulante semplice, sviluppando prima la serie rispetto a una sinusoide modulante e poi rispetto all'altra si trova che

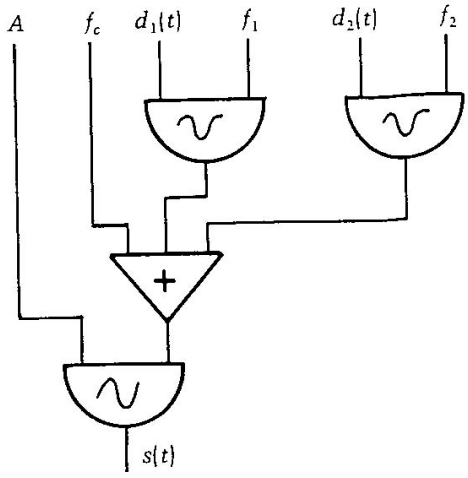
$$s(t) = \sum_k \sum_n J_k(I_1) J_n(I_2) sin[2\pi(f_c + kf_1 + nf_2)t]$$

e quindi lo spettro di s(t) è composto da righe alle frequenze $|f_c \pm kf_1 t \pm nf_2 t|$, ognuna di ampiezza $J_k(I_1)J_n(I_2)$.

Una interpretazione semplice dello spettro così generato si ottiene, nel caso di $f_1 > f_2$, considerando presente solo f_1: in questo modo si genera un segnale con righe spettrali alle frequenze $|f_c \pm kf_1|$.

Se si applica anche la sinusoide a frequenza f_2, queste componenti diventano a loro volta portanti, con bande laterali prodotte da f_2, alle frequenze suddette.

Lo spettro risultante è quindi molto ricco, e al variare degli indici di modulazione acquista diverse caratterizzazioni, tendendo alla sola portante (per indici bassi), e diventando invece quasi rumoroso (per valori elevati degli indici).



Anche in questo caso la doppia modulazione di argomento può essere intesa come la sovrapposizione di due sinusoidi di ampiezze $d_1=f_1I_1$ e $d_2=f_2I_2$ alla frequenza istantanea di un oscillatore sinusoidale, come nello schema in figura.

Anche per i suoni a doppia modulante valgono considerazioni sui rapporti tra le frequenze in gioco ($f_c:f_1:f_2$), analoghe a quelle valide per modulante semplice: a rapporti semplici corrisponderanno quindi spettri simili a quelli trovati usando solo f_1, solo più ricchi, mentre per rapporti complessi si troveranno suoni inarmonici.

Appare evidente il vantaggio della sintesi per modulazione di frequenza: essa è in grado di generare spettri ricchi e complessi, semplicemente combinando un ridotto numero di oscillatori sinusoidali tra loro; inoltre

questi spettri sono controllabili agendo su un numero ridotto di parametri, come gli indici di modulazione e le frequenze in gioco.
In questa trattazione semplificata non ha significato considerare eventuali sfasamenti tra le componenti sinusoidali, in quanto spesso essi sono percettivamente trascurabili.

2. La struttura del brano

Molti aspetti di Stria sono basati sulla sezione aurea (1:1.618…), che, come già ricordato, è stata utilizzata da moltissimi artisti nel corso dei secoli: per esempio, i Greci consideravano questo rapporto simbolo di perfezione e bellezza spaziale e Bela Bartok usò questo rapporto in alcune sue opere come relazione per definire le durate dei movimenti.

Troveremo quindi che, sia i rapporti tra le frequenze e le componenti armoniche, sia i rapporti di durata, sono regolati in Stria dal rapporto di sezione aurea, creando così un'efficace struttura matematica, rigorosamente definita da Chowning.

2.1 Lo spazio delle frequenze

In seguito a una serie di esperimenti sulla sintesi per modulazione FM, Chowning si mise alla ricerca di un rapporto inarmonico con cui ridefinire il concetto di ottava: **serviva un rapporto che generasse componenti spettrali di sintesi (FM) che fossero potenze proprio di quel rapporto**.
Dopo una serie di prove e calcoli eseguiti prima della programmazione (1974, Berlino), Chowning scoprì che la sezione aurea aveva tutte le caratteristiche che egli stava cercando.

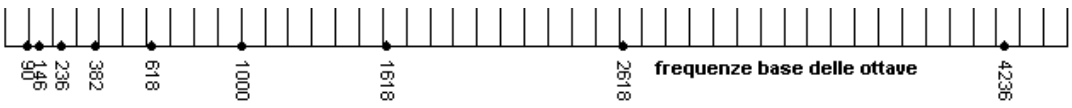
Egli **ridefinì quindi il concetto di ottava**, come distanza tra due frequenze in rapporto 1:1.618 tra loro (invece del tradizionale 1:2 usato nella musica occidentale): ogni ottava così generata fu poi divisa da Chowning in 9 ulteriori frequenze, equispaziate.
Nel seguito, quando si parlerà di ottava, si farà riferimento alla definizione data da Chowning in Stria, e quindi al rapporto 1:1.618.

Detta quindi G=golden mean=1.618 la sezione aurea, considerando come **frequenza centrale dello spazio frequenziale f=1000Hz** (nella musica occidentale la frequenza centrale è f=440Hz, che corrisponde al La sopra il Do centrale del pianoforte), si possono definire i toni base di ogni ottava, alle frequenze

$$G^{-3}f, G^{-2}f, G^{-1}f, f, Gf, G^{2}f \ldots$$

All'interno dell'ottava n-esima, le singole note sono rappresentate dalle frequenze

$$G^{n} f G^{\frac{k}{9}}$$



La figura rappresenta la retta delle frequenze, con le note fondamentali di ogni ottava: ognuna di queste ottave va poi pensata divisa in 9 note secondo il rapporto $G^{\frac{k}{9}}$.

Gli algoritmi prevedono inoltre un'ulteriore divisione dello spazio delle frequenze, e danno la possibilità di generare anche 18 suoni per ottava: il rapporto alla base di questa divisione sarà quindi $G^{\frac{k}{18}}$.

In realtà, per una scelta di dati in input, non si generava mai la possibilità di avere una singola ottava divisa in 18 parti, mentre era frequente il caso in cui lo spazio frequenziale occupato da un intero evento sonoro venisse diviso in 18 note, come sarà chiaro in seguito.

Tra le frequenze fondamentali di ogni ottava c'è un legame dato dalle potenze di G: il vantaggio che G sia proprio la sezione aurea è che questo legame è anche proporzionale, legato cioè a combinazioni lineari a+bG.
Ricordando che la sezione aurea è il limite del rapporto tra due termini consecutivi della successione di **Fibonacci**, sommando due potenze successive di G si ottiene un'ulteriore sua potenza, secondo la tabella:

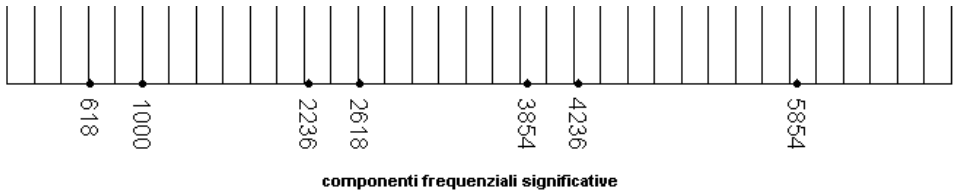
Potenza di G	Combinazione lineare a+bG
$0.056=G^{-6}$	13-8G
$0.090=G^{-5}$	5G-8
$0.146=G^{-4}$	5-3G
$0.236=G^{-3}$	2G-3
$0.382=G^{-2}$	2-G
$0.618=G^{-1}$	G-1
$1=G^{0}$	1
$1.618=G^{1}$	G
$2.618=G^{2}$	1+G
$4.236=G^{3}$	1+2G

Le componenti spettrali che si ottengono sommando tra loro potenze della sezione aurea, sono quindi esprimibili come combinazioni lineari del tipo a+bG.

Chowning decise di sfruttare queste proprietà utilizzando per la sintesi FM un **rapporto tra portante e modulanti basato sulla sezione aurea**.
In questo modo si venivano a generare solo componenti spettrali costituite da somme o differenze di potenze di G, a loro volta in diretta relazione lineare con G.
Veniva così creato un ordine nell'universo frequenziale all'interno del brano, in cui nessuna componente usciva dal rapporto aureo.

A titolo di **esempio** consideriamo il caso di modulazione di frequenza a singola modulante, con portante a 1000Hz e modulante a 1618 Hz, e indice di modulazione relativamente basso, così da poter ritenere significative tre bande laterali per ogni lato.
In questo caso le componenti generate saranno:

|1000-1618|=618	1000	1000+1618=2618
|1000-2*1618|=2236		1000+2*1618=4236
|1000-3*1618|=3854		1000+3*1618=5854



componenti frequenziali significative

Le componenti spettrali generate come combinazioni lineari di potenze di G, a loro volta, possono essere intese come combinazioni a+bG: ciò corrisponde a una costruzione dei suoni robusta dal punto di vista armonico e matematico.

Chowning utilizzò in Stria **otto ottave**, tre delle quali disposte sopra la frequenza di riferimento di **1000Hz**, le restanti disposte al di sotto: le frequenze in gioco (per le fondamentali delle ottave) quindi complessivamente variano tra $G^{-6}*1000=56Hz$ e $G^{3}*1000=4236Hz$.

I dettagli sulle scelte delle frequenze suonate si ritrovano nella parte riguardante la procedura INHARM, più avanti.

2.2 Lo strumento

Utilizzando questa efficace divisione dello spettro in banda audio, Chowning generò tutti i suoni da **un unico strumento**, da lui creato: a partire dai dati di input, gli algoritmi di Stria generavano i trenta parametri necessari alla definizione di ogni strumento.
L'intero brano può essere interpretato come se esso fosse eseguito da un'orchestra di **26 strumenti** di questo tipo, ognuno dei quali emette determinati suoni, uno alla volta, eseguendo parti di partitura: ciò sarà chiarito nella sezione riguardante gli algoritmi.

Lo schema alla base dello strumento è quello del **modulatore FM a doppia modulante**, già richiamato nell'introduzione.
Tutti i modulatori erano di tipo sinusoidale (**funzione seno**), e i due modulatori si sommavano alla **frequenza** della portante.

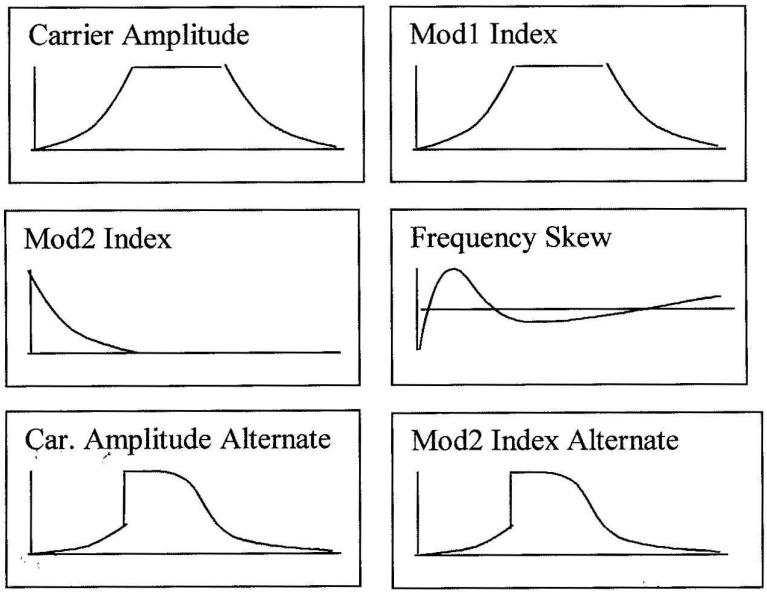
Utilizzando **PD**, è relativamente semplice realizzare uno schema equivalente e rendersi conto della grande variabilità dei suoni ottenibili.

Il modulatore così definito, però, presenta **ampiezza costante del suono e costanza timbrica**. Esso quindi genera suoni monotoni.
L'idea di Chowning fu allora quella di aggiungere ad ogni oscillatore sinusoidale una funzione di **inviluppo non costante**, ottenendo numerosi vantaggi.

Egli associò cioè all'oscillatore principale un inviluppo basato sui parametri di tempo di inizio, durata, ampiezza, durata dell'attacco e durata del decadimento, ottenendo così la definizione di tre fasi: attacco, stato stazionario e decadimento.
Ai due oscillatori che generavano le modulanti, Chowning associò due funzioni di inviluppo definite da due parametri, usati negli algoritmi, che regolavano gli indici di modulazione: **in questo modo si poteva**

controllare la variazione dello spettro del suono, e quindi il suo timbro.

Gli inviluppi associati ai vari oscillatori e generatori di funzione sono rappresentati in figura (gentilmente fornita da Chowning):



Nel caso di portante e modulante c'era anche la possibilità di scelta tra due diversi tipo di inviluppo: quelli alternativi furono usati nel **climax** (punto che vedremo si colloca poco dopo la metà del brano) per creare un effetto quasi rumoroso (che Chowning stesso definisce di tipo "**ssshBoom**").

Nel caso generale l'indice I_1 ha un inviluppo quasi trapezoidale e l'indice I_2 presenta inviluppo esponenziale decrescente, quindi il suono comincia da una modulazione FM praticamente a modulante singola f_2, per evolvere verso un suono a doppia modulante, finire trasformandosi in un suono a modulante singola f_1 e convergere verso la sola portante.

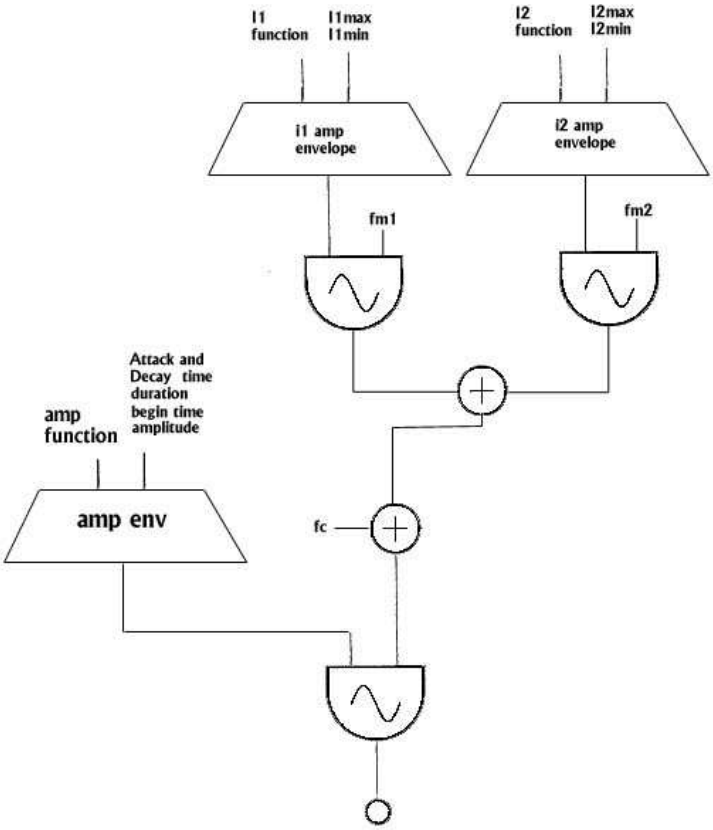
Nel caso alternativo si nota innanzitutto che il suono ha una attacco brusco, che segue un periodo a intensità ridotta.
Inoltre nel momento in cui l'indice di modulazione di f_2 cresce bruscamente si genera uno spettro molto ricco, che genera un'esplosione sonora, il "boom" in questione.
Poi il suono converge verso la sola portante e decade.

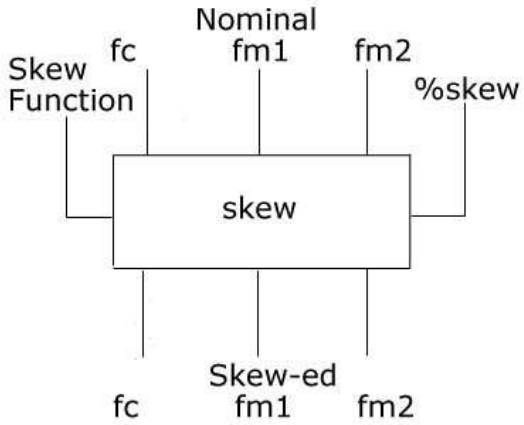
I due oscillatori modulanti permettevano a Chowning di accrescere la densità dello spettro senza tuttavia usare valori elevati per gli indici.
Indici elevati avrebbero ridotto eccessivamente il contributo $J_0(I)$ della portante, cosa che Chowning voleva evitare (per sua ammissione).

Per rompere ulteriormente la regolarità del suono generato da questo strumento, Chowning aggiunse un termine **di leggera deviazione**

frequenziale (skew), che applicava una modifica sia alla frequenza portante, sia alle modulanti, in modo proporzionale.
Si possono quindi considerare le frequenze di portante e modulanti come le uscite di un generatore di funzione, che a partire dai valori nominali dà in uscita le grandezze comprensive di deviazione.
La funzione di skew, nel tempo, ha l'andamento rappresentato in figura, con una forte deviazione all'attacco, che poi velocemente si riduce nel periodo stazionario del suono.
L'ampiezza percentuale dello skew è scelta in base alla frequenza del suono generato, come sarà spiegato meglio nella sezione sugli algoritmi.

Lo strumento completo così realizzato è rappresentato in figura





Nella parte inferiore della figura si nota il generatore di skew, che a partire dai valori nominali di portante e modulanti, aggiunge una deviazione proporzionale a queste grandezze in funzione dei 2 parametri (% skew e funzione di skew) in ingresso, restituendo le grandezze con sovrapposto lo skew: sono queste le frequenze in ingresso agli oscillatori sinusoidali in figura.
Questo è solo uno **schema equivalente**, per intuire il funzionamento dello strumento.

Vale la pena di prendere in esame l'intero **set di parametri** associati a ogni strumento, anche se la comprensione del metodo con cui vengono generati è lasciata a una sezione successiva.

Un primo insieme di parametri definisce **durata, forma e ampiezza del segnale** generato: tra questi troviamo il tempo di inizio del suono, la sua durata, la percentuale legata all'attacco e quella legata al decadimento, il tipo di funzione di ampiezza usata e il valore dell'ampiezza del suono generato; questi sono tutti parametri che costituiscono l'ingresso del generatore di inviluppo dell'oscillatore principale.

Un altro insieme molto consistente di parametri è quello che definisce le **caratteristiche frequenziali** dei suoni.
Tra questi citiamo la frequenza della portante f_c, la frequenza della prima modulante f_{m1} e quella della seconda modulante f_{m2}.
Per ciascuna modulante viene definito un inviluppo dell'indice di modulazione, legato all'andamento di una funzione definita da input (FI_1 e FI_2), che varia tra i valori minimo e massimo definiti per i due indici separatamente attraverso i parametri I_{1max},I_{1min},I_{2max} e I_{2min}.
Viene poi definita la deviazione frequenziale (skew) da applicare alle frequenze di portante e modulanti, attraverso l'impostazione della percentuale di deviazione (% skew) e della funzione che esprime l'evoluzione temporale di questa deviazione (F_{skew}).

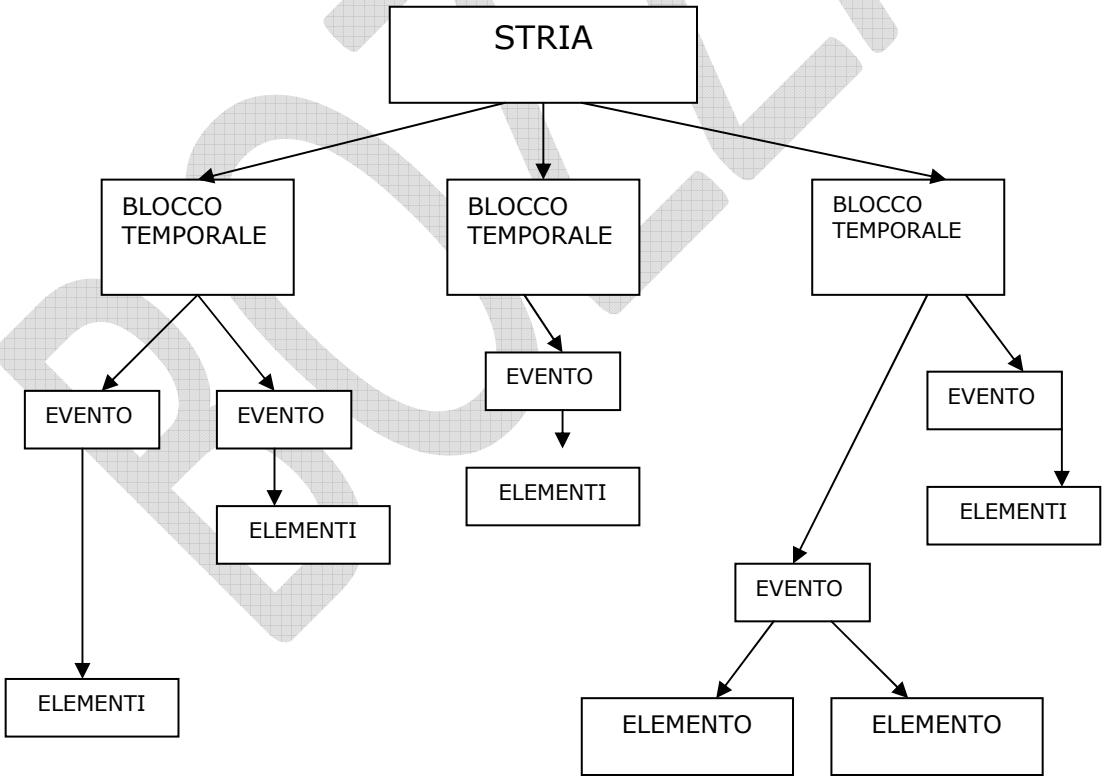
Un ulteriore set di parametri è usato per la **spazializzazione dei suoni**: dato che Stria è stata composta per diffusione quadrifonica, a ogni strumento sono associati i quattro coefficienti moltiplicativi delle ampiezze sui quattro canali, per associare a ogni suono un angolo di provenienza.
È definita anche la percentuale di riverbero che deve essere inserita dopo la generazione di ogni strumento, per rendere le caratteristiche spaziali desiderate.
Un ulteriore termine di spazializzazione è costituito dal parametro DIS_SCALE (scala delle distanze), usato per proiettare il suono nello spazio, impostando il rapporto tra suoni diretti e riverberati, e rendendo quindi l'idea della distanza della sorgente.
Chowning teorizzò i meccanismi psicoacustici attraverso cui avviene la percezione delle distanze nel 1971 nel suo articolo "The Simulation of Moving Sound Sources" e successivamente (anni '90) in "Perceptual fusion and auditory perspective".
In entrambi questi testi si trova teorizzato che la percezione della distanza avviene mettendo in relazione l'intensità del suono diretto (variabile con la distanza dalla sorgente) con l'intensità del suono riverberato (fissa e dovuta all'ambiente).

Questa teoria è applicata in Stria per definire i movimenti apparenti delle sorgenti sonore.

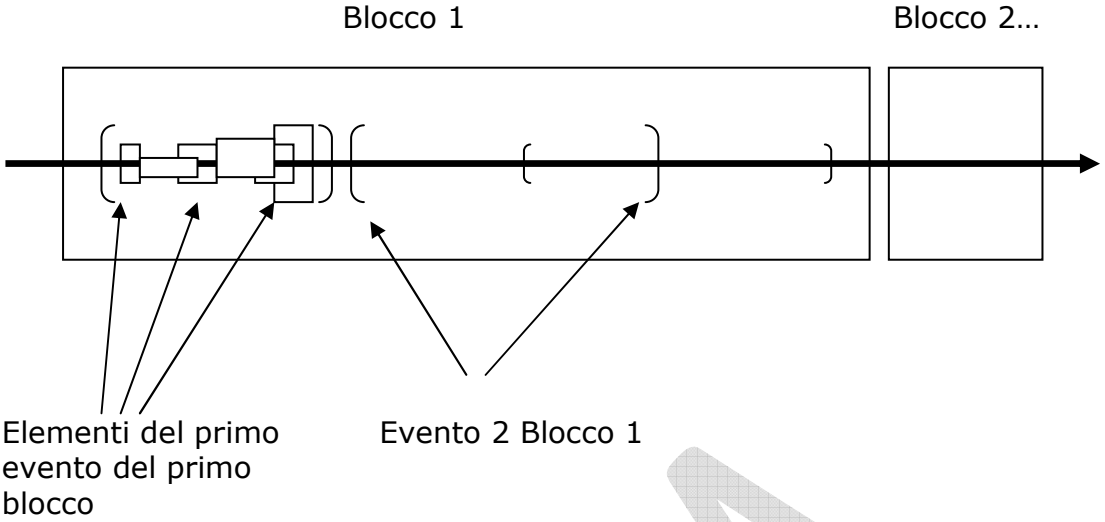
Chowning afferma che in Stria il **controllo spaziale**, basato esclusivamente su angolo e distanza della sorgente (senza effetto Doppler), **non intende essere preciso.**
L'obiettivo che si vuole raggiungere è un **suono** più **amorfo, esteso e indefinito,** come quello che ci può essere in una **cattedrale**: ciò è facilmente percepibile anche all'ascolto.

2.3 La struttura temporale

Come le frequenze, anche le durate dei suoni generati in Stria sono governate dal rapporto aureo.
Per quanto riguarda i tempi, il brano si può considerare composto da una macrostruttura e da una microstruttura.
Un diagramma a blocchi può aiutare la comprensione di questo concetto



Questo schema, nell'evoluzione lineare, equivale al seguente, nel quale i rettangoli più esterni rappresentano i blocchi, che contengono gli eventi (tra parentesi), nei quali si susseguono e si sovrappongono gli elementi (rettangoli più piccoli):



La struttura di Stria può essere interamente divisa in **blocchi**, definiti da input come una sorta di sessioni sonore, ognuna di durata consistente (es. il primo blocco dura 166 secondi). A ciascuno di questi blocchi corrisponde un diverso avvio dell'algoritmo che genera i suoni, e quindi ognuno di essi è salvato in un file di partitura diverso (denominato da Chowning secondo il suo tempo di inizio). Nei listati disponibili (seconda parte, output) le definizioni dei blocchi sono riconoscibili nei file T0.MEM, T466.MEM,…

Ogni blocco così definito può essere costituito da uno o più **eventi** sonori, anche sovrapposti: da input si generano singolarmente le caratteristiche di ogni evento, che riguardano la composizione frequenziale e la distribuzione temporale di tutti i suoni nell'evento.
Sempre nei listati di output, all'interno di ogni blocco si possono chiaramente riconoscere le definizioni dei vari eventi, separate dalla riga di caratteri "**********".

A ogni evento corrisponde una diversa chiamata della procedura di generazione dei suoni (EVENT2).
Uno dei parametri passati a questa procedura è il numero di **elementi** che devono essere generati (variabile elements) all'interno dell'evento.
Ogni elemento può essere considerato un atomo di cui è composta Stria, e corrisponde a un singolo strumento elementare, del tipo descritto nella precedente sessione: nel seguito, visto che Chowning ha denominato ogni strumento generato come elemento, spesso **i due termini saranno usati con lo stesso significato.**

Ogni evento risulterà quindi composto di un certo numero di strumenti, ognuno con la propria partitura, che suoneranno in momenti diversi, e ovviamente in polifonia.
Il massimo numero di strumenti che possono suonare contemporaneamente è definito dalla variabile INSNUM all'interno degli algoritmi, e vale 26.

In Stria Chowning ha utilizzato anche la **ricorsione** per la generazione degli elementi (degli strumenti): gli algoritmi potevano, se adeguatamente impostati da input, generare dei nuovi elementi, oltre a quelli richiesti dal parametro elements in ingresso.

La procedura EVENT2 infatti se erano verificate determinate condizioni sul numero di nest ricorsivi possibili e sulla durata dell'elemento corrente, generava **in sovrapposizione** alla successione dei suoni originali, una serie di nuovi elementi, le cui durate e frequenze variavano rispetto a quelle dei suoni originali, in modo opportuno.
Finita la generazione dei suoni "figli", la procedura portava a termine la generazione dei suoni "padri", da dove era stata interrotta.

Sia nella macrostruttura, sia nella microstruttura, si nota che tutti i legami temporali sono basati sulla sezione aurea.

Considerando l'intero brano, si nota innanzitutto che il **climax**, cioè il picco di intensità acustica, viene raggiunto dopo circa dieci minuti dall'inizio, momento che divide il brano in due parti secondo il rapporto aureo.
Subito dopo il climax l'intensità cala notevolmente, per poi ricrescere verso la fine del brano.

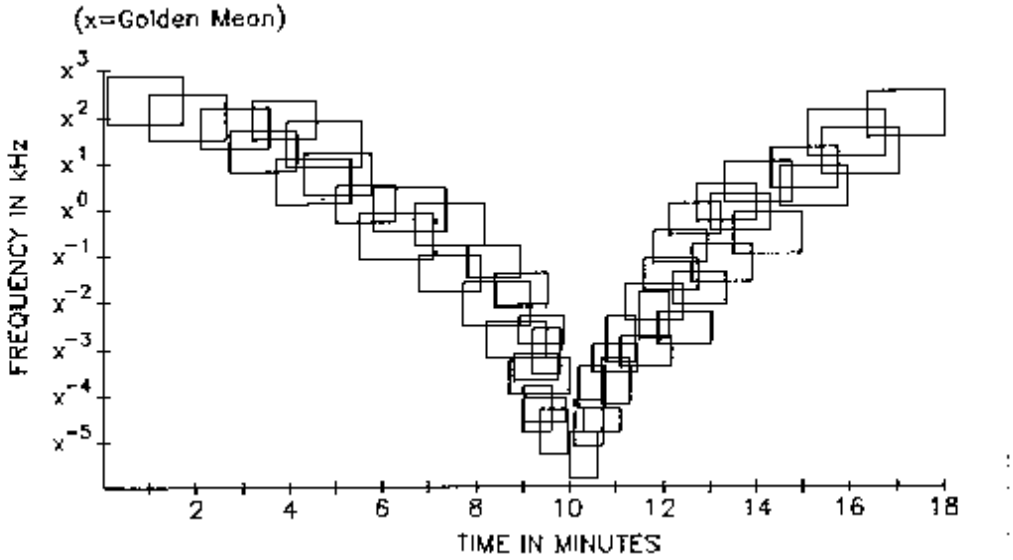
A livello atomico, invece, gli elementi all'interno dei vari eventi sono organizzati secondo una distribuzione temporale dipendente dai parametri globali definiti per l'evento corrente.
A ogni elemento (strumento) sono poi associati i tempi di inzio, attacco, decadimento e fine in base a una successione di Fibonacci basata sul rapporto aureo.

Il criterio secondo cui sono generate le divisioni temporali sarà approfondito nella sezione riguardante l'analisi degli algoritmi.

2.4 Caratteristiche globali

È opportuno mettere in luce alcune caratteristiche globali del brano: innanzitutto come si è già notato, il brano presenta un'intensità crescente dall'inizio fino al punto di climax, intorno ai 10 minuti dall'inizio.
La struttura degli eventi sonori è opposta rispetto a quella usata tradizionalmente a livello compositivo: tradizionalmente, infatti, gli eventi sonori di maggior durata sono quelli legati alle frequenze basse, mentre più brevi sono tenuti i suoni acuti.
In Stria, invece, gli eventi più lunghi sono quelli a pitch maggiore, mentre gli eventi a frequenze basse sono molto più corti.

Il grafico seguente (da "Computer Music", Dodge e Jerse, 1985) rappresenta gli **eventi** di cui è composto il brano, mettendo in evidenza la parte di spettro interessata dall'evento e la sua durata attraverso rettangoli sul piano tempo-frequenza: la frequenza di riferimento di ogni evento è definita attraverso un parametro chiamato da Chowning frequenza base, che è il coefficiente moltiplicativo di portante e modulanti, e rappresenta l'ottava su cui si costruisce l'evento.



Si nota che, all'inizio della composizione, sono presenti suoni alti, che costituiscono eventi di durata considerevole, rappresentati da rettangoli di base maggiore rispetto all'altezza.

Con il passare del tempo, i suoni nella composizione si addensano; nel frattempo cala il pitch degli eventi generati, a cui, nell'impostazione da input dei parametri sonori, Chowning attribuisce durata sempre minore.

I rettangoli rappresentativi degli eventi generati diventano così sempre più stretti ed alti, finchè l'altezza supera la base.

Dopo il climax si ha un nuovo innalzamento del pitch e una nuova rarefazione degli eventi, verso suoni più lunghi e alti.

Anche i **tempi di attacco e di decadimento** dei singoli elementi costituenti gli eventi variano con il pitch dell'evento generato, seguendo le stesse relazioni valide per le durate: per esempio a pitch alto corrispondono attacchi lenti.

Dal grafico si nota anche come nel corso di Stria le frequenze coprano otto ottave, e come vengano tutte spaziate.

3. Gli algoritmi

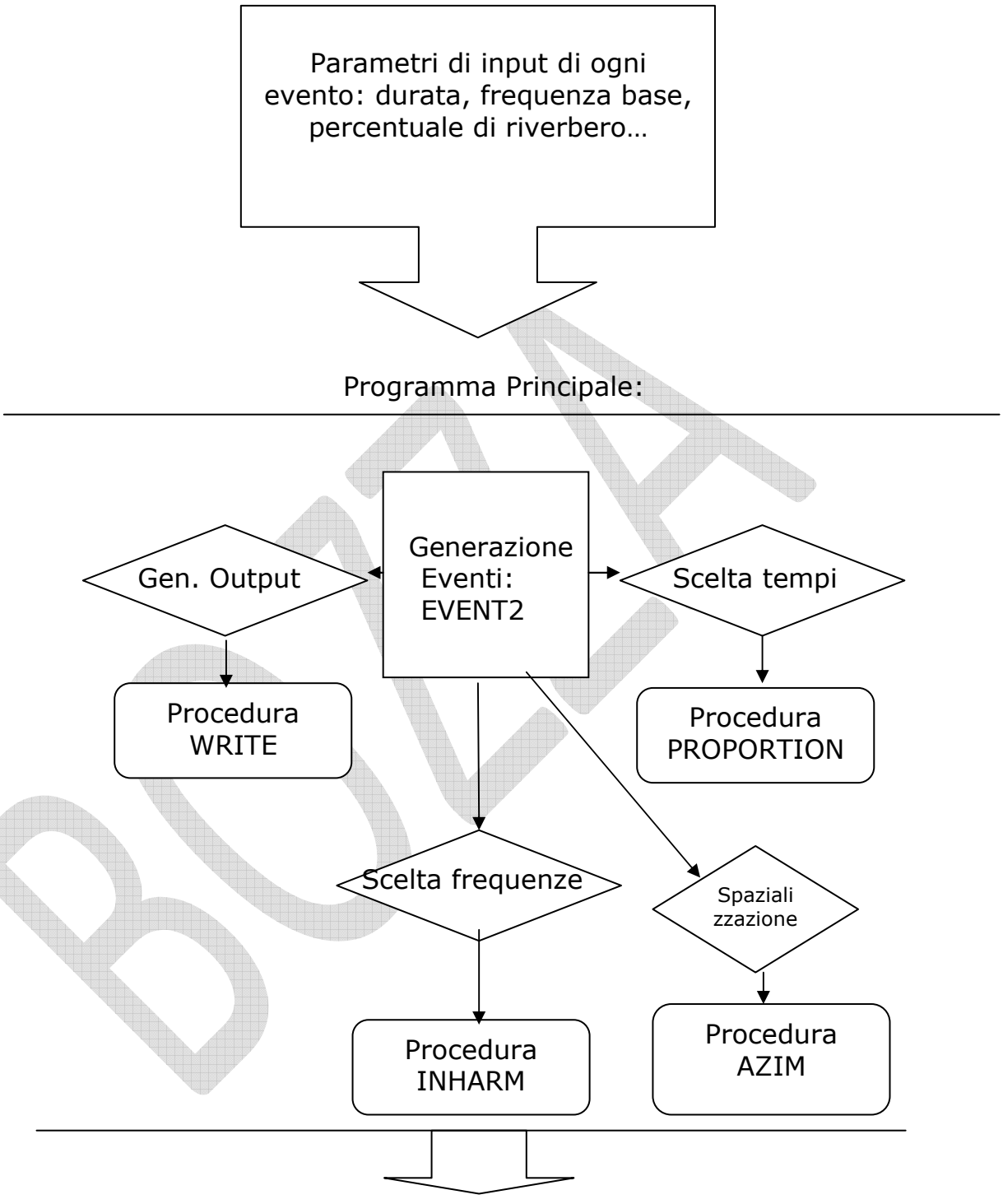
Dopo aver richiamato i concetti della sezione aurea e della modulazione di frequenza, e dopo aver analizzato la struttura del brano dal punto di vista compositivo, è giunto il momento di cominciare l'analisi degli algoritmi, parte essenziale per la comprensione del processo di generazione dei suoni attuato da Stria.

Gli algoritmi sono stati realizzati in **SAIL**, sigla che indica **lo Stanford Artificial Intelligence Language**, un linguaggio di programmazione creato negli anni settanta all'università di Stanford (USA), dove Chowning lavorava.
Questo linguaggio, che non ha avuto molta fortuna, rappresentava la congiunzione tra ALGOL ed il successivo PASCAL, e quindi ha caratteristiche simili a questi due linguaggi; conoscendo PASCAL è relativamente semplice interpretare il codice scritto in SAIL.
Si trovano comunque anche dei tutorial dell'epoca, soprattutto in rete.

Nella pagina seguente è rappresentato uno **schema a blocchi del programma**: il programma principale, chiamato una volta per ogni blocco da generare, riceve in input i parametri di ogni evento del blocco.

La generazione degli eventi avviene attraverso l'utilizzo della procedura EVENT2, che fa riferimento alle altre procedure per la generazione di tempi, frequenze, parametri di spazializzazione ed output.

In uscita al programma rimangono i file di partitura, che verranno processati (suonati) dal software per la generazione dei suoni.



In uscita ci sono i file di partitura (del blocco) nei tre formati definiti, che andranno letti dal software che genera i suoni. Questi file per ogni suono conterranno i parametri generati dalle quattro procedure.

3.1 Struttura dei listati

I listati di cui siamo in possesso sono divisi in **due grandi sezioni**: nella prima troviamo il **codice** che definisce gli algoritmi su cui è basato Stria, mentre nella seconda c'è parte degli **output** generati dai suddetti programmi.
La parte chiaramente più significativa è la prima, dalla quale si possono comprendere i processi matematici che generano le caratteristiche dei suoni, mentre la seconda parte rende conto del tipo di suoni generati, e pone conferma dei risultati trovati.
L'analisi sarà abbastanza dettagliata, e potrà essere di aiuto alla comprensione degli algoritmi; contemporaneamente potrà servire come chiave di lettura del brano e delle intenzioni di Chowning.
Si analizzeranno in successione le procedure chiamate, e infine il programma principale, che consiste praticamente solo nell'accettazione in input delle variabili e nella chiamata alla generazione degli eventi.

Il codice è organizzato in modo di accettare in input i parametri di ogni blocco, diviso in eventi: saranno proprio questi parametri che in ogni evento genereranno i vari elementi sonori (i vari strumenti), sotto forma di file di output; si genererà cioè una partitura (file .SCR), che verrà poi letta da una altro software, funzionale alla generazione dei suoni.

La parte di codice comincia con la definizione delle variabili globali di programma: particolare interesse va alle variabili nella tabella seguente

Nome Variabile o Array	Significato
INSNUM=26	Massimo numero di strumenti che possono suonare contemporaneamente: gli strumenti generati saranno numerati con numeri da 1 a 26. Quando uno strumento avrà finito di suonare, un altro potrà avere lo stesso numero di riferimento, perché nella partitura generata sarà chiaro che si tratta di strumenti distinti.
Vettore PAR[1:30]	È il vettore che verrà usato per definire i parametri dei suoni generati: sarà proprio questo vettore che, stampato su un file di output (.SCR), verrà letto dal software che genererà i suoni. Quasi ognuno degli elementi di questo vettore ha un significato nella definizione di uno dei parametri già citati nella sezione sullo strumento di Chowning.
Vettore ENDTIME[1:INSNUM]	È il vettore che contiene i tempi di fine dei vari strumenti. Grazie a questo vettore si potrà decidere il numero dello strumento da suonare.
DIS_SCALE	Parametro utilizzato per la prioezione spaziale dei suoni e per gestire il rapporto tra suoni diretti e riverberati.

REF_DEG[0:6]	Angolo di riferimento per la generazione del singolo elemento (strumento)
IATT[1:2,1:2], IDCY[1:2:,1:2]	Matrice dei tempi di attacco e di decadimento. Sarà chiaro successivamente perché si usa una matrice.
REV[0:6]	Percentuale di riverbero nell'evento corrente
FREQQ[0:6]	Frequenza base dell'evento corrente
FUNAMP[0:6], FUNI1[0:6], FUNI2[0:6]	Funzioni di inviluppo dei tre modulatori
SKEW[0:6]	Percentuale di skew nell'evento corrente
Freq_space[0:6]	Spaziatura in frequenza: sarà usata per definire la distanza tra le note.
FR[0:2,1:3]	Coefficienti usati per calcolare le frequenze della portante e delle due modulanti
Nest	Valore intero che limita il numero di chiamate ricorsive all'interno di ogni evento
Switch	Intero usato nella definizione della prima modulante
Atswitch	Intero usato per definire una dipendenza dei tempi di attacco e di decadimento dalla frequenza
STRIA	Variabile intera a cui verrà assegnato il valore costante 9, numero chiave nella composizione.
Array di stringhe INSNAME[1:INSNUM], FUN[1:10]	Usati per l'invio di stringhe in output
Ratio	Rapporto aureo, verrà posto uguale a 1.618

Non si sono volute elencare tutte le variabili in gioco, ma solo le più significative, in modo che questo elenco possa essere un utile riferimento nella lettura degli algoritmi.

La porzione di codice immediatamente successiva contiene due procedure, GETOUT e GETOUT2, che chiudono il file di output e lanciano un messaggio di errore nei casi rispettivamente in cui venga superato il numero massimo di strumenti disponibili, e che si verifichi una sovrapposizione non desiderata tra strumenti.

3.2 La procedura WRITE

La procedura **WRITE**, che riceve in ingresso un parametro di tipo intero, di nome INS_CNT, **serve per la gestione dei file di output e la creazione delle partiture.** Anche se non ha un ruolo diretto nella generazione degli eventi sonori, è importante perché costituisce la parte di codice che si rapporta con il mondo e il software esterni al programma.

Prima di scendere nel dettaglio sul codice conviene fare una considerazione: **le partiture generate, a loro modo, sono simili a quelle usate per dirigere un'orchestra**.

Supponiamo infatti di avere un'orchestra composta da INSNUM=26 strumenti uguali, che tutti gli strumentisti siano numerati secondo un indice, e di voler far suonare in ogni momento quelli di indice più basso, a parità di grado di polifonia, cioè di numero di voci strumentali che devono suonare insieme.

Supponiamo inoltre di avere una partitura divisa a pezzi, in cui ogni suono è descritto separatamente: uno strumentista allora potrà suonare le partiture relative a suoni con estensioni temporali distinte, mentre non potrà suonare due suoni che si sovrappongano nel tempo.

Possiamo quindi definire una variabile che indica il numero massimo di strumentisti che hanno suonato contemporaneamente fino a un determinato istante temporale, che indica il massimo grado di polifonia raggiunto fino a quel punto nel brano, e la chiameremo maxnum; un'altra variabile utile è quella che tiene conto di quanti sono i suoni singoli generati fino a un certo istante temporale, e la chiameremo count.

Questi sono ovviamente gli stessi nomi usati in Stria da Chowning per definire le variabili, mentre l'orchestra è l'insieme dei 26 strumenti che possono suonare insieme.

A seconda dei valori di questi parametro si possono verificare tre situazioni:

Se INS_CNT=0	Questo è il caso di inizializzazione in cui infatti la procedura inizializza gli array di stringhe INSNAME e FUN, che serviranno per la generazione delle partiture di output. Poi inizializza i canali per le comunicazioni dei dati, e in particolare rende possibile l'accesso al disco per la creazione, la modifica e il salvataggio di file. A questo punto chiede all'utente il nome dei file su cui andranno salvate le partiture generate nel corso del programma principale, e che verranno lette successivamente da un altro software; ricevuto il nome del file crea: ▪ Un file .SCR (sul canale 1), che rappresenta una partitura a basso livello, costituita da un elenco di parametri numerici e stringhe, che contiene tutte le informazioni per far suonare ogni singolo strumento, che verrà letta dal software per la generazione dei suoni ▪ Un file .MEM (sul canale 2), che contiene un elenco dei parametri a alto livello immessi da input in corrispondenza a ogni blocco, con la loro descrizione; in questi file è riconoscibile l'interazione dell'utente, e la creazione dei parametri globali di blocco e di evento; sono questi i file di output di cui

	siamo in possesso ▪ Un file . REP (sul canale 3), contenente l'elenco di tutti i parametri introdotti da input, senza descrizione Sono tenute in considerazione le varie situazioni di errore, e vengono impostate a 0 le variabili current_time e count (che rappresenta il numero di voci strumentali usate nel blocco), mentre viene attribuito il valore 1 a maxnum, variabile che indica l'indice massimo dello strumento utilizzato (supponendo di numerare gli strumenti da 1 a INSNUM=26), e da un'idea del grado di polifonia nel singolo blocco. Infatti, tenendo conto che quando lo strumento di indice i ha finito di suonare può essere nuovamente utilizzato, nella successiva generazione di una voce si userà nuovamente l'indice i, e non crescerà maxnum. Per elevati gradi di polifonia, tuttavia, il numero maxnum sarà costretto a crescere, visto che non potendo riciclare strumenti già usati se ne dovranno usare di nuovi. WRITE viene chiamata con parametro nullo una volta per ogni blocco di eventi, quindi genera tre file (.SCR, .MEM e .REP) per ogni blocco; ognuno di questi file conterrà anche le informazioni sui singoli eventi. Solo il file .SCR, che è la partitura a basso livello di ogni strumento, definirà accuratamente ogni singolo elemento all'interno di ogni evento. Anche nelle partiture generate da WRITE, quindi, si possono riconoscere la microstruttura (file .SCR) e la macrostruttura (gli altri due file).
Se INS_CNT>0	In questo caso WRITE stampa nel file .SCR di output tutti i parametri riguardanti la voce strumentale corrente: ogni strumento sarà infatti generato dalla procedura EVENT2, che genererà i trenta parametri necessari, e stabilirà a quale strumento associarli; fatto ciò chiamerà WRITE passandole come parametro il numero di strumento scelto. WRITE controlla subito se la variabile PAR[1], parametro che vedremo che indica il tempo di inizio del suono, è minore di CURRENT_TIME, variabile utilizzata per tenere conto dell'evoluzione temporale della partitura, e conoscere sempre l'istante attuale nella generazione dei suoni. Se si verifica questa situazione significa che il suono comincia prima dell'istante attuale di scrittura sulla partitura, e quindi non può essere inserito in partitura: viene allora

	chiamata la procedura GETOUT, che genera un messaggio di errore di miscelazione temporale delle voci strumentali, e chiude il file .SCR, bloccando la partitura al punto di errore. Se invece il tempo di inizio del suono di cui si deve scrivere la partitura è successivo a current_time, la procedura pone current_time uguale al tempo di inizio del nuovo strumento, e comincia le operazioni di scrittura delle partiture sul file .SCR. Se INS_CNT è maggiore del massimo numero di strumenti usati maxnum, si pone maxnum=INS_CNT, per tenere conto del grado di polifonia del blocco. Vengono mandati in output: ▪ Una stringa identificativa del nome dello strumento di indice INS_CNT, tratta dall'array INSNAME[]: le stringhe di questo array, inizializzate nel caso di INS_CNT=0 sono del tipo "INSA" per indice unitario, "INSB" per indice uguale a due e così via fino a "INSZ" per indice uguale a INSNUM=26. Questa stringa serve per comprendere dalla partitura che strumento si sta suonando ▪ Un elenco dei trenta parametri che definiscono le caratteristiche del suono corrente, generato nell'istante current_time dallo strumento INS_CNT: i parametri sono scritti in modo ordinato, dieci per riga, e per i parametri di indice compreso tra 27 e 30, che vedremo che rappresentano i tipi di funzione di inviluppo dei vari modulatori e del generatore di skew, si stampa una stringa del tipo "F1", "F2",… in funzione del valore del parametro. Questa stringa indica quale degli inviluppi è associato a ogni generatore di funzione. Le stringe "FI" sono contenute nell'array FUN[], inizializzato per INS_CNT=0. ▪ Una stringa che indica la fine della partitura relativa allo strumento INS_CNT A questo punto viene aumentato count di una unità: si ricorda che questa variabile indica quanti suoni sono stati generati fino all'istante corrente all'interno del blocco.
Se INS_CNT<0	Questo caso si verifica alla fine della definizione di ogni blocco, da input. Nel file di partitura .SCR viene inserita la stringa "FINISH", che indica la fine del blocco, effettivamente suonabile, mentre i output verso lo schermo e verso il file .MEM vengono mandati i valori di maxnum e di count, che rappresentano il grado di polifonia massimo e il numero di suoni

	generati durante il blocco. Poi viene chiuso il file .SCR. Il blocco così è finito.

3.3 La procedura INHARM: la generazione delle frequenze

La procedura successiva si chiama INHARM, e riceve in ingresso tre variabili, per restituire in uscita un valore reale.
Il funzionamento di questa procedura è strettamente legato alla generazione delle frequenze dei suoni in Stria.
Prima di scendere nel dettaglio è opportuno capire come sono generati questi parametri: innanzitutto, come risulterà evidente in seguito, gli eventi sonori, e quindi i singoli elementi (strumenti) sono generati nella procedura ricorsiva EVENT2, che farà riferimento a INHARM per il calcolo delle frequenze di ogni elemento.

Per la definizione delle frequenze di portanti e modulanti, nel caso più generale EVENT2 fa riferimento a un'espressione che nel caso della portante è del tipo

$$c = fff \cdot freq \cdot f_c$$

dove c rappresenta appunto la frequenza della portante, mentre le altre variabili hanno diversi significati:

- **freq** rappresenta la frequenza base dell'evento, e viene immessa da input. Costituisce in pratica la **scelta dell'ottava di riferimento**, che definisce le altezze dei suoni e le distanze tra le componenti spettrali. Intorno a freq si costruirà l'intero evento.
- **f_c rappresenta il fattore moltiplicativo** che, a partire dal prodotto fff*freq, permette di trovare la frequenza della portante: allo stesso modo ci saranno anche i coefficienti f_{m1} e f_{m2}. Tutti questi coefficienti sono immessi da input secondo **potenze di 1.618**.
- **fff** è un parametro, diverso per ogni elemento (cambia a ogni chiamata di INHARM), calcolato da EVENT2 attraverso la procedura INHARM, che può essere inteso come **frequenza di scala**. Moltiplicato per il valore freq (nota fondamentale dell'ottava di riferimento) definisce a quale nota si sta facendo riferimento, considerando la divisione dello spazio delle frequenze realizzato da Chowning, che come già detto ha diviso lo spazio delle frequenze in ottave di 9 o 18 note.

A partire dal prodotto fff*freq utilizzato come massimo comun divisore, secondo la spaziatura basata sul rapporto 1:1.618, si generano quindi portante e modulanti.

Valori tipici per queste variabili sono dell'ordine di potenze di 1.618 per fff e f_c, mentre freq è del tipo $1000*1.618^k$.

INHARM riceve in ingresso i parametri:

space	Viene definito direttamente da input, e nella chiamata da programma principale il riferimento è $FREQ\_SPACE = ratio^{power}$ dove ratio=1.618 e power è un valore intero, che rappresenta una sorta di fattore di pesatura nella generazione del parametro fff, come vedremo in seguito. **Power rappresenta anche il numero di ottave occupate dall'evento sonoro corrente, intorno alla frequenza base freq, cioè in sostanza space è lo spazio frequenziale occupato dall'evento**: questo intervallo frequenziale sarà poi diviso in 9 o 18 note.
Divx	**Rappresenta il numero di note in cui dividere lo spazio frequenziale occupato dall'evento e quindi le power ottave in questione**: in realtà nel momento della chiamata questo parametro sarà sempre uguale a 9, e verrà raddoppiato eventualmente durante INHARM
Dense	È il parametro inserito da input attraverso cui si può scegliere se attuare la divisione dello spazio frequenziale occupato dall'evento in 9 o in 18 intervalli

La procedura fa uso inoltre di alcune **variabili di tipo OWN**, cioè allocate staticamente al momento della compilazione.
Queste variabili sono proprie della procedura, e ciò significa che mantengono il loro valore a ogni chiamata, senza poter essere modificate nei momenti in cui la procedura è inattiva.
All'inizializzazione del programma vengono poste a zero tutte le variabili, e quindi anche quelle di tipo OWN.

In uscita la procedura passa il valore

$$scale\_freq = space^{num/divx}$$

dove il parametro num (intero di valore compreso tra 0 e 9 o tra 0 e 18) viene calcolato in INHARM secondo un procedimento che verrà spiegato successivamente, e assume valori diversi a ogni chiamata della procedura, e quindi per ogni elemento generato.

Valendo

$$space = ratio^{power}$$

si ha che il valore generato in uscita sarà

$$scale\_freq = (ratio^{power})^{num/divx} = ratio^{\frac{power \cdot num}{divx}}$$

Questo valore sarà passato nella procedura EVENT2 alla variabile fff, e definirà il coefficiente moltiplicativo per la creazione della nota di interesse a partire dalla frequenza di base freq.
È importante notare che questo valore cambia di elemento in elemento, nel susseguirsi delle chiamate a INHARM, al variare di num.

Nel caso di **power=1**, per **divx=9**, si ha che

$$scale\_freq = ratio^{\frac{num}{divx}}$$

e quindi questo caso rappresenta la possibilità di effettuare spostamenti di una singola nota all'interno dell'ottava, attraverso prodotto fff*freq.

Se **power** invece è **diverso da 1**, rappresenta un coefficiente di pesatura, che definisce quanto peso ha num nello spostamento delle frequenze dalla frequenza base: power positivo comporterà che un aumento di num di un'unità comporti un aumento di power note nel prodotto fff*freq rispetto alla frequenza base, mentre power negativo provocherà spostamenti di fff*freq verso il basso di |power| note in corrispondenza di un incremento unitario di num.

Un'altra interpretazione di power tiene conto del fatto che questo parametro, come già detto, può essere anche inteso come numero di ottave occupato da un evento, sopra o sotto la frequenza di riferimento: questo numero di ottave viene diviso in 9 o 18 divisioni, creando al variare di num le possibili note generate nell'evento.

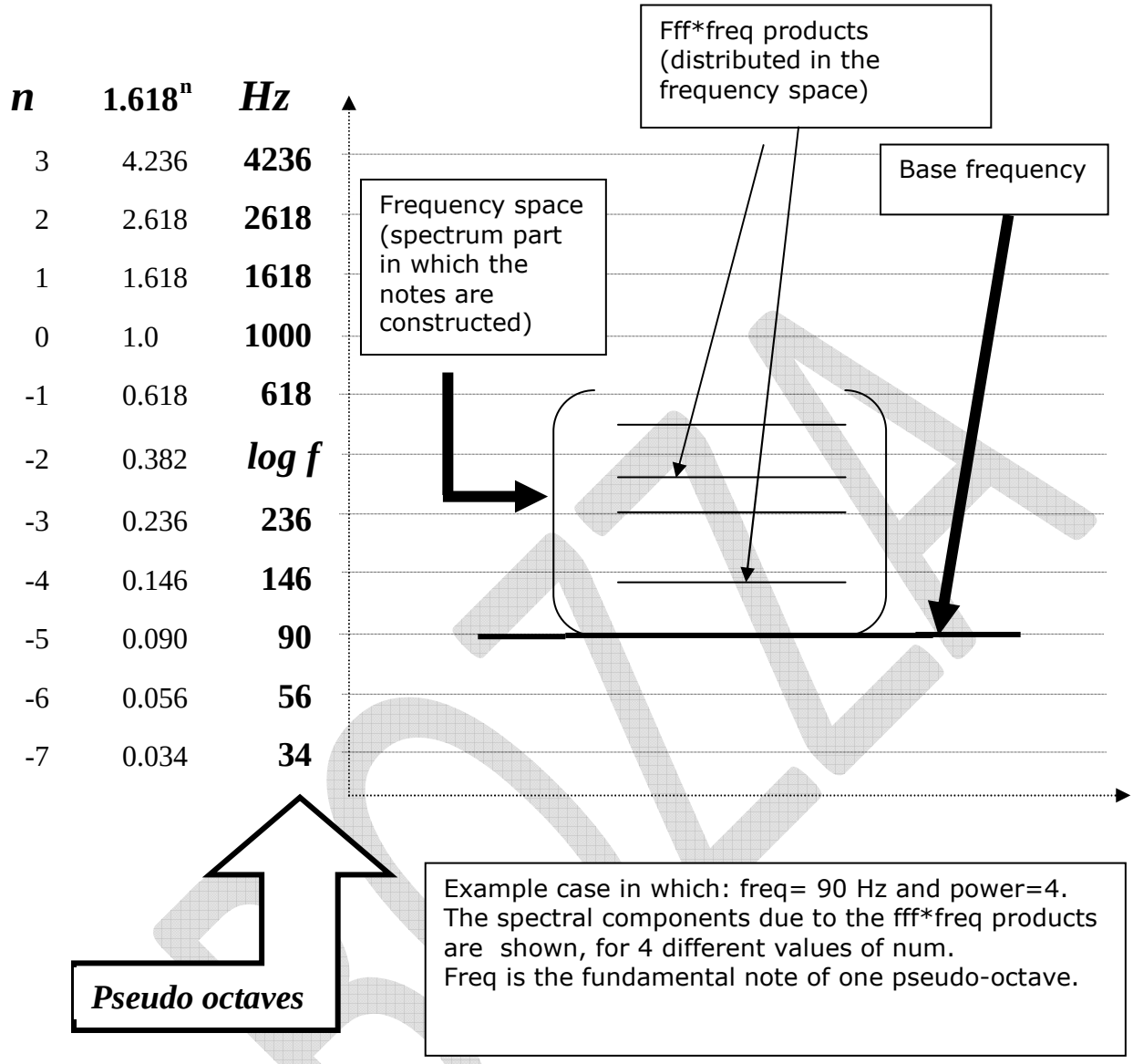
La variabile fff, quindi, serve per il calcolo della nota a cui si fa riferimento a partire dalla fondamentale dell'ottava, e assume un valore diverso a ogni chiamata di INHARM, a causa del fatto che il valore di num varia a ogni avvio, secondo una tabella definita più avanti.
Num*power quindi rappresenta lo spostamento della nota suonata rispetto alla fondamentale dell'ottava, mentre power rappresenta una misura dell'ampiezza dell'intervallo frequenziale minimo che può esserci tra due note dell'evento in questione.

È possibile dare un'interpretazione della variabile space: questo parametro descrive l'entità della spaziatura tra due note generate per valori consecutivi di num, definendo l'estensione spettrale possibile di ogni evento, grazie anche al valore di freq, frequenza base.

Infatti la nota più lontana dalla frequenza base che può essere generata in un evento è quella che si ottiene quando l'esponente di space è unitario, e ha frequenza

$$space \cdot freq = ratio^{power} \cdot freq$$

Space quindi rappresenta la dimensione dello spazio occupato nello spettro dall'evento (spazio frequenziale occupato dall'evento), dividendo il quale in 9 o 18 parti, si possono collocare le note.
È importante notare che nei casi in cui |power|=1, cioè nei casi in cui gli eventi spaziano su una sola pseudo-ottava, Chowning ha usato solo divisioni in 9 note, in modo che non ci siano mai più di 9 note nella stessa ottava, e che non si generino suoni troppo rumorosi.
Divisioni più fini invece sono utilizzate quando si tratta di eventi che occupano più pseudo-ottave.



Questa figura chiarisce i concetti di freq, fff e space per un caso particolare in cui freq=90 Hz e power=4.
Si notano i prodotti fff*freq tracciati per quattro diversi valori di num: si riconosce anche (tra parentesi) lo spazio frequenziale dell'evento, definito attraverso la variabile space.

A questo punto rimane da chiarire come vengano generati i valori successivi di num, utilizzati per definire i parametri fff, cioè le frequenze dei vari suoni generati.
Dal codice di INHARM si vede che durante la prima chiamata (quando cioè la variabile booleana onc che viene impostata nel programma principale è vera) INHARM esegue una serie di istruzioni di inizializzazione.

La prima cosa che fa è impostare i valori di una tabella (di tipo OWN) avente dieci elementi, con i numeri da zero a 9: grazie a questi numeri si

ricaverà poi il valore di num, sempre compreso tra zero e 9, da utilizzare nella definizione della nota.

La tabella è la seguente:

```
ftab[0]←0
ftab[1]←4
ftab[2]←3
ftab[3]←8
ftab[4]←6
ftab[5]←5
ftab[6]←7
ftab[7]←9
ftab[8]←2
ftab[9]←1
```

Per alcuni blocchi è stata usata una tabella leggermente diversa, nella quale i valori di ftab[3] e ftab[7] risultavano scambiati.

Le variabili OWN j,ik sono poste a -1, mentre la variabile perm è posta a 1. Di queste variabili si capirà il significato in seguito.
A questo punto la variabile onc viene posta falsa, e l'inizializzazione non verrà più eseguita fino al prossimo blocco (o almeno fino al prossimo evento, come si vedrà parlando della variabile CCKK).

Supponiamo per adesso che dense sia uguale a 0, e cioè che si voglia dividere lo spazio frequenziale occupato dall'evento (di ampiezza uguale a power ottave) in 9 note.

La variabile k viene incrementata di uno e si definisce la variabile intera kk, complemento a 9 di k.

Si decide poi, in base al valore di perm (a cui possiamo dare il significato di variabile di permutazione) come generare num, tra i quattro casi seguenti:

`Se perm=1`	`Num viene posto uguale a ftab[k]`
`Se perm=2`	`Num viene posto uguale a ftab[kk]`
`Se perm=3`	`Num viene posto uguale a STRIA-ftab[k]=9-ftab[k]`
`Se perm=4`	`Num viene posto uguale a STRIA-ftab[kk]=9-ftab[kk]`

A questo punto, se dense=0, si è completato il calcolo di num, e la procedura può trovare il valore scale_freq da passare in uscita.

Quando k viene incrementata di uno, prima della decisione del valore di num in base a perm, si può verificare il caso che k diventi maggiore di 9, creando così un'uscita dal campo di valori validi per la lettura della tabella.
In questo caso la procedura riporta k a zero, e se dense è uguale a zero (nove divisioni) aumenta perm di una unità (se perm supera 4 lo si riporta a 1).

A questo punto possiamo riassumere il funzionamento di INHARM nel caso di dense=0: a partire da k=0 comincia la generazione dei

numeratori dell'esponente che definisce scale_freq e ne genera uno a ogni chiamata.

Perm parte da 1, e quindi per k che varia da 0 a 9, i valori generati saranno i valori della tabella, letti in ordine.
Quando k supera 9 viene riportato a 0, e essendo dense=0, perm viene aumentato di uno: k quindi tornerà a variare da 0 a 9, ma essendo cambiato il valore di perm la lettura della tabella avverrà secondo l'indice kk=9-k, e quindi i successivi dieci valori di num saranno i valori contenuti nella tabella, letti però con indice decrescente da 9 a 0.
I successivi venti valori di num saranno i complementi a 9 dei venti valori precedenti: ciò risulta evidente dalla definizione di num per perm = 3,4.

I valori di num generati, e quindi anche i valori di scale_freq passati in uscita, presentano quindi una periodicità con periodo di ripetizione 40 (num e k, si ricorda, sono di tipo OWN, e quindi manterranno il valore a ogni chiamata, così come perm e j).
Nessun evento genera 40 strumenti, quindi in nessun evento si avranno successioni di strumenti aventi gli stessi valori di fff di successioni di strumenti già suonati, visto che il valore di fff cambierà per ogni elemento, e l'intero insieme di valori di fff si ripete con periodicità uguale a 40.
Ciò evita ogni possibilità di monotonia melodica all'interno degli eventi: le successioni di suoni saranno sempre diverse, pur potendo toccare le stesse note.

Rimane da analizzare cosa succede per **dense diverso da zero**, quando cioè sia stato impostato un numero di 18 divisioni per lo spazio frequenziale occupato dall'evento.
In questo caso il valore di perm non viene variato quando k supera 9, e la variabilità nella generazione dei valori di num e quindi nella generazione delle frequenze è ottenuta in un altro modo.
Il valore di perm quindi è quello presente alla chiamata di INHARM, e non viene variato durante la generazione del valori di num, quindi si userà sempre la stessa regola anche nelle successive chiamate (essendo num di tipo OWN).

Dividendo lo spazio frequenziale in 18 spazi si dovrà porre il denominatore a 18, con l'istruzione `divx←divx*2`, e anche num dovrà variare tra 0 e 18: in determinate condizioni quindi si aumenterà di 9 il valore di num, ottenendo così la funzionalità richiesta.
Dopo la scelta di num in base al valore di perm sono poste delle istruzioni, eseguite solo se dense>0: la prima riga raddoppia il denominatore, mentre le altre aumentano di 9 il valore di num nei casi in cui:

- k è multiplo di 3 e j è maggiore di zero
- k non è multiplo di 3 e j è minore di zero

La variabile j, di cui finora non si è parlato, varia il suo segno ogni volta che k supera 9.

A questo punto si può spiegare anche il funzionamento di INHARM anche per dense>0.

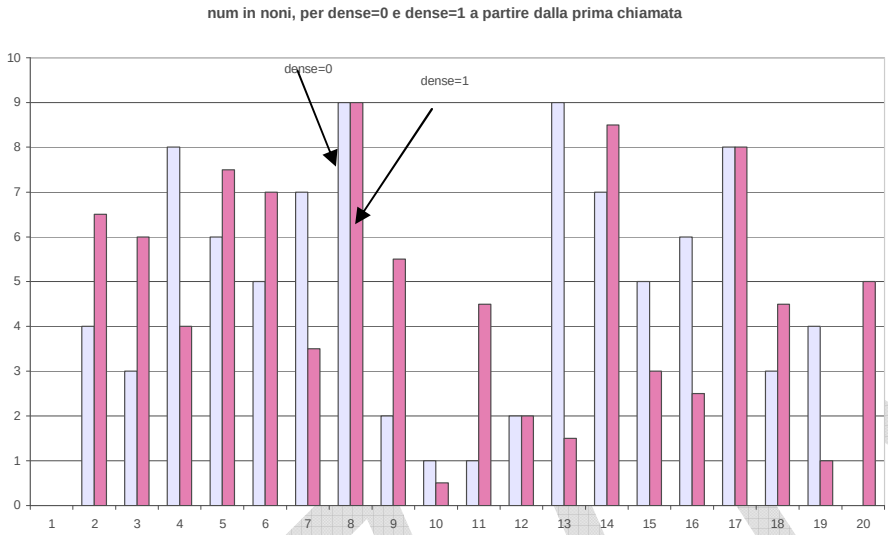
Supponiamo che si tratti di una serie di venti chiamate a partire dal primo avvio, condizione non restrittiva per capire il funzionamento della procedura: perm varrà allora 1, e k varierà due volte tra 0 e 9, generando i valori di num corrispondenti a due letture consecutive della tabella; nel primo passaggio di k da 0 a 9 j è uguale a -1, mentre nel secondo vale +1. I valori di num in uscita saranno:

k	J	Num (18 div)	Num (9 div)
0→ multiplo	-1	0	0
1	-1	4+9=13	6.5
2	-1	3+9=12	6
3→ multiplo	-1	8	4
4	-1	6+9=15	7.5
5	-1	5+9=14	7
6→ multiplo	-1	7	3.5
7	-1	9+9=18	9
8	-1	2+9=11	5.5
9→ multiplo	-1	1	0.5
0→ multiplo	1	0+9=9	4.5
1	1	4	2
2	1	3	1.5
3→ multiplo	1	8+9=17	8.5
4	1	6	3
5	1	5	2.5
6→ multiplo	1	7+9=16	8
7	1	9	4.5
8	1	2	1
9→ multiplo	1	1+9=10	5

I valori successivi eventualmente generati sarebbero una ripetizione di questi, con periodo di ripetizione uguale a 20: nessun evento ha più di venti elementi, e tenuto conto che le chiamate ricorsive (come si vedrà in seguito) generano elementi solo con dense=0, si ha che nessuno strumento dello stesso evento avrà lo stesso valore di fff di un altro strumento appartenente allo stesso evento.
Si nota che num assume nel giro dei venti valori successivi di k, tutti i possibili valori tra 1 e 18, rendendo così possibile la divisione delle ottave in 18 intervalli.

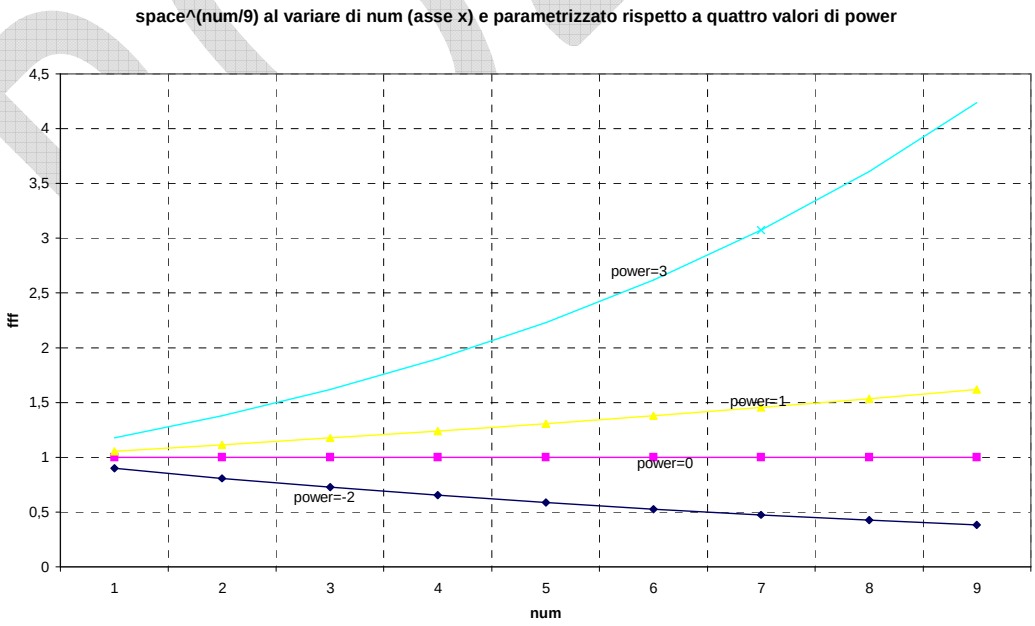
L'ultima colonna della tabella precedente rappresenta il valore di num, inteso considerando la divisione delle ottave in 9 note (9*num/18): in questo modo risulta chiaro che si vanno a considerare anche i mezzi toni, e si può valutare l'andamento di num nelle successive chiamate di INHARM, confrontandolo con il caso di dense=0.

Nel grafico seguente si nota per le prime venti chiamate a partire dal primo avvio di INHARM la differenza tra i numeratori in noni generati dai due casi dense=0 e dense=1.
All'opzione sul valore di dense corrisponde quindi una scelta sulla linea melodica seguita dal prodotto fff*freq.



Per una maggiore comprensione del funzionamento di INHARM, torniamo a considerare il caso di dense=0.
Il grafico successivo rappresenta i valori di fff, calcolati al variare di num tra 0 e 9, per diversi valori di power nell'espressione

$$space = ratio^{power}$$

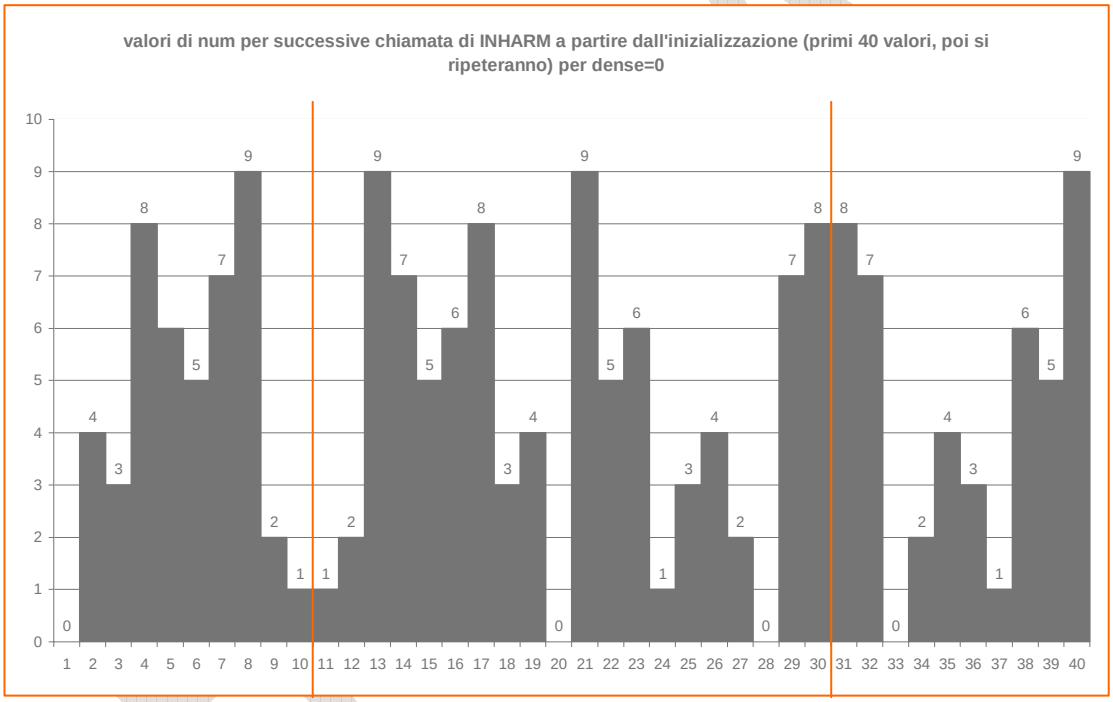


Si nota che il peso di num dovuto ai diversi valori di power, varia nei vari casi: per power = 0, space=1 e quindi fff è unitaria; sarà quindi la frequenza base a determinare l'altezza dei suoni generati.

Per power<0 fff risulterà < 1, comportando uno spostamento verso il basso, rispetto alla frequenza base al variare di num.
Per power>0 fff sposterà le frequenze verso l'alto.

Si nota che a power=1 corrisponde uno spostamento sulle nove note dell'ottava definita da freq nel prodotto fff*freq, visto che i valori di fff variano tra 1 e 1.618 al variare di num tra 1 e 9.

Il grafico seguente, invece, rappresenta i quaranta valori consecutivi di num calcolati per dense=0, supponendo che INHARM sia stata chiamata per la prima volta.
Si notano le linee di simmetria in corrispondenza delle letture della tabella per valori di k e di kk (ricordiamo che la lettura di ftab[kk] da valori speculari a quelli letti per ftab[k]).



Si nota la grande variabilità di num nelle varie chiamate, che rende possibile lo spazzolamento di molte frequenze nella generazione dei suoni.

Nella generazione di un nuovo evento si può scegliere di **re-inizializzare** la procedura INHARM, ricostruendo la tabella originale e ripartendo dai valori base di num, perm e j, oppure di continuare con i valori lasciati in memoria all'ultima chiamata della procedura (nell'ultimo elemento costruito).

Ciò equivale a decidere se **ricominciare** a seguire il grafico precedente dall'inizio, **o continuarlo** dal punto in cui si era arrivati nella lettura precedente.

Nella generazione delle frequenze di portante e modulanti attraverso i prodotti del tipo di

$$c = fff \cdot freq \cdot f_c$$

avviene un fatto di particolare importanza: sia le frequenze della portante sia quelle delle modulanti sono moltiplicate dal coefficiente fff*freq.
Ciò comporta che al variare di fff*freq non si ha una semplice traslazione dello spettro, dovuta alla variazione della portante, ma **si ha anche un restringimento o un allargamento delle righe spettrali, dovuto al fatto che insieme alla portante variano anche le modulanti**: a un aumento della portante per una crescita di fff corrisponderà un allargamento dell'intero spettro (visto che si allargano gli intervalli tra le righe spettrali) e a un calo della portante corrisponderà uno spettro più stretto (le righe spettrali si avvicinano).

3.4 La procedura PROPORTION e la generazione dei tempi

Questa procedura serve per calcolare alcuni parametri legati alle **caratteristiche temporali** dei suoni all'interno dei vari eventi.

Accetta in ingresso tre parametri interi, e restituisce un valore reale: prima di scendere nel dettaglio sul funzionamento della procedura, conviene capire a cosa servono i dati da essa generati.

Bisogna ancora una volta considerare ciò che succede nella procedura EVENT2.che genera gli eventi, e quindi i singoli suoni.
Per adesso ovviamente si considereranno soltanto gli aspetti di questa procedura legati a PROPORTION, trascurando gli altri, che verranno considerati in una sezione successiva.

Appena EVENT2 viene chiamata, calcola un valore, denominato **sc_prop**, che esprime un **fattore di pesatura temporale globale** e verrà usato per il calcolo dei tempi di inizio e fine dei suoni, oltre che per la determinazione dei parametri in ingresso alle chiamate ricorsive.

Per calcolare questo valore EVENT2 considera il **numero di elementi sonori che dovrà generare nell'evento**(valore contenuto nella variabile elements), e genera un parametro di somma (sum), sommando ciclicamente il valore di PROPORTION corrispondente a successive terne di parametri in ingresso (una per ogni elemento sonoro da generare) a sum stesso.
Così si ottiene una somma che definisce il peso temporale dell'intero insieme di elementi (senza considerare le ricorsioni): l'inverso di questo valore è sc_prop.

Nella generazione dei singoli elementi (strumenti), EVENT2 chiama nuovamente PROPORTION, per calcolare il peso temporale di ogni elemento: per ogni elemento sonoro viene generato il parametro prop=PROPORTION(terna di ingresso)*sc_prop, che è il prodotto del valore di PROPORTION calcolato in corrispondenza della terna corrispondente all'elemento corrente e del fattore di pesatura globale.

I valori di PROPORTION in uscita sono tali che **ogni cinque valori la loro somma sia unitaria**, e vengono calcolati in modo che **ogni 5 elementi si ripetano** (senza considerare le ricorsioni).

Il valore di prop calcolato per ogni elemento ne può definire la durata, e serve per determinare il tempo di inizio dello strumento successivo, per scegliere su che suoni intraprendere le chiamate ricorsive e per pesare durate e attacchi dei suoni generati ricorsivamente.

Quanto detto è sufficiente per comprendere la grande utilità di questa procedura, e verrà ripreso più avanti quanto riguarda la procedura EVENT2.

PROPORTION accetta in ingresso:

which	Parametro intero usato per definire due metodi di scelta del risultato della procedura, a seconda che il suo valore sia 0 o 1. Nel momento in cui EVENT2 viene chiamata, la sua variabile which viene inizializzata a 0 (questa variabile è allocata dinamicamente, essendo EVENT2 ricorsiva, e quindi sarà anche inizializzata a ogni chiamata), e poi viene usata come parametro per PROPORTION, che si troverà sempre a funzionare con which=0. Un solo modo di scelta dei parametri sarà quindi definito
col	Valore intero che sarà utilizzato col significato di colonna di lettura di una tabella
propcnt	Valore intero che sarà utilizzato col significato di riga di lettura di una tabella

Allo stesso modo di INHARM, PROPORTION riesce a **riconoscere la prima chiamata dalle altre**, attraverso il parametro once, vero se si tratta della prima chiamata, che viene impostato all'avvio del programma principale.

Alla prima chiamata procede quindi all'inizializzazione delle variabili di tipo OWN, tra cui:

L'array OWN fib[], inizializzato secondo i valori Fib[1]←0.06 Fib[2]←0.1 Fib[3]←0.16 Fib[4]←0.26 Fib[5]←0.42 Fib[6]←0.149 Fib[7]←0.241 Fib[8]←0.39 Fib[9]←0.387	Questo array contiene i primi valori di due **successioni di Fibonacci**, e un ulteriore valore, fib[9]=0.387. La parte di questo array che viene usata in Stria è costituita dai **primi cinque elementi**, che hanno due proprietà matematiche particolari: ▪ la loro **somma** è **unitaria** ▪ rappresentano in approssimazione alcune **potenze della sezione aurea**, a partire da $1.618^{-6} \cong 0.06$, $1.618^{-5} \cong 0.1$…. **Queste relazioni non sono esatte**, perché se lo fossero state la somma dei primi cinque termini non sarebbe stata unitaria, fattore importante per la pesatura dei tempi degli elementi nei vari eventi I rimanenti valori nella tabella non sono usati in Stria: la generazione degli eventi

	in Stria avviene attraverso la procedura EVENT2, ma dal codice a disposizione si può intuire che originariamente ci fosse la possibilità di scegliere anche altre due procedure per generare gli eventi, le procedure EVENT0 e EVENT1. Si può ritenere (non avendo prove contrarie), che chiamate a PROPORTION che utilizzassero questi valori di fib[i] e valori di which diversi da zero avvenissero in seguito alla generazione di questi tipi di eventi.
I valori di una tabella OWN di riferimenti: tab[1,1]←4 tab[2,1]←1 tab[3,1]←2 tab[4,1]←5 tab[5,1]←3 tab[6,1]←5 tab[7,1]←8 tab[8,1]←7 tab[9,1]←3 tab[1,2]←3 tab[2,2]←5 tab[3,2]←1 tab[4,2]←2 tab[5,2]←4 tab[6,2]←7 tab[7,2]←6 tab[8,2]←8 tab[9,2]←9	Si sono elencati tutti i valori della tabella, per rendere più esplicito quanto succede in PROPORTION. Questa matrice 9x2 mappa ogni coppia (riga,colonna) in un numero da 1 a 9, che verrà utilizzato per calcolare il risultato. Per ragioni che saranno evidenti nella sezione riguardante EVENT2, in Stria vengono lette solo le prime cinque righe, che fanno riferimento solo ai primi cinque elementi dell'array fib[].
l=9, k=-1 ,j=-1	Variabili intere di tipo OWN, usate nel caso di which diverso da zero, e quindi non usate in Stria.

Per il calcolo del risultato da fornire in uscita, PROPORTION distingue i casi di which=0 e di which=1.

Se which=0 fornisce in uscita il valore fib[tab[propcnt,col]], cioè il valore contenuto nella posizione dell'array fib[] indicata dal valore contenuto alle coordinate (propcnt,col) in tab[].

Se which=1 viene mandato in uscita il valore fib[propcnt], cioè si esegue una lettura diretta dall'array, senza passare attraverso la tabella.

Successivamente viene cambiato il segno di j, si decrementa l di uno e si genera un nuovo valore di propcnt, dato dall'espressione propcnt=propcnt+l*j
Se nel diminuire l di un'unità si ottiene l<1, allora viene riportato il valore di l a 9, e viene posto propcnt uguale a 10.

Questo secondo caso non viene usato in Stria, e quindi dai listati non se ne comprende l'esatto funzionamento.

3.5 La procedura AZIM e la posizione angolare dei suoni

La **spazializzazione** in Stria è resa attraverso la definizione **di un angolo di riferimento per ciascun elemento**: a ogni strumento è quindi associato un angolo, da cui il suono deve sembrare provenire (altri parametri di spazializzazione già citati sono il riverbero e DIS_SCALE, la scala delle distanze).

Gli altoparlanti, rispetto alla direzione frontale sono posti agli angoli 45°, 135°, 225° e 315°, e per ciascuno di essi viene calcolato un fattore di moltiplicazione dell'ampiezza del segnale in ingresso legato a ciascun suono: l'ampiezza del suono di ogni strumento viene cioè mandata verso i quattro altoparlanti moltiplicata per quattro diversi parametri direzionali.

Questi parametri sono calcolati dalla procedura AZIM.
Questa riceve in ingresso un valore reale, la variabile deg, e non restituisce valori in uscita, ma **agisce direttamente sulle variabili globali** rappresentate dai parametri di spazializzazione PAR[21], PAR[22], PAR[23] e PAR[24], impostandone i valori in riferimento a ogni suono.
Sarà la chiamata della procedura **WRITE** da parte di EVENT2 che, finita la generazione di tutti i PAR[], manderà in output al file .SCR anche i dati sulla spazializzazione legati allo strumento in questione.

Il metodo di calcolo dei parametri per la quadrifonia è abbastanza semplice: **consideriamo per esempio il parametro PAR[23],** che esprime il fattore di ampiezza dell'altoparlante a 225°.

Questo parametro è diverso da zero soltanto per valori dell'angolo di riferimento compresi tra 135° e 315°, e ciò intuitivamente si spiega con il fatto che suoni provenienti da altre direzioni saranno per forza generati da altri altoparlanti.

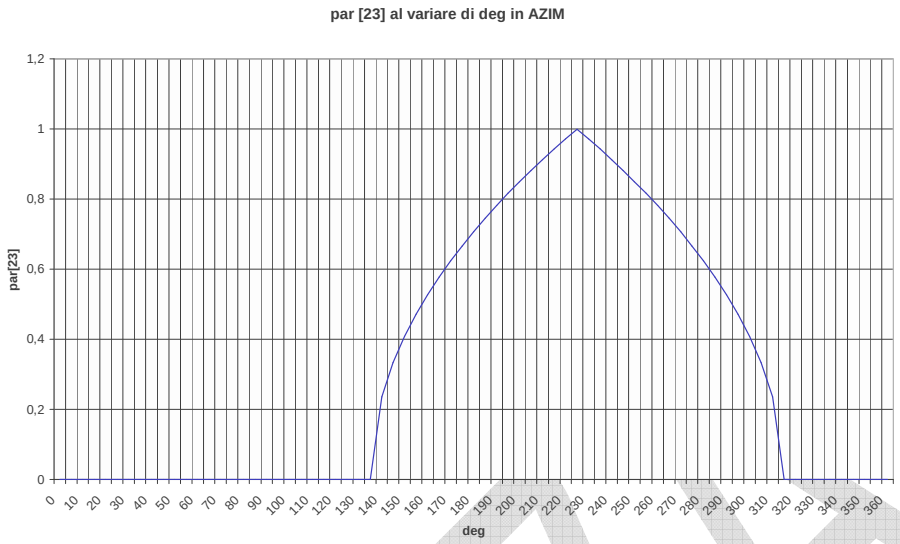
Per la determinazione del suo valore si divide poi l'intervallo in cui è diverso da zero in due parti, ponendo per 135<deg<225

$$PAR[23] = \sqrt{\frac{\deg - 135}{90}}$$

e per 225<deg<315

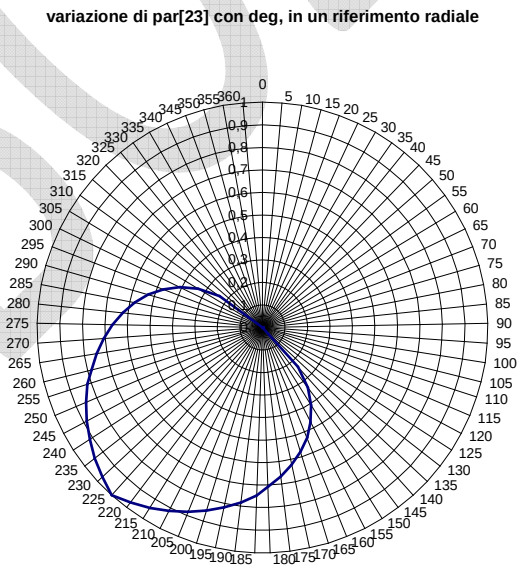
$$PAR[23] = \sqrt{1 - \frac{\deg - 225}{90}} = \sqrt{\frac{315 - \deg}{90}}$$

La variazione di questo parametro è messa in evidenza dal grafico seguente:

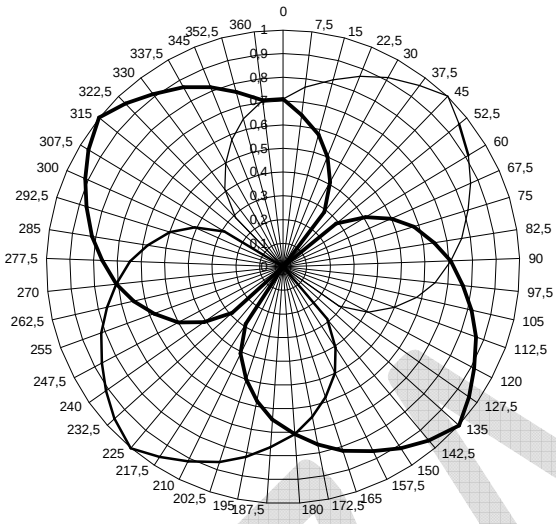


I metodi di determinazione degli altri parametri sono analoghi, e i grafici relativi sono curve uguali a questa, traslate intorno all'anglo di collocazione dell'altoparlante interessato.
Ogni altoparlante diffonderà quindi con amplificazione unitaria i suoni provenienti dalla sua direzione, mentre il fattore di scala calerà allontanandosi da essa.

Visti in un riferimento radiale, i quattro parametri assumono la caratteristica forma a lobi, rappresentata nel grafico seguente per PAR[23]



E nel seguente per tutti i parametri



3.6 La procedura EVENT2: la generazione degli eventi

La generazione degli eventi descritti in ogni blocco da sezioni separate di input, è affidata alla procedura EVENT2.
Nel programma principale, come già anticipato, si nota la possibilità di scegliere il **tipo di evento** da generare, attraverso la scelta del parametro event_num, che può assumere i valori 0,1,2.
La chiamata di un evento corrisponde alla chiamata di una tra le tre diverse procedure: EVENT0, EVENT1 e EVENT2.
Stria è stata generata solo a partire da eventi di tipo 2, e per questo gli eventi di tipo 0 e 1 non sono descritti dai listati di cui siamo in possesso.

EVENT2 è la procedura che a partire dai parametri che le vengono passati in ingresso, genera i singoli elementi, cioè le singole voci strumentali, calcolando per ognuna di esse i trenta parametri da inviare attraverso WRITE al file .SCR, cioè alla partitura.

È una procedura di tipo **ricorsivo**, e quindi tutte **le sue variabili sono allocate dinamicamente a ogni chiamata**: non esisistono quindi variabili di tipo OWN, e a ogni chiamata tutti i parametri sono **inizializzati a zero.**
Ogni ricorsione genererà ulteriori elementi (suoni figli), che si sovrapporranno a quelli generati fuori dal nesting ricorsivo (suoni padri).

Cominciamo l'analisi di questa procedura dall'esame dei parametri in ingresso, **passati per valore**:

beg	Tempo di inizio dell'evento da generare, variabile reale
dur	Durata dell'evento, reale
At_dur	Durata dell'attacco dell'evento, cioè dell'intervallo di tempo in cui durante l'evento possono nascere nuovi suoni. È l'attacco DELL'EVENTO, non in relazione con i tempi di attacco dei suoni
elements	Numero di elementi da generare, senza tenere conto della ricorsione (si tratta quindi del numero di suoni padri)
freq	Frequenza base (per il significato di questo parametro si veda la sezione riguardante INHARM)
col	Parametro intero usato per le chiamate a PROPORTION
divx	Parametro che serve a INHARM per generare le frequenze di scala: vale 9 in ogni chiamata di EVENT2, sia da programma principale, sia ricorsivamente
propcnt	Parametro intero per le chiamate a PROPORTION, usato per scorrere le tabelle
dense	Variabile reale per la scelta del numero di note in un'ottava, usata da INHARM
atswitch	Variabile reale usata come interruttore: se vale 1 i tempi di attacco e di decadimento dei vari strumenti sono calcolati in base alla frequenza della nota generata, altrimenti sono calcolati in modo proporzionale ai parametri ricevuti da input relativi a attacchi e decadimenti.
space	È uguale a $ratio^{power}$, e come già visto nella sezione riguardante INHARM, esprime una quantificazione della spaziatura tra le frequenze delle singole note generate, e dell'estensione spettrale dell'evento
nextbeg	Tempo di inizio del suono successivo a quello corrente
Sc_prop	Parametro di pesatura globale, utilizzato per la divisione del tempo tra i vari suoni
rev	Percentuale di riverbero da applicare ai suoni

Prima di vedere come la procedura imposta i parametri, è opportuno fare un elenco completo dei parametri che definiscono ogni singolo suono (elemento):

PAR[1]	Tempo di inizio del suono, in secondi
PAR[2]	Durata del suono, in secondi
PAR[3]	Ampiezza del suono
PAR[4]	Frequenza della portante, in Hz
PAR[5]	Frequenza della prima modulante, in Hz
PAR[6]	Minimo dell'indice di modulazione della prima modulante, I_{1min}
PAR[7]	Massimo dell'indice di modulazione della prima modulante, I_{1max}
PAR[8]	Frequenza della seconda modulante, in Hz
PAR[9]	Minimo dell'indice di modulazione della prima modulante, I_{2min}
PAR[10]	Massimo dell'indice di modulazione della prima modulante, I_{2max}
PAR[11]	Tempo di attacco del suono, in secondi
PAR[12]	Tempo di decadimento del suono, in secondi
PAR[13]	Percentuale di skew
PAR[14]	Parametro legato alla spazializzazione: viene usato per proiettare i suoni nello spazio attraverso il rapporto tra suoni diretti e riverberati
PAR[15]	Percentuale di riverbero
PAR[16-20]	Non usati in Stria
PAR[21-24]	Coefficienti di ampiezza dei segnali sui quattro altoparlanti
PAR[25,26]	Non usati in Stria
PAR[27]	Tipo di inviluppo di ampiezza del suono modulato
PAR[28]	Tipo di funzione di skew
PAR[29]	Tipo di inviluppo per l'indice di modulazione della prima modulante
PAR[30]	Tipo di inviluppo per l'indice di modulazione della seconda modulante

Nel programma principale è presente la variabile globale RHY, che durante l'intera esecuzione di Stria è usata come costante, posta uguale a 5.
Questo valore è scelto in modo tale di condizionare, nel codice di EVENT2, il funzionamento di PROPORTION, facendo in modo che prelevi i dati solo da certe posizioni dell'array fib[].

Ricevuto in ingresso il numero elements di elementi padri da generare (cioè elementi non ricorsivi), EVENT2 confronta questo numero con RHY (=5), e se è diverso comincia la valutazione del parametro **sc_prop**, per la pesatura dei tempi di inizio dei suoni.
Per fare ciò utilizza il parametro ausiliario sum, che parte da zero, e definisce un ciclo sulla variabile cntx, che va da 1 a elements.
A ogni ciclo calcola il resto della divisione di cntx per 5, e lo usa come parametro di riga per PROPORTION, i cui altri due parametri sono la colonna col, ricevuta alla chiamata, e il valore which, variabile locale di EVENT2, che viene allocata e inizializzata a ogni chiamata della procedura, e quindi non essendo modificata prima di questa parte di codice, vale zero.

Si nota quindi che con PROPORTION si vanno a leggere solo i primi valori dell'array fib[], mentre gli altri non vengono utilizzati: l'indice di riga, infatti, sarà sempre minore di 5.

La variabile sum alla fine del loop contiene la somma dei valori di PROPORTION calcolata sulla colonna col, al variare di cntx secondo l'andamento 1,2,3,4,5,1,2,3,4,5,1,2,…: PROPORTION, calcolata su queste serie di valori, fornisce le successioni di numeri seguenti, a seconda della colonna

Col=1	Col=2
0.26	0.16
0.06	0.42
0.1	0.06
1.42	0.1
0.16	0.26
…si ripete	…si ripete

Si nota che ogni 5 valori su entrambe le colonne la somma vale 1.

Quindi, supponendo che a ogni elemento numerato da 1 a elements sia attribuito il peso calcolato da PROPORTION, alla fine del ciclo sum rappresenta la somma dei pesi di tutti gli elementi: i valori calcolati per le due colonne sono chiaramente diversi, a parte per elements multiplo di 5, ma per elements maggiore di 5 si possono considerare praticamente uguali.

Da sum si calcola

$$sc\_prop' = \frac{1}{sum} \cdot sc\_prop$$

che è il fattore che verrà usato per calcolare i pesi dei singoli eventi sul tempo totale.
Sc_prop è passato durante la chiamata a EVENT2, e per la prima chiamata (che genera i suoni padri) vale 1, mentre per le chiamate ricorsive sarà uguale al valore di sc_prop del suono padre che le ha generate.
Le chiamate ricorsive quindi avranno un fattore di pesatura in generale minore di quello dei suoni padri corrispondenti, visto che in generale elements è sempre maggiore o uguale a 5, e perciò sum è sempre maggiore o uguale a 1.

A questo punto, dopo aver mandato al file .SCR la stringa che segnala l'inizio dell'evento, EVENT2 comincia un ciclo iterativo, eseguito secondo la variabile cnt, che va da 1 a elements.
Vengono cioè generati, uno dopo l'altro, i vari strumenti e ne vengono mandati i parametri nel file di partitura (.SCR), attraverso WRITE.

Which viene impostata a 0, e tale rimane fino alla fine della procedura, imponendo a PROPORTION di leggere i valori attraverso la tabella, e non direttamente.

La procedura deve capire qual è il prossimo strumento da far suonare, tenendo conto che cercherà sempre di far suonare lo strumento libero di indice più basso: per fare questo inizializza a 1 la variabile nextins, indice

del prossimo strumento da suonare, e fa un ciclo sul numero di strumenti (26).

In questo loop valuta se il tempo di fine dello strumento relativo all'attuale valore di nextins (endtime[nextins], dove endtime[] è il vettore dei tempi di fine dei vari strumenti, come già detto) è maggiore del tempo di inizio del suono da generare, e in questo caso incrementa nextins di 1, visto che lo strumento in questione deve ancora finire di generare il suono, e non può essere riutilizzato.
Appena trovato uno strumento libero il valore di nextins rimane invariato fino alla fine del ciclo, e nextins conterrà l'indice del primo strumento libero.
Se si è superato il valore nextins=insnum=26, la procedura EVENT2 chiama GETOUT, che segnala l'eccesso di voci strumentali, chiude la partitura e esce dal programma.

Trovato qual è lo strumento che suonerà, bisogna definirne i parametri.

Si comincia dal calcolo di **prop**, per l'elemento corrente, attraverso l'espressione

$$prop = PROPORTION(which, col, propcnt) \cdot sc\_prop$$

Propcnt alla prima chiamata di EVENT2 vale 1, e quindi PROPORTION comincia la scansione delle righe della tabella, a partire dalla prima, per la prima chiamata (consideremo poi il caso delle ricorsioni).
La colonna impostata per la prima chiamata è la prima, visto che col passato per valore vale 0, e la procedura l'ha aumentato di uno dopo la chiamata (questo per chiamate non ricorsive).

A ogni elemento padre viene quindi associato un valore prop, che esprime il prodotto tra il peso temporale dell'elemento corrente e il fattore sc_prop che scala questo peso in rapporto al totale dei pesi: chiaramente si verifica che

$$\sum prop = \sum PROPORTION(...) \cdot sc\_prop = \frac{\sum PROPORTION(...)}{\sum PROPORTION(...)} = 1$$

questa proprietà deve valere, in modo che la distribuzione dei tempi durante il periodo di attacco dell'evento (periodo in cui durante l'evento possono cominciare i suoni) ricopra tutto l'intervallo definito dalla variabile at_dur in ingresso.

Il tempo di inizio del suono, contenuto nel parametro PAR[1], viene impostato in base al valore di nextbeg, valore che per il primo suono vale beg (inizio dell'evento), mentre per le successive viene calcolato in base alla durata dell'attacco dell'evento.
Si pone quindi

PAR[1]=nextbeg

E si calcola l'inizio del suono successivo (non ricorsivo) attraverso la formula

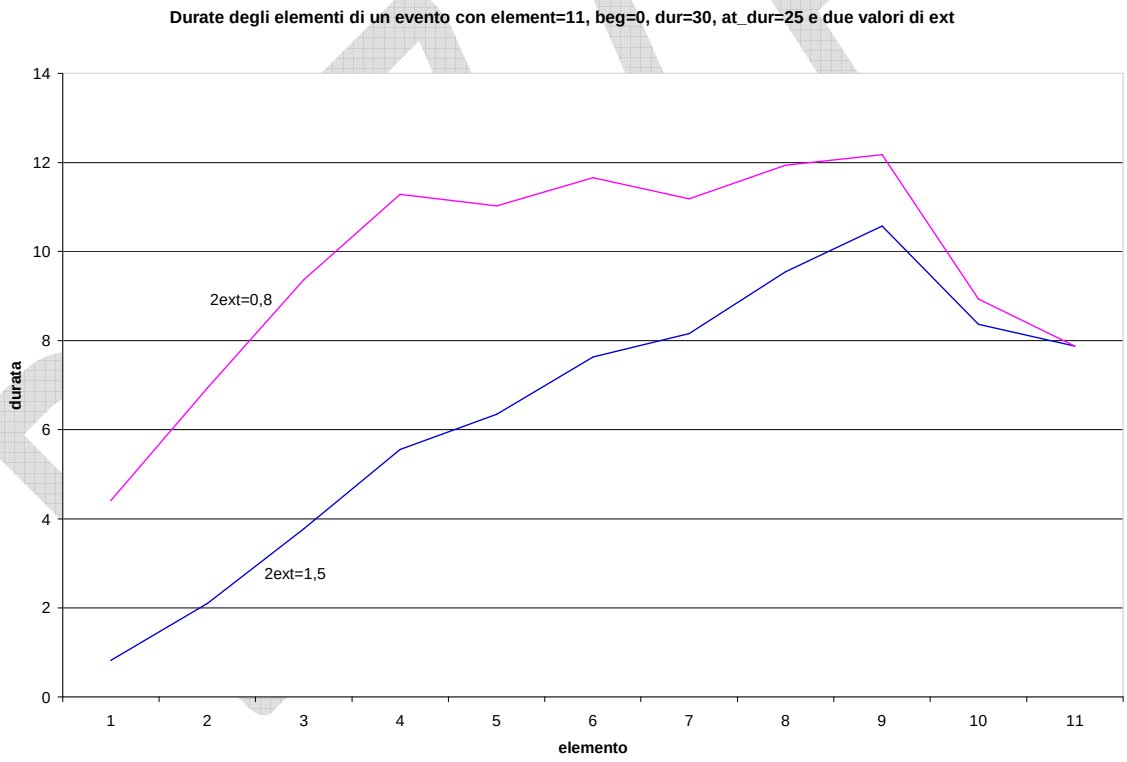
$$nextbeg' = nextbeg + prop \cdot at\_dur$$

Il prossimo suono inizierà quindi a una distanza dal precedente pari alla proporzione di at_dur data da prop, cioè pari alla parte di evento in cui possono nascere i suoni pesata da prop.

Per definire la durata del suono si utilizza la formula

$$PAR[2] = (beg + dur - PAR[1]) \cdot \left(\frac{cnt}{elements} \right)^{2 \cdot ext}$$

dove il primo termine tra parentesi rappresenta il tempo tra l'inizio del suono corrente e la fine dell'evento, e il secondo definisce un fattore esponenziale di pesatura dei tempi.
Il significato di questa espressione è il seguente: la durata di un suono viene determinata tenendo conto di quanto tempo rimane tra l'inizio del suono e la fine dell'evento, e scalando questa quantità di un fattore legato in modo esponenziale al numero dell'elemento che si sta generando.
La variabile ext è globale, e per default è posta a 0.75, ma può essere cambiata da evento a evento nel range tra 0.5 e 0.75 (in realtà in Stria si usano anche valori minori): variando ext si ottiene una diversa distribuzione delle durate, che valorizzerà i primi suoni per ext basso (v. grafico seguente).



La formula per calcolare PAR[2] in pratica aumenta il fattore di pesatura temporale al diminuire del tempo residuo, in modo che la durata rimanga consistente: infatti più elementi si sono già generati, meno tempo rimane, e più cresce cnt.
Dal grafico si intuisce che l'aumentare del termine esponenziale supplisce abbondantemente al calare del tempo residuo, tant'è che in questo caso quando questo si riduce, comunque la durata cresce.

Un'ulteriore condizione sulla durata è posta dal fatto che si desidera che il suono successivo cominci mentre è ancora presente il suono corrente: se quindi la somma di inizio e durata del suono

corrente non supera il tempo di inizio del suono successivo, **allora si impone la sovrapposizione,** con l'espressione

$$PAR[2] = (nextbeg - PAR[1]) \cdot 1.25$$

Se non c'è sovrapposizione la durata viene quindi posta uguale alla distanza tra gli inizi dei due suoni, moltiplicata per 1,25, in modo che ci sia sovrapposizione.

A questo punto si può definire il tempo di fine dello strumento corrente (nextins), ponendolo pari alla somma di inizio e durata, secondo

$$endtime[nextins] = PAR[1] + PAR[2]$$

Il prossimo spezzone di procedura è dedicato al **calcolo delle frequenze** e dei parametri di modulazione, e incomincia con la chiamata alla procedura INHARM, per generare il valore di fff (per l'elemento corrente), attraverso l'espressione

$$fff = INHARM(space, divx, dense)$$

dove space rappresenta il valore della frequenza di spaziatura, e vale

$$space = ratio^{power}$$

e viene direttamente dall'input dell'evento, così come dense e divx (=9).

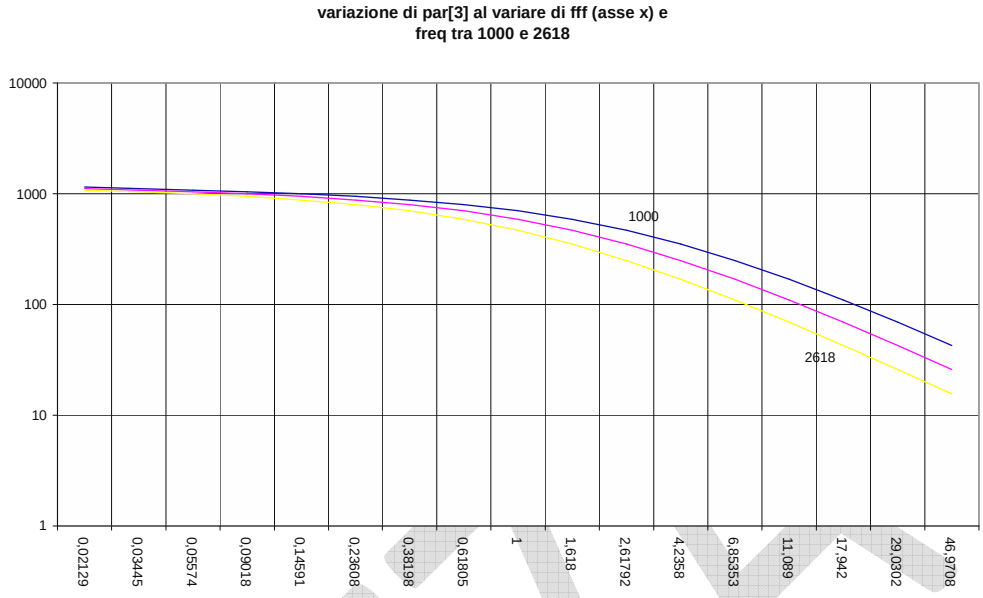
Si è già visto come INHARM genera in fff la frequenza di scala, che verrà usata per definire le note da suonare, in base al prodotto per la frequenza di base.

La decisione dell'ampiezza del suono viene presa una volta conosciuta la frequenza del suono da generare, e ne dipende direttamente, attraverso l'espressione

$$PAR[3] = 400 \cdot \frac{10 - \log(fff \cdot freq)}{3 + \frac{fff \cdot freq}{1000}}$$

Il grafico seguente rappresenta la variazione dell'ampiezza (PAR[3]) al variare di fff (per num=9 e dense=0, cioè per le fondamentali di ogni ottava), parametrizzato rispetto a 3 valori di freq (1000, 1618 e 2618).
La funzione, di tipo passabasso, ha il massimo per frequenze basse (le minori possibili in Stria) intorno a 1000, e cala sia all'aumentare della frequenza base freq, sia all'aumentare della frequenza di scala fff.
In generale quindi cala anche all'aumentare della frequenza del suono generato.

Per frequenze piccolissime tenderebbe a infinito, ma queste frequenze non sono generate dal programma.



La procedura non conosce il valore di num utilizzato da INHARM, perché questa variabile non è accessibile: allora utilizzando l'espressione

$$num' = STRIA \cdot \frac{\log fff}{\log ratio} = STRIA \cdot \log_{ratio} fff = 9 \cdot \log_{ratio} ratio^{\frac{power \cdot num}{divx}} = power \cdot num$$

calcola il valore di num*power, dato che STRIA=divx=9.

Una volta calcolata fff, da freq e dai valori delle variabili FR[2,1], FR[2,2] e FR[2,3], che rappresentano i coefficienti moltiplicativi f_c, f_{m1} e f_{m2} di cui si è parlato nella sezione relativa a INHARM (relativi all'evento 2, visto che la forma generale di questi parametri è del tipo FR[event_num,i]) si possono calcolare le frequenze della portante e delle modulanti.

La portante è posta uguale a

$$PAR[4] = fff \cdot freq \cdot f_c$$

mentre la decisione della frequenza della prima modulante è basata sul valore di switch: con questo parametro si può infatti decidere di impostare PAR[5] in modo di mantenere per tutti gli elementi nell'evento corrente lo stesso valore frequenziale per la prima riga spettrale inferiore dovuta alla prima modulante (switch=-1) o per la prima riga superiore (switch=1), mentre per switch=0 si usa un'ulteriore regola di decisione.
Se non si sceglie di mantenere la prima frequenza superiore o inferiore costante, queste variano di elemento in elemento, al variare di num.

Vediamo cosa succede per switch=-1: in questo caso si vuole che la sola prima modulante generi righe spettrali tali che la prima riga a sinistra della portante, quella cioè a frequenza PAR[4]-PAR[5], non dipenda da fff, e nella fattispecie da num, e quindi sia uguale per tutti gli elementi generati.

Per fare questo si pone

$$PAR[5] = fff \cdot freq \cdot [-ratio^{\frac{STRIA-num'-1}{STRIA}} \cdot (f_c - f_{m1}) + f_c]$$

in modo che la prima riga inferiore sia alla frequenza

$$PAR[4] - PAR[5] = ratio^{\frac{num'}{STRIA}} \cdot freq \cdot ratio^{\frac{STRIA-num'-1}{STRIA}} \cdot (f_c - f_{m1}) = (f_c - f_{m1}) \cdot freq \cdot ratio^{8/9}$$

e effettivamente la sua frequenza non dipenda da num, e sia costante per tutti gli elementi considerati; la successione di suoni generati al variare di num presenterà diversi valori di frequenze portante e modulante, ma tutti gli elementi avranno la prima riga a sinistra della portante alla stessa frequenza.

Ciò, a livello acustico, comporta che i suoni generati nello stesso evento abbiano sempre una componente fissa, intorno alla quale la seconda modulante andrà a inserire ulteriori componenti: gli elementi dell'evento avranno quindi una caratteristica in comune, pur variando portante e modulanti al variare di fff da suono a suono.
Per gli indici di modulazione in gioco (che si hanno valore medio intorno a 1.5-2) dalle funzioni di Bessel si intuisce che il termine frequenziale generato mediante FM a frequenza uguale alla portante può non essere tanto significativo, mentre possono risultare assai più significativi i termini del primo ordine: è per questo che Chowning ha posto la possibilità di caratterizzare un evento con suoni che mantengano invariate le prime righe vicino alla portante.

Per mantenere invariata la prima riga sopra la portante si utilizza una formula simile, che è facilmente interpretabile dal codice, alla luce di quanto detto.

L'altro caso interessante è quello in cui si abbia switch=0.
In questa circostanza la prima modulante viene **definita senza tenere conto di quali saranno le righe spettrali che genererà una volta combinata con la portante:** si possono verificare due situazioni, discusse in seguito.

Se la portante ha frequenza inferiore a 250 Hz e il parametro f_{m1} è maggiore di 1.618 si presuppone che la modulante (PAR[5]) potrebbe diventare troppo grande rispetto alla portante se calcolata come prodotto fff*freq*f_{m1}, quindi si pone

$$PAR[5] = 1.618 \cdot fff \cdot freq$$

limitando così la prima modulante in frequenza, e **tenendo così lo spettro più raccolto intorno alla portante**.
Se invece la condizione citata non è verificata, si pone direttamente

$$PAR[5] = fff \cdot freq \cdot f_{m1}$$

ottenendo la prima modulante.

La frequenza della seconda modulante è ottenuta semplicemente da

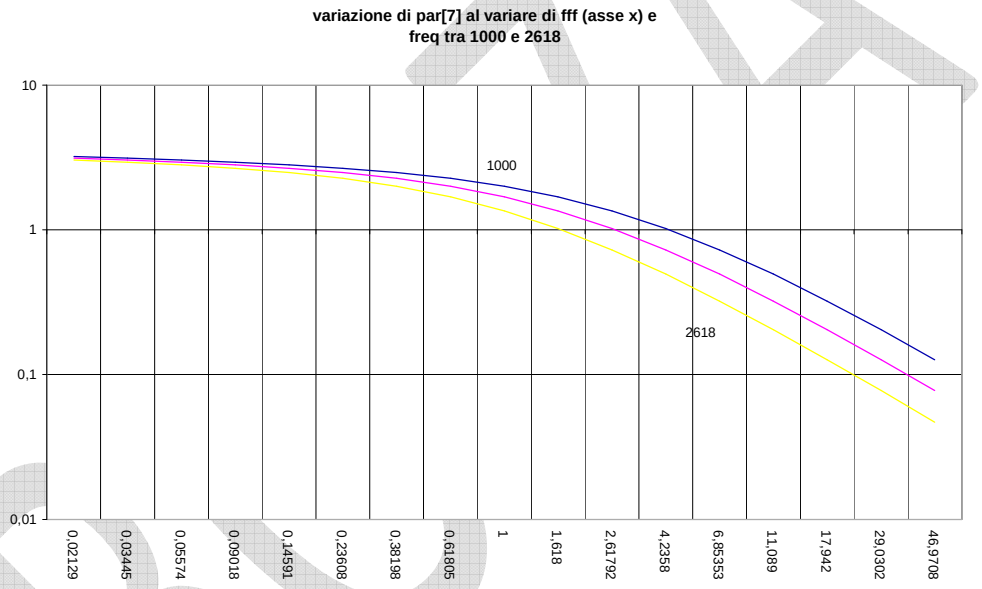
$$PAR[8] = f_{m2} \cdot fff \cdot freq$$

Gli indici di modulazione, si è già intuito, sono definiti attraverso i loro valori minimi e massimi, oltre che con l'impostazione del tipo di inviluppo usato.

Il massimo indice di modulazione della prima modulante (I_{1max}) è ottenuto attraverso una legge dipendente dalla frequenza, simile a quella utilizzata per il calcolo dell'ampiezza del segnale in uscita all'oscillatore principale:

$$PAR[7] = \frac{11 - \log(fff \cdot freq)}{3 + \frac{fff \cdot freq}{1000}}$$

che è tracciata nel grafico seguente al variare di fff sulle fondamentali delle ottave, parametrizzata rispetto alla frequenza base.



Così come l'ampiezza del segnale modulato, anche l'indice di modulazione calerà con la frequenza base, e con la frequenza di scala fff, cioè con la frequenza del suono generato.

Una conseguenza di ciò è che **se le frequenze in gioco sono alte, lo spettro risulterà largo, ma calando gli indici di modulazione massimi, saranno meno le componenti laterali significative, e quindi la banda occupata da ogni strumento rimarrà limitata**.

La scelta di I_{1min} viene fatta in due modi, a seconda che la portante sia maggiore o minore della prima modulante: nel primo caso si pone PAR[6]=PAR[7]/2, mentre se la modulante domina, si pone PAR[6]=PAR[7]/4.
Se la modulante è grande, ciò comporta che la banda del suono sia larga, ma se si permette all'indice di modulazione di scendere ulteriormente si permette alla banda dei vari strumenti di stare stretta.

Per la seconda modulante si pone poi:

$$I_{2min}=PAR[9]=0 \text{ e } I_{2max}=PAR[10]=par[7]+2$$

Anche per la seconda modulante quindi il massimo dell'indice di modulazione è determinato in modo dipendente dalla frequenza, ma il minimo è fisso a 0.

Determinati i parametri frequenziali tocca adesso ai parametri **temporali** legati ai tempi di attacco e di decadimento dei suoni.
Da input, nella definizione degli eventi, si inseriscono due valori per i tempi di attacco, e due valori per i tempi di decadimento: cerchiamo di capire come vengono utilizzati. Per definire le caratteristiche di ttui i suoni dell'evento.

L'idea è che ci sia una variazione continua dei parametri temporali da elemento a elemento, e come variano i tempi di inizio e le durate in base alla sezione aurea, così variano anche i tempi di attacco e di decadimento, da suono a suono: il primo elemento generato avrà un tempo di attacco INIT ATT (attacco all'inizio), mentre l'ultimo avrà un tempo di attacco END ATT (attacco alla fine).
I suoni intermedi avranno tempi di attacco ottenuti per interpolazione esponenziale nel range tra questi due valori.

Per realizzare questa interpolazione si utilizza il parametro interp, definito attraverso

$$\text{int}\, erp = \frac{PAR[1]-beg}{at\_dur}$$

che essendo il rapporto tra il tempo tra l'inizio dell'evento e l'inizio del suono corrente e la durata dell'attacco dell'evento, costituisce una misura della porzione di evento che si è già generata, e assume valori compresi tra 0 e 1.

La formula che genererà i tempi, quindi, dovrà dare in uscita il valore iniziale di attacco o decadimento (es. INIT ATT) per interp=0, quando cioè si tratta del primo suono, e il valore finale (es. END ATT) per interp=1, quando cioè si tratta dell'ultimo suono.

I parametri riguardanti i tempi possono essere immessi da input sia in forma percentuale (relativa), sia in forma assoluta (introducendo il loro opposto).
A questi due modalità di input corrispondono quindi due diverse formule di calcolo dei tempi di attacco e decadimento, che concettualmente, però sono simili.

Consideriamo il calcolo del tempo di attacco per l'elemento corrente, con parametri inseriti in modo percentuale.
La formula che determina il tempo di attacco per questo suono è:

$$PAR[11] = PAR[2]\cdot INITATT\cdot\left(\frac{ENDATT}{INITATT}\right)^{\text{int}\, erp}$$

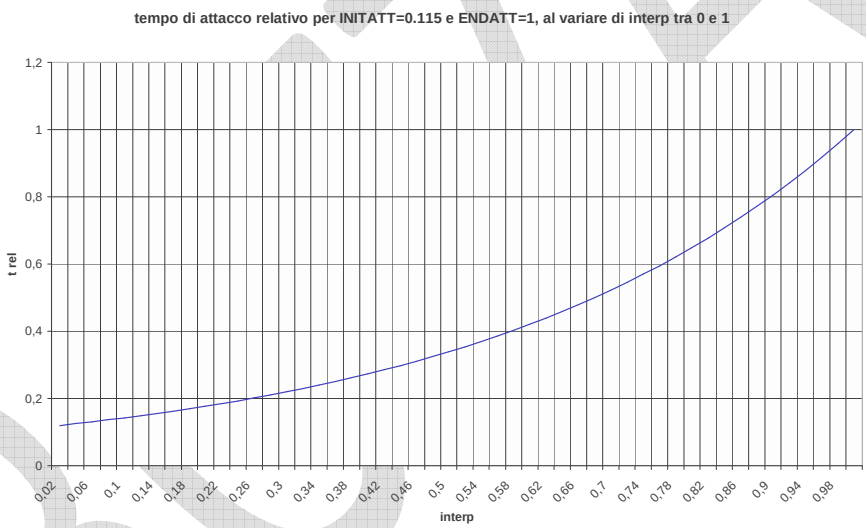
dove nel codice si ha la corrispondenza IATT[event_num,1]=INITATT e IATT[event_num,2]=ENDATT, per l'evento event_num (=2 in Stria).

Si nota subito che per interp=0 vale PAR[11]=PAR[2]*INITATT, mentre per interp=1 si ha PAR[11]=PAR[2]*ENDATT: per valori intermedi di interp si ha l'interpolazione esponenziale.

Una formula identica si ha per i parametri IDCY[], che esprimono i tempi di decadimento, calcolati allo stesso modo e memorizzati in PAR[12].

Nel caso di definizione assoluta dei parametri in ingresso non ci sarà chiaramente PAR[2] a moltiplicare la definizione di par[11], mentre nella definizione di PAR[12] ci sarà il termine PAR[2]-([PAR[11]+0.01), che corrisponde alla differenza tra durata e attacco, arrotondata con il termine 0.01.

Il grafico seguente rappresenta il calcolo del termine relativo del tempo di attacco per valori di interp che variano in modo continuo tra 0 e 1, mentre INITATT=0.115 e ENDATT=1: si nota come l'interpolazione tra i due valori sia esponenziale.



Nella definizione dei tempi di attacco e di decadimento c'è la possibilità di scegliere da input che questi varino con la frequenza, invece di essere legati a valori di tempi immessi dall'esterno: per attivare questa funzionalità basta porre, nell'inserimento dei parametri, atswitch=1.

In questo caso EVENT2 calcola il tempo di attacco relativo attraverso la formula

$$tmpl = 0.333 \cdot \left(\frac{\log(fff \cdot freq)}{\log 2618} \right)^4$$

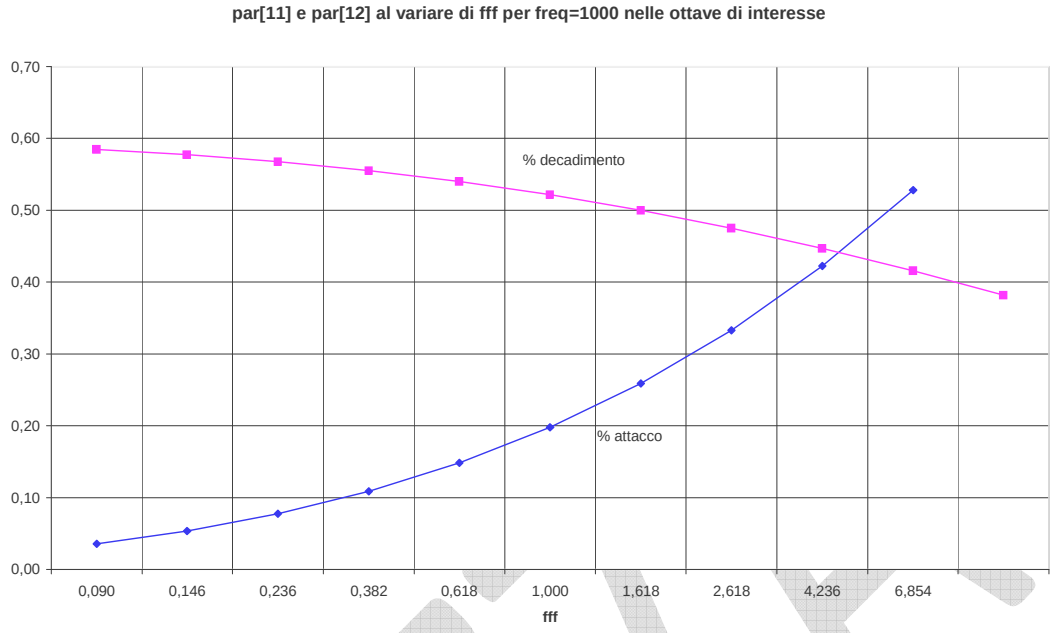
che quindi risulta crescente con la frequenza.
Si avrà quindi

$$PAR[11] = PAR[2] \cdot tmpl$$

e per il tempo di decadimento si pone

$$PAR[12] = PAR[2] \cdot [0.6 - \log(1 + tmpl)]$$

I parametri così ottenuti hanno l'andamento seguente:



Si nota la dipendenza logaritmica dalla frequenza freq*fff delle grandezze in gioco: in ascissa c'è la frequenza di scala fff, mentre freq è scelta costante a 1000.
I valori di fff*freq sono quelli relativi alle fondamentali delle ottave di interesse.

I suoni a frequenza più bassa avranno quindi attacco più breve, mentre i suoni a frequenza più alta avranno attacco più lungo.
Per il decadimento il ragionamento è opposto.

Il parametro PAR[13] definisce la percentuale di skew, e è messo anch'esso in relazione alla frequenza attraverso una funzione simile alle precedenti, che è definita da:

$$PAR[13] = 0.03 \cdot \frac{9 - \log(fff \cdot freq)}{3 + \frac{fff \cdot freq}{1000}}$$

da cui si comprende che la deviazione (skew) cala all'aumentare delle frequenze in gioco.

Dal grafico di questa funzione si nota che ha un comportamento di tipo passabasso.
Considerando la curva della soglia differenziale (jnd) di frequenza, si nota che l'orecchio umano in bassa frequenza ha una peggiore risoluzione frequenziale rispetto all'alta frequenza.
È quindi logico che Chowning abbia scelto di far calare la percentuale di skew con la frequenza, in modo che variazioni in alta frequenza abbiano entità minore, e non risulti esagerata la deviazione frequenziale introdotta rispetto a quella in bassa frequenza.

Il parametro DIS_SCALE può essere inserito in modo che vari nel corso dell'evento o in modo che stia fisso (inserendo l'opposto): nel primo caso si pone

$$dis = (cnt \cdot DIS\_SCALE)^{0.8}$$

ottenendo un valore crescente del parametro di spazializzazione dis al crescere dell'elemento generato, e quindi al passare del tempo nell'evento. **Ciò corrisponde a un avvicinamento o a un allontanamento apparente della sorgente sonora mentre produce la successione di suoni nell'evento.**

In caso di riferimento fisso si pone dis=-DIS_SCALE., ottenendo che la distanza apparente della sorgente non vari nel corso dell'evento.

Definito dis si pone

PAR[14]=1/dis

Il parametro relativo al riverbero, comune a tutti i suoni padri dell'evento, è passato con il parametro PAR[15]=rev.

Per il calcolo dei coefficienti di quadrifonia, si passa alla procedura AZIM il parametro deg, calcolato in base all'angolo di riferimento per l'evento corrente REF_DEG secondo la regola

$$\deg = REF\_DEG[2] + 360^{\left(\frac{cnt}{elements-1}\right)^{0.2}}$$

In questo modo a ogni elemento generato viene assegnato un valore diverso dell'angolo di riferimento, e si introduce una rotazione dei suoni con angolo crescente esponenzialmente con cnt, indice dell'elemento padre generato: ogni strumento avrà quindi sempre angolo maggiore, finchè non si completerà l'intero giro, alla conclusione della chiamata di EVENT2.
Questo è quello che succede in assenza di ricorsioni: vedremo più avanti che le ricorsioni introducono ulteriori spostamenti angolari.

AZIM quindi calcola i parametri di ampiezza sui quattro canali, PAR[21-24].
I parametri PAR[27-30] vengono impostati rispettivamente in base ai valori di FUNAMP[2] (tipo di inviluppo di ampiezza per l'evento 2), 6 (la funzione di skew è fissa), FUNI1[2] e FUNI2[2] (tipo di funzione per i due indici di modulazione per l'evento 2).

Se PAR[27]>6, si va a cambiare I_{2max}, alzandolo in bassa frequenza e abbassandolo in alta frequenza: ciò è legato alle caratteristiche delle funzioni di inviluppo.

Inoltre se la funzione di ampiezza è PAR[27]=3 o 5 si rende PAR[13] uguale a un decimo del suo valore, e si riduce così la percentuale di skew.

Se poi PAR[29]=2 o 8 si moltiplicano i parametri PAR[6], PAR[7], par[9] e PAR[10] per 0.4, riducendo così tutti gli indici di modulazione.

Di queste modifiche finali ai parametri non si comprende bene il significato, non conoscendo a che tipo di funzioni di inviluppo fossero associati queste variabili.

A questo punto tutti i parametri sono stati generati, e si può scrivere la partitura a basso livello (nel file .SCR) per lo strumento nextins, chiamando la procedura WRITE in riferimento a questo strumento.

Finita la generazione di questo suono, **può innestarsi una chiamata ricorsiva**, che genera altri suoni (figli) a partire dai parametri dei suoni originali (padri).
Il verificarsi della chiamata ricorsiva è **subordinato** al verificarsi contemporaneo di **due condizioni**, una dipendente dai parametri di input, e l'altra legata alle caratteristiche del suono che si sta generando.

Per quanto riguarda la prima, si deve notare che il programma contiene la variabile globale intera **nest**, che rappresenta il **numero totale di chiamate ricorsive possibili durante un evento**, e viene impostata da input alla generazione dell'evento: la chiamata ricorsiva potrà avvenire soltanto se il numero di chiamate ricorsive già effettuate, mantenuto nella variabile nest_cnt, non supera il numero di ricorsioni possibili, contenuto appunto in nest.

L'altra condizione perché possa avvenire il nesting ricorsivo è la validità dell'uguaglianza

$$prop = EMBED \cdot sc\_prop$$

con EMBED=0.42 costante globale definita all'inizio del programma principale.
Valendo

$$prop = PROPORTION(which, col, propcnt) \cdot sc\_prop$$

si intuisce che la condizione per la chiamata ricorsiva si trasforma nella condizione

$$PROPORTION(which, col, propcnt) = EMBED = 0.42$$

si vuole cioè che perché l'evento attuale possa essere padre di un evento ricorsivo, il valore di PROPORTION a esso associato sia uguale a 0.42, condizione che si verifica per un solo valore di propcnt per ciascun valore di col: ciò è evidente andando a vedere che nella tabella in PROPORTION c'è un solo riferimento per ciascuna colonna a fib[5]=0.42.

Si ricorda che durante il loop che genera gli elementi col è fisso, e propcnt è del tipo 1,2,3,4,5,1,2,3….

Sarà possibile un solo nesting ricorsivo ogni 5 elementi, e in corrispondenza dell'elemento potenziale padre si instaurerà il ciclo solo se non si è già superato il numero massimo di chiamate ricorsive per l'evento.

Andiamo adesso a vedere che parametri sono passati in ingresso a EVENT2, quando viene chiamata ricorsivamente: la tabella seguente riporta nella colonna di sinistra il nome del parametro in ingresso a EVENT2, nella seconda il valore passato ricorsivamente per quel parametro e nella terza l'interpretazione di ciò che succede.

beg	par[1]	L'inizio dell'evento generato ricorsivamente sarà contemporaneo all'inizio del suono padre che lo ha creato, creando così un'esplosione di suoni in corrispondenza alla chiamata ricorsiva.
dur	dur*prop	La durata dell'evento generato ricorsivamente risulta uguale alla durata dell'evento padre, scalata dal fattore di proporzionalità dell'evento che ha scatenato la ricorsione: in particolare si nota che gli eventi ricorsivi saranno più brevi degli eventi che li hanno generati, creando dinamicità
At_dur	At_dur*prop	Allo stesso modo di dur, anche l'attacco dell'evento è scalato secondo prop, risultando quindi minore del corrispondente tempo per l'evento originale
elements	Se era >9 lo pone a 9, altrimenti rimane uguale	In questo modo si limita il numero di elementi generati ricorsivamente a 9 per ogni chiamata ricorsiva, evitando esplosioni sonore concentrate in brevi intervalli di tempo, che comporterebbero gradi troppo elevati di polifonia e potrebbero generare errori. Il numero massimo di strumenti generabili da un evento quindi è uguale a elements+nest*9.
freq	Freq*fff	La nuova frequenza base è la frequenza che generava le note nell'evento padre, e quindi non rappresenta una potenza a esponente intero di 1.618, ma l'esponente può essere frazionario.
col	=col	Si mantiene la stessa colonna. In questo modo, visto che alla chiamata EVENT2 incrementa di uno la variabile col, si ottiene che i suoni ricorsivi siano generati con l'altra colonna rispetto a quella che li ha generati, ottenendo una maggiore varietà per le frequenze generate.
divx	=STRIA	=9. Non viene mai cambiato nel corso di Stria.
propcnt	Uguale al valore attuale	In questo modo nella generazione di suoni ricorsivi si comincerà a leggere la tabella in PROPORTION dalla riga alla quale è stata interrotta la generazione dei suoni padri.
dense	0	I suoni generati ricorsivamente sono generati sullo spazio a 9 note per ottava.
atswitch	0	I suoni generati ricorsivamente hanno attacco e decadimento dipendente dai parametri introdotti in ingresso, e non dalla frequenza.
space	Scelto	**1. Se space era >1 → space'=1/ratio**

	in base al valore di space per l'evento padre	**2. Se space era <1 → space'=ratio** **3. Se poi il nuovo valore di freq è >1618 si pone in ogni caso space'=1/ratio.** Ricordando che $space = ratio^{power}$, la condizione space>1 equivale alla condizione power>0: l'idea è quindi che se power era >0 (e quindi le note venivano generate in crescendo rispetto alla fondamentale, al variare di num>0, v. INHARM) si pone power=-1, e le note degli elementi ricorsivi saranno generate al di sotto della fondamentale, mentre si farà esattamente l'inverso nel caso che l'evento padre avesse power<0. Così facendo si ottiene che le frequenze su cui sono basate portanti e modulanti (fff*freq) non divergano e non convergano a 0 in seguito a chiamate ricorsive: infatti se nell'evento padre fff portava la frequenza a crescere, nell'evento figlio con il nuovo valore di space (<1) le frequenze tenderanno a calare, e viceversa. Inoltre si pone in ogni caso |power|=1, e quindi si trovano al variare di num note che che al minimo sono spaziate da $ratio^{1/9}$, limitando così la banda dei suoni generati: se infatti power fosse in modulo maggiore di 1 si genererebbero note spaziate al minimo da $ratio^{power/9}$, che genererebbero suoni a spettro più ampio. Inoltre gl eventi generati ricorsivamente avranno estensione spettrale complessiva uguale a un'ottava. La terza condizione ha un significato simile: se la nuova frequenza base è maggiore di 1.618 si pone in ogni caso power=-1, in modo che i suoni ricorsivi generati presentino frequenze contenute, e non si possano avere componenti spettrali troppo elevate per i suoni ricorsivi.
Sc_prop	Sc_prop dell'evento padre	In questo modo il fattore di scala sc_prop relativo ai suoni ricorsivi sarà moltiplicato per sc_prop dell'evento che li ha generati: ciò corrisponde, visto che in generale sc_prop<1 (di solito elements>5), a una riduzione dei pesi prop calcolati per gli elementi figli, che quindi avranno intervalli di separazione tra un suono e l'altro ridotti sia per la diminuzione di at_dur, sia per la diminuzione di sc_prop.

		I suoni figli risulteranno quindi molto più vicini, a confermare l'idea della ricorsione come esplosione sonora.
rev	1.2*rev	Per i suoni ricorsivi viene aumentato il riverbero, in modo che abbiano un impatto spaziale maggiore, visto che tra l'altro durano meno.

Alla chiamata ricorsiva viene incrementata anche la variabile che contiene il numero di nest ricorsivi già effettuati nell'evento, la variabile nest_cnt: in questo modo si rendono possibili solo nest ricorsioni in un evento.
Durante la chiamata ricorsiva, inoltre, si pone la variabile booleana chek al valore FALSE, indicando che è in atto una ricorsione; la variabile viene posta a TRUE alla fine del nesting.

Quando EVENT2 ha finito di eseguire le generazioni degli elementi, prima di finire esegue due ulteriori importanti istruzioni.

La prima va a modificare la variabile globale REF_DEG, ponendola uguale a deg-90: alla fine di un evento (ricorsivo o meno) EVENT2 imposta il nuovo angolo di partenza per l'evento successivo in base all'angolo di riferimento dell'ultimo suono generato, inserendo una rotazione di 90 gradi in senso orario.

In questo modo si ottiene che in un susseguirsi di eventi (normali e ricorsivi, e anche normali, se non si altera da input il valore di REF_DEG) alla fine di ogni evento ci sia una rotazione dell'angolo di riferimento, in base all'ultimo suono generato: **alla rotazione dovuta al susseguirsi dei suoni in un evento si sovrappone la rotazione in senso opposto e più lenta dovuta alla successione di eventi.**

Un ulteriore fattore di dinamismo spaziale è legato alla ricorsione: infatti si verifica spesso che nelle chiamate ricorsive il numero di elementi padri sia diverso dal numero di elementi figli, e quindi ricordando la formula

$$\deg = REF\_DEG[2] + 360^{\left(\frac{cnt}{elements-1}\right)^{0.2}}$$

essendo variato il valore di elements nella chiamata ricorsiva, la rotazione avverrà a una velocità maggiore (elements passa da n>9 a 9**), e i suoni ricorsivi sembrano girare più velocemente.**
Ciò, unito alle minori durate temporali, aumenta l'idea di velocità e il dinamismo: **la ricorsione diventa quindi un'esplosione improvvisa di suoni veloci e vicini temporalmente, che girano intorno all'ascoltatore com maggiore rapidità rispetto ai suoni padri**.

L'altro importante parametro globale variato alla fine di EVENT2 è last_beg, che viene posto uguale all'attuale valore di PAR[1], e quindi indica il tempo di inizio dell'ultimo suono generato.

Riassumiamo ora le caratteristiche dei suoni generati ricorsivamente, considerando una chiamata di EVENT2 da programma principale.
Per rendere più chiaro cosa succede consideriamo un esempio: sia elements=11 il numero di elementi, sia dense=1 il valore del parametro

che esprime la scelta di avere 18 note per ottava e si scelga di determinare le durate di attacchi e decadimenti in modo dipendente dalla frequenza, ponendo atswitch=1.
Supponiamo nest=1.

Appena chiamata, EVENT2 calcola il parametro sc_prop sulla prima colonna, e definito il parametro comincia il ciclo per la definizione degli 11 elementi padri.
Per ognuno calcola il parametro prop, e incrementa propcnt (che parte da 1, alla prima chiamata) continuando con la generazione di tutti i parametri, arrivando a chiamare WRITE per generare la partitura.
Generati i parametri di ogni suono padre, la procedura verifica se sono verificate le due condizioni per la ricorsione, e se in corrispondenza a uno di questi elementi esse valgono, comincia la chiamata ricorsiva.
Questa nuova chiamata di EVENT2 **genererà 9 elementi figli** (element è limitato a 9 per le chiamate ricorsive), di cui il primo comincerà nello stesso istante del suono padre che li ha generati: l'intero evento ricorsivo avrà tempi di durata (dur e at_dur) scalati di un fattore prop rispetto a quelli del suono padre corrispondente, e essendo in generale prop<1, si ottiene che questo evento sarà più breve, risultando **quindi una maggiore concentrazione di suoni durante la ricorsione.**
Saranno anche più brevi le distanze tra un elemento e il successivo, visto che sc_prop' è minore di sc_prop.

La riga di lettura della tabella al primo giro di ricorsione sarà la stessa a cui si è interrotta la generazione dei suoni originali, ma sarà cambiata la colonna, alla chiamata di EVENT2: ciò comporta una diversa distribuzione dei tempi di inizio e delle durate delle voci strumentali ricorsive.
Il vantaggio ottenuto è una maggiore variabilità dei suoni.

La nuova frequenza base è diversa rispetto a quella relativa ai suoni originali, e rappresenta la frequenza del tono su cui erano costruite portanti e modulanti nel suono padre generatore dell'evento ricorsivo: tutti i suoni ricorsivi avranno quindi in comune questo riferimento al padre che li ha generati, mentre sarà variato anche space, per tenere entro certi limiti le frequenze in gioco nell'evento ricorsivo.

I suoni figli inoltre, durando meno, avranno bisogno di maggiore riverbero, mentre le ottave saranno divise sono in 9 note, per questi suoni; gli attacchi e i decadimenti saranno determinati inoltre attraverso i parametri da input, togliendo il legame dalla frequenza.

Finita la chiamata ricorsiva, continuerà la generazione dei suoni, a partire dalla posizione angolare modificata di 90° rispetto all'ultimo suono generato, dall'istante nextbeg calcolato prima di cominciare la ricorsione e dalla riga propcnt dove si era interrotta la generazione.

3.7 Il programma principale

Le pagine successive di codice costituiscono il programma principale, con le parti di inizializzazione delle variabili, l'acquisizione da input dei parametri e la generazione degli eventi.

Nella parte di inizializzazione si pongono i valori di default nelle variabili che potranno poi essere modificate da input: sono definite anche le variabili che sarebbero usate da EVENT0 e EVENT1, come si nota per esempio dal fatto che si impostano i valori di REF_DEG[i] per i=0,1,2. Come già detto più volte i valori delle variabili relativi a eventi diversi dal EVENT2 non sono utilizzati in Stria, e quindi le definizioni di queste variabili sono inutili ai fini del brano.
Lo stesso si nota in tutte le variabili vettoriali.
Tra le variabili con valore di default si ricordano l'angolo di riferimento, il valore di DIS_SCALE, i tre coefficienti f_c, f_{m1} e f_{m2}, i tempi di attacco e decadimento e i tipi di funzioni di inviluppo dei modulatori.

Sono inoltre poste a TRUE le variabili onc e once, che servono per le inizializzazioni di INHARM e PROPORTION.

Inizializzate queste prime variabili, si chiama WRITE, con parametro uguale a 0: WRITE quindi definisce e apre i file che verranno usati per contenere le informazioni sul blocco.

Visto che ogni volta che il programma viene fatto girare acquisisce un blocco di eventi, è organizzato secondo un loop, che comincia subito dopo che i tempi ENDTIME[i] sono stati posti a zero per ciascun strumento: nessuno strumento è quindi inizializzato, e si comincerà dal primo.

Il loop viene ripetuto ogni volta che l'utente decide di aggiungere un evento al blocco, e viene finito quando si decide di finire il blocco.

La parte più consistente del ciclo è l'acquisizione delle variabili, riportate nell'elenco seguente, insieme alle modalità di acquisizione:

event_num	Tipo di evento (0,1 o 2): in stria solo 2 è usato, chiamando EVENT2.
Ev_beg	Tempo di inizio dell'evento, passato a EVENT2
Ev_dur	Durata dell'evento, passata a EVENT2
Ev_at_dur	Durata di attacco dell'evento, passata a EVENT2
ext	Valore compreso tra 0.5 e 0.75, anche se in realtà in Stria si usano anche valori minori. È globale
atswitch	Uguale a 1 per attacchi e decadimenti dipendenti dalla frequenza, passato a EVENT2
CCKK	Si può impostare, attraverso l'acquisizione di questo parametro, di fare un reset della tabella di INHARM nell'evento corrente: basta porre il valore CCKK=1. INHARM, infatti, esegue l'inizializzazione anche in questo caso. Ciò è utili soprattutto per ricominciare dai valori iniziali di num e perm e ricominciare la

	sequenza dei num dall'inizio.
Freqq[event_num]	Frequenza base dell'evento: può essere variata o lasciata uguale
FREQ_SPACE[event_num]	Per la definizione di space in EVENT2, si può impostare questo valore, acquisendo un nuovo valore di power per l'espressione $space = ratio^{power}$.
switch	Solo per event_num=2 (in Stria). Viene impostato secondo i valori visti: ▪ -1 per mantenere costante la prima banda sotto alla portante per il segnale modulato, nei vari elementi, indipendentemente da num ▪ 0 per determinazione normale della prima modulante ▪ +1 per mantenere costante la prima banda superiore per i vari elementi, indipendentemente da num
FR[event_num, 1], FR[event_num, 2], FR[event_num, 3]	Rappresentano i parametri f_c, f_{m1} e f_{m2}, e vengono acquisiti da input.
Funamp[event_num], funi1[event1], funi2[event_num]	Inviluppi di ampiezza e degli indici di modulazione
elements	Numero di elementi
dense	Acquisito solo per gli eventi 1 e 2 (in Stria), per scegliere il numero di note in un'ottava.
nest	Numero di ricorsioni per l'evento 2
IATT[event_num,1], IATT[event_num,2]	Tempi di attacco INIT ATT e END ATT del primo e dell'ultimo elemento dell'evento: possono essere acquisiti in forma relativa o assoluta (inserendo l'opposto). Acquisiti solo se atswitch=0, come deve essere.
IDCY[event_num,1], IDCY[event_num,2]	Tempi di decadimento INIT DCY e END DCY del primo e dell'ultimo elemento dell'evento.
Rev[event_num]	Percentuale di riverbero.
DIS_SCALE[event_num]	Deviazione spaziale: si può rendere la distanza della sorgente variabile o fissa (inserendo il valore opposto)
REF_DEG[event_num]	Angolo di riferimento dell'evento: può essere lasciato invariato o essere cambiato.

Dopo la lettura dei parametri si impostano alcune altre variabili e si fa la chiamata all'evento da generare: in Stria si chiama EVENT2.

Finita la generazione dell'evento, e della relativa partitura si inviano in output il tempo di inizio dell'ultimo strumento generato nell'evento

(lastbeg calcolato in EVENT2), e si calcola anche il tempo di fine dell'ultimo strumento che suona, mandandolo in output.

A questo punto si chiede di scegliere se generare altri eventi o meno, accettando da input la variabile events: se si inserisce un numero diverso da -1 si torna all'inizio del ciclo di acquisizione delle variabili, e si genera un nuovo evento nel blocco corrente.
In caso contrario si chiama WRITE con parametro -1, che chiude il file di partitura (.SCR) e finisce il blocco, mentre vengono chiusi anche i file .MEM e .REP.
Sono indicati in output il numero di strumenti usati e il numero delle chiamate agli strumenti fatte nel blocco.

Il programma così è finito.

Bibliografia:

- FM Theory and applications, J. Chowning and D. Bristow, 1986
- Computer music, C. Dodge and T. A. Jerse, 1985
- Music, cognition and computerized sound, Autori Vari, tra cui J. Chowning per "Perceptual fusion and auditory perspective"
- Sintesi dei segnali audio, di De Poli, Drioli e Avanzini, 1999
- SAIL TUTORIAL, Nancy W. Smith, 1976
- La sezione aurea, C. J. Snijders, 1993
- Listati originali degli algoritmi descritti, e dei file .MEM di output generati
- Comunicazione diretta con John Chowning, per alcuni dettagli irreperibili in letteratura

Indice: