GRAMophone Manual v0.5.2

INTRODUCTION

GRAMophone is an algorithmic generator of music composition. The music is generated using two kinds of formal grammar: Chomsky's regular grammar (or Type 3) for a TOP DOWN approach to the composition and a reduced version of Lindenmayer grammar for a BOTTOM UP approach.

BASIC CONCEPT OF GRAMophone

GRAMophone is partly based on an idea of Jon McCormack's, who invented the idea of a *virtual player* (virtual musician). The *player* in question is associated with a MIDI track, and interprets instructions telling it what to do. Generally, they say *play notes* (send MIDI messages). GRAMophone's *players* together make up an orchestra, which plays a *composition.*

Any number of *players* can play a *composition*, but in practice the hardware used might impose an upper limit.

In general every *player* plays an instrument and each has a different set of grammar rules.

An individual *player* is characterised by a set of parameters which are shared by the whole orchestra and/or a personal parameter set.

The orchestra's parameters consist of:

- the kind of grammar used (Chomsky or Lindenmayer);
- the metronome;
- the measure;
- the number of iterations used in the production process.

Each individual *player*'s parameters consist of:

- the kind of grammar used (Chomsky or Lindenmayer);
- the instrument;
- the MIDI channel associated with the player;
- the number of iterations used in the production process;

A *player*'s notes have a current state consisting of:

- octave
- volume
- duration
- release

These characteristics can be controlled parametrically by a *player* declaring its associated variables.

GRAMophone, then, allows for the non-deterministic generation of music, using either Chomsky or Lindenmayer grammar.

“Give Me A" (The “Hello, World!” for GRAMophone”)

To introduce you to the basic ideas, here is the simplest algorithmic composition that can be generated with GRAMophone: this composition simply generates the note A and is presented through both the Chomsky and Lindenmayer methods.

```
composition “Give Me A” of “Schroeder” {

  //this composition generates the A note with the Chomsky grammar

  grammar chomsky
  tempo 120
  time_signature 4/4
  %
  player Schroeder {
    instrument 0
    %
    @composition->A[,,,];
  }
}

composition “Give Me A” of “Schroeder” {
  //this composition generates the A note with the Lindenmayer grammar
  grammar lindenmayer
  tempo 120
  time_signature 4/4
  %
  player Schroeder {
    instrument 0
    %
    axiom->A[,,,];
  }
}
```

THE KEYWORDS *composition* E *of*

All compositions must begin with the keyword *composition* followed by a string (in inverted commas) containing the name of the composition. This must be followed by the keyword *of* then another string containing the copyright of the piece.

THE COMPOSITION BLOCK

The composition block is placed in brackets. It is subdivided into three sections: one section defines parameters of the composition, one declares and initiates any global variables and an *orchestra* section where the *players* who will 'play' the piece are defined. The first two sections are separated by the % symbol.

THE *player* KEYWORD

Each player is defined with the keyword *player*, followed by an identifier.

THE *player* BLOCK

The player block is placed in brackets and is divided into three sections: one section defines the parameters of the track associated with the player, one declares any local variables for the player and one is for the formal rules. The first two sections are separated by the % symbol.

COMMENTS

In GRAMophone, comments are *C-like*: they must begin with the character pair '/*' and end with the character pair '*/.' There must be no space between the asterisk and the slash. Everything between these pairs of symbols will be ignored by the GRAMophone parser.
Whole lines of comments may also be included. Lines of comments begin with

.the symbol // and end at the end of the line, as in the two initial examples

Section defining the composition's parameters

The parameters shared by all the orchestra's players are declared here. The parameters that may be declared are:

- grammar
- resolution
- iterations
- tempo
- time_signature

This section must end with the % symbol.

grammar

This parameter is obligatory and defines the kind of grammar to be used in the generation. This can be either *chomsky* or *lindenmayer.*

resolution

This parameter defines the number of time units of ¼ duration. If omitted, the default value 480 will be used.

iterations

This parameter defines the number of iterations contained in the generation. Its meaning depends on the kind of grammar chosen, as explained below. If omitted, the default value 1 will be used.

tempo

This parameter defines the composition's rhythm. If omitted, the default

value 120 will be used.

time_signature

This parameter defines the composition's measure. If omitted, the default value 4/4 will be used.

Section declaring the composition's global variables

The variables control the parameters of a note's attributes, as explained below.

Section defining the player's parameters

Each player's personal parameters and variables are declared here. The personal parameters that may be declared are:

- instrument
- channel
- iterations

This section must end with the % symbol.

instrument

This parameter indicates the player's instrument type. GRAMophone's instrument set is the same as that of *General MIDI.* The acceptable range of values is 0 to 127; there are therefore 128 instruments to choose from. A table showing the instrument codes appears below:

0	Piano
1	Brite Piano
2	HammerPiano
3	Honkey Tonk
4	New Tines
5	Digital Piano
6	Harpsichord
7	Clavi
8	Celesta
9	Glocken
10	Music Box
11	Vibes
12	Marimba
13	Xylophon
14	Tubular Bell
15	Santur
16	Full Organ
17	Percussive Organ
18	BX-3 Organ
19	Church Organ
20	Positive
21	Musette
22	Harmonica
23	Tango
24	Classic Guitar
25	Acoustic Guitar
26	Jazz Guitar
27	Clean Guitar
28	Mute Guitar
29	Overdrive Guitar
30	Distorted Guitar
31	Harmonics
32	Jazz Bass
33	Deep Bass
34	Pick Bass
35	Fretless Bass
36	Slap Bass 1
37	Slap Bass 2
38	Syntethized Bass 1
39	Syntethized Bass 2
40	Violin
41	Viola
42	Cello
43	Contra Bass
44	Tremolo String

45	Pizzicato
46	Harp
47	Timpani
48	Marcato
49	Slow String
50	Analog Pad
51	String Pad
52	Choir
53	Doo Voice
54	Voices
55	Orchestra Hit
56	Trumpet
57	Trombone
58	Tuba
59	Mute Trumpet
60	French Horn
61	Brass Section
62	Synthetized Brass 1
63	Synthetized Brass 2
64	Soprano Sax
65	Alto Sax
66	Tenor Sax
67	Baritone Sax
68	Sweet Oboe
69	English Horn
70	Bassoon Oboe
71	Clarinet
72	Piccolo
73	Flute
74	Recorder
75	Pan Flute
76	Bottle
77	Shakhukuhachi
78	Whistle
79	Ocarina
80	Square Lead
81	Saw Lead
82	Caliope Lead
83	Chiff Lead
84	Charang Lead
85	Air Chorus
86	Rezzo4ths
87	Bass & Lead
88	Fantasia
89	Warm Pad
90	Poly Synth Pad

91	Ghost Pad
92	Bowed Pad
93	Metal Pad
94	Halo Pad
95	Sweep Pad
96	Ice Rain
97	Soundtrack
98	Crystal
99	Atmosphere
100	Brightness
101	Goblin
102	Echo Drop
103	Star Theme
104	Sitar
105	Banjo
106	Shamisen
107	Koto
108	Kalimba
109	Scotland
110	Fiddle
111	Shanai
112	Metal Bell
113	Agogo
114	Steel Drums
115	Wood Blok
116	Taiko Drum
117	Melodic Tom
118	Synth Tom
119	Reverse Cymbal
120	Fret Noise
121	Noise Chiff
122	Seashore
123	Birds
124	Telephone
125	Helicopter
126	Stadium!!
127	Gunshot

If omitted, the default *instrument* value 0 is used.

channel

This parameter defines which Midi channel will be associated with the player. There are 16 possible channels. Channel 10 is reserved for percussion

instruments. If omitted, the default *channel* value 1 is used.

iterations

This parameter defines the number of iterations in the generation. Its meaning depends on the kind of grammar chosen, as explained below. If the iterations parameter has been included in the composition declarations, the latter declaration will be ignored.

Section declaring the player's local variables

The variables control the parameters of a note's attributes, as explained below.

Notes in GRAMophone

HOW NOTES ARE WRITTEN DOWN IN GRAMophone

Notes are the first category of terminal symbols GRAMophone.

GRAMophone uses the English notation for notes:

A B C D E F G

The names of notes must be written in capital letters.

The flat and sharp symbols are represented by 'b' and '#' respectively; no space should appear between these symbols and the name of the note: A#, Gb, etc.

NOTE ATTRIBUTES

Notes can have four attributes in GRAMophone: *octave, velocity, duration* and *release.* The octave attribute varies between –2 and 8, while the velocity and release attributes vary from 0 to 127. If the note is written without attributes, then the following default values are used: 3 for octave, 64 for velocity and release. The current default value for duration is a crotchet. In the

example, “Give me A” is written simply as A[,,,]. This means that an A is generated at the third octave, with a duration of ¼ and a velocity and release of 64.

DEFINING THE ATTRIBUTES OF A NOTE

The attributes of a note are defined by writing them inside the square brackets which follow the name of the note, without spaces. A note can have four attributes at most and each attribute type may have only one value. The attributes must be defined in the following order:

1. octave
2. velocity
3. duration
4. release

If all three attributes are not defined, the default value is used for the missing ones. Here are some examples of notes with attributes:

- `C[2, 50+60/2, 240*4,]` – plays a C at the second octave, with a velocity of 80, duration of 960 (minim with a resolution of 480) and a release of 64 (default value);
- `Db[4,,,]` – plays a D flat at the fourth octave, using the default values for velocity, duration and release;
- `F#[,,,]` – use the default values for all the attributes;

  Incorrect examples are:
- `Db[3, 127, 960, 64, x]` – too many attributes (x is a variable).

PAUSE

Pauses are another category of terminal symbol in GRAMophone. They are indicated by the letter R and only take a duration type attribute. If unspecified, the default resolution value is used. Attributes are defined in the same way as for notes. Here are some examples of pauses:

`R[480/2]` – pause with a duration of 240;

`R[]` – use the default value for the attribute of type duration.

CHORDS

Chords are the final category of terminal symbol used in GRAMophone. A chord is a series of notes played simultaneously. In GRAMophone, notes played in a chord are enclosed between two '^' symbols. Here are some examples of chords:

- `^C[,,,]E[,,,]G[,,,]^` - plays a C major chord, using each note's default values.
- `^A[2,80,240,]C[2,,240,]E[2,,240,]^` - plays an A minor chord with duration 1/8, with all notes at the second octave and velocity 64 (default value), with the first note of the chord played with a velocity of 80 and the remaining two at a velocity of 64 (default value).

THE ROLE OF R IN COMPLEX CHORDS

The notes of a chord do not always have the same duration. For example it is possible that, while the note C[2,,1920,] of duration 4/4 is playing, the musician has to play four crotchets in the following order: C[,,,], E[,,,], G[,,,], Bb[,,,]. There has to be a way of telling GRAMophone that the notes C[2,,1920,] and C[,,,] must start at the same time, that E[,,,] must begin after a pause of 1/4, G[,,,] after 2/4 and Bb[,,,] after 3/4. In GRAMophone this is written as follows:

```
^C[2,,1920,]C[,,,]R[]E[,,,]R[960]G[,,,]R[1440]Bb[,,,]^
```

In other words, every note in the chord can be preceded by a pause definition representing the time to wait before playing the note.

It does not matter which order you write the notes down in a chord. The chord in the example above can also be written:

```
^R[]E[,,,]C[2,,1920,]R[1440]Bb[,,,]C[,,,]R[960]G[,,,]^
```

IDENTIFIERS

Some of GRAMophone's language entities, variables, macros and non-terminal symbols in Chomsky grammar for example, must have names by which they can be identified. These names are called *identifiers* and are chosen by the composer.
GRAMophone's identifiers follow the system of identifiers used in the programming language Pascal. In fact an identifier is made up of a letter followed by a sequence of letters or digits. GRAMophone's identifiers must also be written in lower case.

Chomsky Grammar

NON-TERMINAL SYMBOLS

In Chomsky grammar non-terminal symbols are used to give a structure or 'style' to the musical composition. They are written with an '@' immediately followed by an identifier. The Chomsky grammar used by GRAMophone is context free so the head of the production can only be a non-terminal.

THE NON-TERMINAL SYMBOL @composition

This non-terminal symbol, which corresponds to the final composition of a single player, is obligatory.

PRODUCTION OPERATOR

This is defined by the character sequence '->' and separates the head of the production from the body.

BODY OF THE PRODUCTION

This may contain sequences of terminal (notes, pauses and chords) and non-terminal symbols. Each production must end with a semi-colon.

| (OR) OPERATOR

A production may be non-deterministic: in other words it may present two or more choices during generation. The body of a non-deterministic production is made up of the various choices separated by the | operator. For example

```
@non_det->A[,,,]B[,,,]@Seq1|^A[,,,]B[,,,]C[,,,]^@Seq2R[]C[,,,];
```

is a non-deterministic production.

MEANING OF ITERATION IN CHOMSKY GRAMMAR

In Chomsky grammar a production may include cycles, i.e. production bodies containing non-terminal symbols that refer to the production actually being produced. For example:

```
@Sequenza1->B[,,,]A[,,,]C[,,,]@Sequenza1;
```

To avoid an infinite loop during generation, the non-terminal symbol @Sequenza1 is processed an equal number of times to the iterations parameter.

Lindenmayer Grammar

Lindenmayer grammar only deals with terminal symbols and GRAMophone's version can be context-free or work in a polyphonic context. Therefore, single notes or chords can appear at the head of the production. All productions are separated by a semi-colon.

AXIOM

This is the initial production from which generation begins. It is obligatory.

PRODUCTION OPERATOR

This is defined by the character sequence '->' and separates the head of the production from the body.

| (OR) OPERATOR

A production may be non-deterministic: in other words it may present two or more choices during generation. The body of a non-deterministic production is made up of the various choices separated by the | operator. For example

```
A[,,,]->A[,,,]B[,,,]|C[,,,]D[,,,];
```

is a non-deterministic production.

MEANING OF ITERATIONS IN LINDENMAYER GRAMMAR

At each step all the grammar's productions are simultaneously applied to the note string. In this case the iterations parameter represents the number of steps to be carried out.

Use of variables

DECLARATION AND INITIALISATION OF VARIABLES

GRAMophone is able to control the attributes of a note parametrically through the use of variables. These variables are declared in the player's declaration section and may be of the following types: octave, velocity, duration and msb. A variable is declared by writing its type followed by one or more identifiers separated by a comma. The declaration must end with a semi-colon. A

player’s identifier must be declared only once.

The following are correct declarations:

```
velocity x, y;
octave oct, z;
duration w;
```

The following are incorrect declarations:

```
velocity x, x;
octave z;
duration z;
```

Following the declaration section and before the grammar it is possible to initialise variables by means of the = operator. The following is an example of declaration and initialisation:

```
velocity x;
x=0;
```

USING VARIABLES WITH NOTES

Variables are used in note attribute expressions. GRAMophone controls the types within expressions, so it is not possible to add an octave variable to a velocity variable, for example. The following is an example of a note variable:

```
velocity x;
duration z, w;
A[4,x,z+w,].
```

EXAMPLE

```
composition “Crescendo” of “Schroeder” {
```

```
  //this composition generates 64 A notes with a growing velocity

  grammar chomsky
  tempo 120
  tempo_signature 4/4
  iterations 64
  %
  player Schroeder {
    instrument 40
    %
    velocity x;
    x=0;

    @composition->A,x=x+1,,]@composition;
  }
}
```

CONDITIONS

In both Chomsky and Lindenmayer grammars it is possible to define conditions for the variables in the production body. If the condition is true, the production is executed; otherwise it is not. A condition is defined immediately after the name of the production by means of the '?' symbol, followed by one or more Boolean expressions.
The Boolean operators are:

- ! not
- && and
- || or

The relational operators are:

- == equal
- != different
- < minor

- > greater
- <= minor or equal
- >= greater or equal

The following is an example of a conditional production.

```
@battuta?x!=0->A[,x=x-10,,]@battuta;
```

which means: while x is not equal to zero, generate the @battuta production; otherwise do not.

Discography, GRAMophone's library

GRAMophone is able to include external libraries, called *discographies.* To include a discography in a source file, use the keyword *discography* followed by its file name. A discography can be included at any point in the source file, as long as its contents match the position of the source where it has been included.

Macros

Macros can be defined using the keyword *define,* followed by a lower-case identifier and a string placed in inverted commas. Macros must be defined at the beginning of the source composition, before the *composition* keyword. For example, in order to simply write *a* instead of A[,,,], the following macro must be defined:

```
define a "A[,,,]"
```

Functions in GRAMophone

THE *repeat()* FUNCTION

The repeat() function takes an msb type value plus a Chomsky or Lindenmayer sequence. It enables the included sequence to be repeated a

number of times that is equal to the msb type value.

THE *rand()* FUNCTION

The rand() function takes an expression and returns a random value which is less than the value of the expression.

Melodic operators in GRAMophone

transpose()

The transpose() operator takes an msb type value plus a Chomsky or Lindenmayer sequence. It generates a sequence in which all the notes in the relevant sequence are transposed by a number of semitones equal to the msb type value.

inversion()

The inversion() operator takes a Chomsky or Lindenmayer sequence. It generates a sequence in which the intervals between the first and the other notes in the sequence taken are calculated in reverse.

retrograde()

The retrograde() operator takes a Chomsky or Lindenmayer sequence. It generates a sequence which is the contrary of the sequence inserted.

USE OF GRAMophone

GRAMophone works from the command line and has the following syntax:

```
user@host:$gramophone [-c|-d]sourcefile [midifile]
```

where:

`sourcefile` is a file containing the author's formal rules.

`midifile` is the final composition in midi format. Unless otherwise indicated, a file named composition.mid will be generated.

`[-c]` is an option allowing you to control the syntax of the source without generating music.

`[-d]` is an option allowing you to activate debug mode, with video output of the composition and player parameters plus the generated notes.